SKETCH-FINDER: UMA ABORDAGEM EFETIVA E EFICIENTE PARA RECUPERAÇÃO DE IMAGENS COM BASE EM RASCUNHO PARA GRANDES BASES DE IMAGENS

CARLOS ALBERTO FRAGA PIMENTEL FILHO

SKETCH-FINDER: UMA ABORDAGEM EFETIVA E EFICIENTE PARA RECUPERAÇÃO DE IMAGENS COM BASE EM RASCUNHO PARA GRANDES BASES DE IMAGENS

Tese apresentada ao Programa de Pós-Graduação em Computer Science do Instituto de Ciências Exatas da Federal University of Minas Gerais como requisito parcial para a obtenção do grau de Doutor em Computer Science.

Orientador: Arnaldo de Albuquerque Araújo

Belo Horizonte

Outubro de 2014

CARLOS ALBERTO FRAGA PIMENTEL FILHO

SKETCH-FINDER: EFFICIENT AND EFFECTIVE SKETCH-BASED RETRIEVAL FOR LARGE IMAGE COLLECTIONS

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Advisor: Arnaldo de Albuquerque Araújo

Belo Horizonte

October 2014

Fraga Pimentel Filho, Carlos Alberto

P644s Sketch-Finder: efficient and effective sketch-based retrieval for large image collections / Carlos Alberto Fraga Pimentel Filho. — Belo Horizonte, 2014
xxix, 124 f. : il. ; 29cm

Tese (doutorado) — Federal University of Minas Gerais
Orientador: Arnaldo de Albuquerque Araújo

1. Computer Science. 2. Sketch-Based Image Retrieval. 3. Multimedia. 4. Scalability. I. Título.

CDU 519.6*84(043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Sketch-finder: efficient and effective sketch-based retrieval for large image collections

CARLOS ALBERTO FRAGA PIMENTEL FILHO

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. ARNALDO DE ALBUQUERQUE ARAÚJO - Orientador
Departamento de Ciência da Computação - UFMG

PROF. ADRIANO ALONSO VELOSO
Departamento de Ciência da Computação - UFMG

PROF. BENJAMIN BUSTOS
Universidad de Chile - Santiago de Chile

PROF. GUILLAUME GRAVIER
INRIA

PROF. NEUCIMAR JERÓNIMO
Instituto de Computação - UNICAMP

PROF. WILLIAM ROBSON SCHWARTZ
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 08 de outubro de 2014.

Para meus pais Carlos Pimentel e Fátima Maynart, pessoas que tanto amo e que sempre me incentivaram.

Carlos A. Fraga Pimentel Filho

Acknowledgments

First I would like to thank my advisor, professor Arnaldo de Albuquerque Araújo for the support and trust in my work. I also thank very much professor Michel Crucianu and Valérie Gouet-Brunet for advising me in France and guiding the first steps of this work. And to professor Benjamin Bustos for the cooperation in this work. I also thank Luiz Byrro that directly cooperated on the development and running of several experiments for this work.

I am thankful to NPDI and CEDRIC laboratories for receiving me in their groups. To all my colleagues in NPDI, I would like to stress my gratitude to Virginia Fernandes, Sandra Ávila, Júlia Oliveira, Thiago Cunha, Marcelo Coelho, Kleber Souza, Henrrique Silva, Carlos Caetano, Bruno Teixeira, Guilherme Nascimento, Lilian Chaves, Cássio and my colleagues in CEDRIC, Virginie Thion, Marin Ferecatu, Nicolas Travers, Hamza Zrig and Andrei Stoian. It was very pleasant to study and work with you all!

I also thanks my friends in France, Sandra Ávila (yes again), Andrea Silva, Reginaldo, Raimundo Júnior and Vanda. Thank you for giving me support on a foreign country.

I am grateful to the administrative staff of PPGCC/UFMG and CNAM, in particular to Renata Viana, Sônia Borges and Sheila for providing all I needed without obstacles, even with so many requests and lots of paperwork.

I also thank to Maria Teresa for hosting me in Belo Horizonte with all her kindness and to *Cité Internationale Universiraire de Paris* for hosting me in Paris, a so pleasant place to live. I also thank to my colleagues Diego, Rodrigo Rogrigues, Glauber Cabral, Cicero, Yuri and Gustavo that shared the same house in part of my time in Belo Horizonte.

I am also very grateful to CNPq, CAPES, COFECUB and FAPEMIG, for the financial support to this work.

Finally, I thank the support of all my family and a very special thanks to my parents who always trusted and encouraged me to keep following on the search of knowledge.

"Life is what happens when you are
busy making other plans."
(John Lennon)

Resumo

O barateamento dos dispositivos de armazenamento de grandes volumes de dados tem contribuído com o surgimento de grandes bases de dados, todavia, ter a informação armazenada não é o suficiente, pois é importante acessá-la de forma rápida e precisa para que essa informação seja de fato útil. Neste cenário, dentro do conjunto de informações multimídia, o volume de informação visual também vem crescendo e carece de métodos apropriados e eficientes de recuperação. Embora seja possível indexar e recuperar imagens com métodos tradicionalmente utilizados para texto, com base em palavras-chave ou "rótulos", a mídia visual pode ser melhor recuperada a partir de informação visual, uma vez que esta é a sua natureza. Desse modo, a presente tese vem propor um novo método para recuperação de imagens feitas a partir de um rascunho que pode ser rapidamente confeccionado pelo usuário.

Dentre as várias abordagens visuais de recuperação de imagens existentes, o uso de uma imagem de rascunho permite que o usuário expresse a imagem que deseja buscar de forma visual, simples e rápida. O maior desafio desta forma de busca consiste em encontrar uma representação para o conteúdo visual que permita comparar, de forma eficiente, a similaridade entre o rascunho e as imagens da base de dados, mantendo ainda a precisão dos resultados e tendo uma solução escalonável para grandes bases de dados.

Esta tese propõe uma abordagem para recuperação de imagens com base em rascunho onde, tanto o rascunho quanto os contornos das imagens da base de dados são representados e comparados no domínio comprimido da transformada *wavelet*. Assim, apenas os dados mais relevantes, provenientes do rascunho e das imagens são usados na representação e comparação dos mesmos. O uso comprimido da informação é similar aos tradicionais métodos de compressão de imagens com perda e traz como vantagem um reduzido volume de dados para indexar grandes bases de imagens. Consequentemente, um índice pequeno e robusto torna a resposta às consultas mais rápida. Para melhorar a eficácia do método, este trabalho, propõe também, uma comparação dos contornos das imagens mais relevantes da busca fora do domínio comprimido. Essa

comparação verifica a consistência espacial entre os traços do rascunho e os contornos das imagens.

A indexação da base de imagens utiliza índices invertidos, tanto para as informações comprimidas quanto para os contornos das imagens. O uso de índices invertidos melhora ainda mais a eficiência da abordagem proposta. Além do mais, a solução permite que seja possível ajustar o tamanho do índice com base na taxa de compressão de dados, de modo similar ao ajuste que o usuário faz na compressão de imagens, reduzindo a qualidade para ganhar espaço de armazenamento. No índice, esse ajuste afeta seu tamanho e reflete o balanço entre eficiência e precisão das buscas, podendo o seu tamanho ser facilmente adequado aos recursos computacionais disponíveis.

Uma avaliação comparativa entre abordagens tradicionais usando a base de dados de imagens de Paris e um subconjunto da base de dados do ImageNet com 535 mil amostras, revela que a presente solução é superior à abordagem da mesma categoria, porém sendo ao menos uma ordem de magnitude mais rápida. A presente abordagem também é comparada com outros métodos de recuperação de imagem na base de dados do*Flickr15K*. Embora essas outras abordagens utilizem técnicas de histogramas de características visuais e objetivos de busca diferentes do nosso, essa comparação situa a precisão da nossa abordagem entre esses métodos.

Ao final, é apresentada uma aplicação prática de recuperação de imagem com base em rascunho para dispositivos móveis na plataforma *Android*. A aplicação utiliza o método de busca proposto nesta tese e tem uma interface de busca e visualização dos resultados bastante simples e intuitiva.

Palavras-chave: Recuperação de imagem com base em rascunho, indexação multimídia, escalabilidade, processamento de imagens.

Abstract

The cheapening of storage devices for large data volumes has contributed to the emergence of large datasets. However, to have stored information is not enough and to make it really useful. It is important to access needed data quickly and accurately. On this scenario, within the set of multimedia information, the amount of visual information has also been growing and it lacks appropriate and efficient recovery methods. While it is possible to index and retrieve images with traditional methods used for text, based on keywords or tags, the visual media can be best recovered from visual information, since it is the image nature. Thus, this dissertation proposes a new method for sketch-based image retrieval, once that the sketch can be quickly and easily drawn by the user.

Among various image retrieval approaches, the use of sketches lets one express a precise visual query with simple and widespread means. The challenge consists on representing the image dataset features on a structure that allows one to efficiently and effectively retrieve images on a scalable system.

We put forward a sketch-based image retrieval solution where both sketches and selected contours extracted from the images are represented and compared on the wavelet domain. The relevant information regarding to query sketches and image content has thus, a compact representation that can be readily employed by an efficient index for retrieval by similarity. The use of compressed information is similar to traditional lossy image compression methods and it brings as advantage a small size for the dataset index enabling the indexing of big data. Consequently a smaller and robust index provided by compression makes the answer of the queries faster. To improve the effectiveness of the method, this work also proposes a comparison of the most relevant image contours provided by the query performed in the compressed-domain. This comparison verifies the spatial consistency among the image contours and the sketch.

The dataset indexing uses inverted lists either for the compressed information either for the image contours. The use of inverted lists improves even more the efficiency of the proposed approach. Furthermore, with this solution, it is possible to adjust the

index size based on the compression rate, in a similar way it is used on traditional lossy image compression reducing quality to gain space. This adjustment affects the index size and reflects on the balance between effectiveness and efficiency that can be easily modified in order to adapt to available resources.

A comparative evaluation with a traditional method on the Paris dataset and a subset with 535 thousand samples issued from ImageNet dataset shows that our solution overcame effectiveness of traditional methods while being more than one order of magnitude faster. The approach proposed in this dissertation is also compared to other retrieval methods that use bag of visual features on the *Flickr15K* dataset. Although these methods have different query objectives and techniques, this comparison places our approach among them.

Finally, we put forward a practical mobile application for sketch-based image retrieval for *Andoid* platform. The application uses the proposed approach of this dissertation and presents an easy and intuitive interface to create a sketch and visualize the results.

Palavras-chave: Sketch-Based Image Retrieval, multimedia indexing, scalability, image processing.

List of Figures

List of Tables

Acronyms

AMT Amazon Mechanical Turk

BoVW Bag-of-Visual-Words

CBIR Content-Based Image Retrieval

CDI Compressed-Domain Index

CM Chamfer Matching

CMY Cyan Magenta Yellow

CPU Central Process Unit

DCT Discrete Cosine Transform

DFT Discrete Fourier Transform

DT Distance Transform

FMIQ Fast Multiresolution Image Querying

GF-HOG Gradient Field HOG

HSV Hue Saturation Value

IR Information Retrieval

JPEG Joint Photographic Experts Group

KLT Karhunen-Loève transform

OCM Oriented Chamfer Matching

HELLO Histogram of Edge Local Orientations

HOG Histogram of Oriented Gradients

HTML HyperText Markup Language

NIL Number of Inverted Lists

PCA Principal Component Analysis

QBIC Query-by-Image Content

QVE Query-by-Visual Example

RGB Red Green Blue

SBIR Sketch-Based Image Retrieval

SIFT Scale-Invariant Feature Transform

SVM Support Vector Machine

UCM Ultrametric Contour Map

VIR Visual Information Retrieval

VGG Visual Geometry Group

Contents

Chapter 1

Introduction

The advent of digital cameras and the wide spread of mobile devices with camera has encouraged people to take several pictures and to share them on the Internet. Online photo-sharing, such as *Flickr*[1] on the web, and *Instagram*[2] on mobile devices, allows users to upload their photos with diverse content, generating databases with millions or billions of images. For example, instagram has reached 20 billion shared images so far[3]. Allied to video-sharing distribution, multimedia usage has brought a new revolution on media sharing. All those factors have created a wide spread demand for visual retrieval approaches and encouraged researchers to develop new tools and ideas. For example, with the popularity of the touch screen devices, drawing a digital sketch is easier and faster than on traditional computer devices. Thus, query an image by sketch can be a useful tool on mobile devices to find the desired object on images locally stored, or even, on the cloud connecting to a remote appropriate service.

Content-based image retrieval (CBIR) aims to deal not only with the absence or insufficiency of annotations for most of the images, but also to support alternative retrieval approaches, relying on visual perception, which is more appropriate in many scenarios. Within CBIR, *sketch*-based image retrieval (SBIR) aims to return images that are similar to a sketch made by the user (typically a simple set of strokes). SBIR is particularly adapted to situations where the user has a mental image of what he/she is searching. On this scenario, a sketch image is specially useful when the image dataset is not annotated or the user has no similar example image to use as query input. SBIR is also adequate when the user *imagines* a configuration of lines that can not be described in words. As discussed by Smeulders et al. [2000], "*Pictures have to be seen*

[1]Flickr – `http://www.flickr.com/`

[2]Instagram – `http://instagram.com/`

[3]Source: `http://blog.instagram.com/tagged/instagram_news`

and searched as pictures: by objects, by style, by purpose", and why not, by sketch?

There are two important challenges on SBIR: (*i*) finding a relevant visual content representation associated to a similarity measure that allows effective comparison with a query that is not a picture, but rather a drawing made by a user, and (*ii*) making retrieval scalable to large image datasets by building an appropriate index structure able to better exploit the content representation by the similarity measure. We consider that both challenges should be jointly addressed to find efficient solutions for the SBIR problem. Thus, we put forward here a solution where both, sketches and natural contours extracted from the images are represented and compared in the compressed-domain of wavelets, for an efficient query, that also uses the pixel domain for precision refinements. The relevant information regarding to image content (as well as, query sketches) has, thus, a compact representation that can be readily employed by an efficient index for retrieval by similarity. Exploring these two SBIR challenges is the main subject of the present dissertation.

Effectiveness was the major issue on SBIR since the introduction of this research area in early 1990's. However, significant advances on efficiency were only made recently (*e.g.*, [Cao et al., 2011]), making practical applications possible. Among the existing SBIR proposals, we focus on [Jacobs et al., 1995; Cao et al., 2011; Eitz et al., 2011], which we consider particularly relevant.

In [Jacobs et al., 1995], the authors used the wavelet domain to represent the images of the dataset. The wavelet decomposition allows a good image approximation with just a few amount of data. This same property is successfully used for lossy image compression [DeVore et al., 1992]. Typically, in this context, just a few wavelet coefficients with the largest magnitudes are used to represent an approximation of the original image, allowing the construction of a very small index for the dataset. The mentioned approach uses color sketches and the query may be interactive, *i.e.*, while the user draws the query image, a preview of the results is automatically shown. However, this approach is evaluated in a small set of images and the evaluation is performed on painting images instead of real photographs.

In the work presented in [Cao et al., 2011], named Mind-Finder, a black and white line-based sketch approach is presented for a contour-based matching algorithm. This approach estimates the similarity between a sketch and natural contours of index images. The authors indexed the edge segments, including orientation information, and evaluated their method in a collection of more than two million images. The problem of this approach is the high memory cost to hold the dataset index in main memory, which restricts this approach to the size of the available memory.

In [Eitz et al., 2011], the authors presented a benchmark approach for the SBIR

task in large image dataset with a new descriptor based on the "bag-of-features". They also presented an interesting study with human comparison and comprehension about how line-based sketches and real images are similar or not.

The present dissertation is focused on SBIR problem for large datasets. Our goal is to retrieve in these large datasets all images that are visually similar to the query sketch object's shape at similar scale and position *i.e.* affine transform sensitive.

1.1 Motivation

During the last decades, many challenges have been proposed for the CBIR and SBIR problems[4]. Among the challenges of CBIR in a dataset, we can cite: (i) how to find a whole or partial image copy with accuracy on a good response time, even if the image suffered changes on its geometry or it is represented in a compressed version; (ii) how to retrieve images similar to some given example; (iii) how well the query results can be improved with user feedback; (iv) how well may similar natural images be retrieved by a hand-drawn sketch; (v) how quickly an image can be found while browsing. All those propositions, and more specificity the proposition (iv), as well as, the increasing number of publications on this research line motivated this dissertation.

Within CBIR challenges mentioned, SBIR approaches are useful to solve the image retrieval problem in many scenarios. Probably the most important case can be associated to the possibility to graphically specify the query. With a hand-drawn query image, the user can graphically describe the target image that he/she is looking for in the dataset on a free way that other approaches cannot, as free as a hand-draw picture can be done. SBIR approaches are ideal for these cases, where scale, position and rotation of the objects can be pictorially described and this information plays an important role on the query. Comparing with the query by text, for example, this specification is important, once that text information does not bring the desired results in terms of visual object description, but rather, just semantic specification. The goal of SBIR in this case is not just semantic, *i.e.*, the problem is to return images with the desired semantic object as it is specified by the sketch, and not just the object in any position and/or scale. An other important use for SBIR lies when the image dataset is not annotated at all or in part. In this case, an image dataset can be indexed for SBIR without need to annotate all the images, what saves several hours of human or machine work, that in both cases can be imprecise and subjective. However, even in annotated datasets SBIR can also still be used, where both, annotation and visual description

[4]CBIR Challenges – `http://www.benchathlon.net/resources/challenges.html`

complements one each other. Reader may think that visual description can be given by an example image, it is true, but the user needs to first find an image that describe his/her needs, falling down on the chicken-and-egg problem, *i.e.* to make a search the user needs a similar image, but to have this image he/she may need to search on the dataset. In this case, the user can produce a sketch and perform a first query with this sketch, then it is possible to use some results as example to a second query based on an example image. Finally, SBIR can be used as a single tool or can complement other CBIR approaches, *e.g.*, query-by-text, by-example, by-painting or by-icon.

An other motivation for this work is that just a few authors have been addressing the problem of SBIR and some current approaches still keep low effectiveness and/or efficiency, specially in large datasets. Most existing SBIR approaches lack on the problem of scalability due to the issue of efficient sketch-based image retrieval. Achieving scalable and efficient methods for SBIR can be used for several applications, such as:

Web Search: Internet image search engines like *Google*[5] and *Yahoo*[6] are currently the only way to search images on the web. The problem of these tools is that they are usually based on text information close to the image, instead of exploring its visual content. Just recently *Google* introduced the query-by-example, but not yet by sketch. The query-by-sketch can be an useful tool for searching web images complementing the query-by-text, as a primary search to get some example images or even as a single query method.

Personal Photo Search: The organization and management of personal photographic collection becomes more difficult as the collection increases in size. Many people have thousands of photos on their personal computers which are partially or not organized at all. Searching a photo in a collection requires a lot of effort in big collections and time consuming. In this scenario, an application for image query by sketch is an ideal application because the user know the photo the he/she wants to retrieve and can sketch it.

Mobile Image Search: The wide spread of mobile devices equipped with cameras and several gigabytes of memory has generated lots of images on these devices. The facility of drawing a sketch on the touch screen of the mobile device and the growing of the number of images is another motivation for building SBIR tools, adaptable to memory limitations like on these devices. Also, this kind of tool

[5]Google search engine – `http://www.google.com/`
[6]Yahoo search engine - `http://www.yahoo.com/`

can be used for local image retrieval or access a remote server to search images by sketch on the cloud.

While there are many applications for text-based image retrieval, no practical solution exists on the web, personal computer and mobile devices. Thus, those problems and other potential applications motivated the development of the present dissertation.

1.2 Hypotheses

Considering the issues raised on the previous section, our hypothesis is that the use of the compressed-domain index can be successfully used to support sketch-based image retrieval with better efficiency on terms of index size, memory usage, query speed; and at the same time, maintaining at least the same effectiveness comparing to the approaches described in literature.

The lossy image compression, which we intent to use, discards some information, mostly related to spatial image redundancy which is usually not considered crucial. However, the raised questions are: (i) "Is the information lost in the compressed domain impactful to SBIR effectiveness?"; and (ii) "Is the compressed domain efficient on terms of memory and query speed costs?".

A second hypothesis is to index the features using inverted files, in this case, one inverted list per visual word. The benefit we want to achieve is scalability in order to provide an approach capable to support growing image datasets. The verification consists on discovering if the proposed index structure is really scalable and viable for image indexing applied to the SBIR problem.

1.3 Goals

The objective of our approach for SBIR is to retrieve the most similar images to the query sketch input sensitive to affine image transforms. We consider the similarity between a sketch and one image using three main criteria: (i) *shape sensitive*, which means that the contour shape of the target image must be as close as possible to the sketch; (ii) *position sensitive*, which means that image contours should be as close as possible to the sketch strokes in terms of position; and (iii) scale sensitive, which means that the result images should present the objects in a similar scale to the ones drawn on the query sketch.

1.4 Contributions

The main contributions of this dissertation are:

1. A new approach for Sketch-based image retrieval using both the compressed-domain and the pixel domain indexes combining two comparison measures. We combine the efficiency of the compressed-domain of wavelet index, which is our main original contribution, and effectiveness refinement with the pixel domain comparison, based on the Oriented Chamfer Matching (OCM) [Cao et al., 2011] in a single similarity measure. Regarding to the compressed-domain indexing of the image dataset, a new visual word[7] structure is proposed in order to represent and encode image edges in a compact set, what we call here **contour signature**. This visual word structure is based on the wavelet coefficient and presents a very compact information of the image contours. The visual word is composed by the wavelet coefficient position, its sign and the orientation of the edges. These contour signatures make the dataset index much smaller than traditional approaches, as well as, it is possible to set the desired size of the index according to the machine capabilities of memory. It is possible to create smaller indexes as much as necessary, gaining also efficiency, if it is acceptable to reduce a little bit the effectiveness of the queries results, on a similar idea that the user opts to stronger image compression losing some image quality.

2. A new similarity measure is proposed for comparing the contour signatures in the wavelet domain. This similarity is based on computing the number of visual words of the contour signature matched between the query sketch and the target images.

3. An *Android* prototype application for SBIR using the proposed approach here was built, thus testing the SBIR in a real and easy application for final users and filling a gap of SBIR for mobile application on Android.

4. The creation of a collection with more than 100 sketches drawn by several volunteers for the VGG Paris dataset[8]. Also, the construction of a ground-truth for a subset of ImageNet[9] and Paris dataset is an other contribution.

[7] The "visual words" come from a "dictionary" induced by quantizing the feature space of a low-level local descriptor.

[8] Visual Geometry Group – `http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/index.html`

[9] ImageNet – `http://image-net.org/`

1.5 Outline

The chapters that comprise this dissertation are organized as follows.

Chapter 2 - Background Provides a review on the basic concepts of content-based and sketch-based image retrieval. This chapter also presents the main concepts and definition for the image retrieval task. Related topics like: visual features, semantic gap on image retrieval, query modalities, contour detection, as well as, other important concepts used in this dissertation are also presented.

Chapter 3 - Literature Review Provides a wide-range literature review on content-based and sketch-based image retrieval works. The literature review addresses the main works on content-based image retrieval focusing on sketch approaches ranging from the first steps to the latest publications, as far as we know. This chapter also presents on detail two important works close related to this dissertation, as well as, presents the main concepts on image indexing, image retrieval on compressed-domain, inverted lists and the wavelet transform.

Chapter 4 - Sketch-Finder Approach Presents a detailed description of our proposed approach and its main aspects. We describe the similarity measure and the index structure for supporting scalable growing image datasets. The approach is described on its first version and the new features introduced to improve the effectiveness of the method.

Chapter 5 - Experimental Setup and Image Dataset Analysis presents three datasets used in this dissertation to evaluate and compare efficiency and effectiveness of our proposal. We describe a small dataset, mostly used for tuning the parameters of our approach and evaluate effectiveness, a big dataset to evaluate efficiency, and a third dataset to compare our approach with several others. This chapter also describes the methodology used in our experiments.

Chapter 6 - Experimental Evaluation Presents a comparative evaluation of our approach and the work described in [Cao et al., 2011] (Mind-Finder). Several experiments are presented justifying how details on the approach were defined and how parameters were adjusted. Chapter 6 also compares the effectiveness and efficiency of our approach with Mind-Finder in a dataset with 535 thousand images, and with other five approaches, that use local descriptors on the *Flickr15K* dataset.

Chapter 7 - Application: A Mobile Prototype for Sketch-Based Image Retrieval Presents an *Android* mobile prototype application for sketch-based retrieval using our approach on a remote server to process the queries. The structure used to implement this client/server architecture, as well as, the mobile and server applications are presented in details in this chapter.

Chapter 8 - Conclusion and Future Work Concludes this dissertation and presents some directions for future work.

Chapter 2

Background

The term *Information Retrieval* (IR) was mentioned for the first time by Moores [1951] to describe the process through which an user can convert a set of input information request into an useful collection of references. Moores defines information retrieval as: *"embraces the intellectual aspects of the description information and its specification for search, and also whatever systems, techniques, or machines that are employed to carry out the operation"*. Although Moores had defined the IR for text purposes, his definition of retrieval system can be perfectly used for visual information retrieval.

For Gupta and Jain [1997], Visual Information Retrieval (VIR) systems go beyond text-based descriptors to elicit, store and retrieve content information in visual media. Or simply, VIR is a methodology that searches and retrieves images and videos in datasets. The basic premise behind VIR systems is that visual media should be as easily retrieved as we do on textual information.

The visual information associated to visual media (image or video) is divided in two categories: (i) information about the object, called metadata, and (ii) information contained inside the object, called visual features. The Metadata is alphanumeric and usually is expressed as a relational schema in a database while the visual features are used to build the index for image retrieval. These features are mainly obtained using computer vision or image processing and the processes are based on color, texture, shape, image structure and/or spatial relation among the objects [Del Bimbo, 1999; Liu et al., 2007].

In the remainder of this dissertation we introduce fundamental concepts of content-based image retrieval, low level features, semantic gap on image description, query modalities of image retrieval, the user objective when searching an image and contour detection.

2.1 Content-Based Image Retrieval (CBIR)

By nature, image is a complex media because of its unstructured information, which makes the search task extremely difficult. The unstructured information contained inside images is difficult to be automatically extracted, organized, analyzed and transformed on useful and semantic information. The techniques applied to search visual information in a structured index are grouped under the collective name of *content-based retrieval* [Castelli and Bergman, 2004].

Content-Based Image Retrieval (CBIR) is the task of searching and retrieving images from an indexed dataset using its visual features. For Datta et al. [2006], CBIR is any technology that in principle helps to organize digital images datasets by their visual content and according to Venters et al. [2005], CBIR is a general term used to describe automatic or semi-automatic features extraction, indexing, and retrieval of images by their visual characteristics.

2.1.1 Low Level Features

Image features at low level are the basis of CBIR and SBIR approaches. These features can be global or local. On the first case, the entire image is used to obtain the features while on the second case, specific regions are used. Image analysis and pattern recognition algorithms yield the extraction of numeric descriptors which gives a quantitative measure of these features. In the following sections we describe the main features used to stand CBIR and SBIR.

Color Feature

Color is a visual feature which is immediately perceived when we look at some image. In the CBIR domain, color is one of the most widely used features. Thus, color patterns must be represented by a scheme where the chromatic properties are well described.

Color perception is a neurological and physiological stimulation derived from physical electromagnetic radiation that strikes the retina. The human color vision perception, or visible range, varies from $350 - 780nm$ wavelength [Del Bimbo, 1999]. The response of human visual system sensors are composed by two classes of receptors: **cones** and **rods**. The cones are highly sensitive to colors in red, green and blue and each eye has between six and seven millions of cones. The rods, in much larger number, between 75 and 150 millions, are involved on low level illumination sense [Gonzalez and Woods, 2006; De Grandis, 1986]. These sensors give the perception of brightness, chromaticity and saturation.

In computer systems, colors are typically represented in space model of three dimensions. Some models like **RGB, CIE** and **XYZ** are physiologically inspired because they are based on how colors appear to human observer. Hardware-oriented models are based on the characteristics of some devices such as TV monitors and printers. The main models of this category are: **RGB, CMY** and **YIQ**. Another important category of color model is the user-oriented. They are based on the human perception of colors. The tree components of human perception are: hue, saturation and brightness. Some examples of user-oriented color models are: **HSL, HCV, HSV, HSB, MTM, L*u*v*, L*a*b*** and **L*C*h***.

In this way, the color can be defined on a wide variety of color spaces, each one most recommended for its specific application. Description of color spaces can be found in [Del Bimbo, 1999; Gonzalez and Woods, 2006; Plataniotis and Venetsanopoulos, 2000].

Among different approaches, the works of Kurita et al. [1992]; Flickner et al. [1995]; Jacobs et al. [1995]; Smith and fu Chang [1996]; Smith and Chang [1996]; Carson et al. [1999]; Chalechale et al. [2005]; Datta et al. [2007] use color as an important feature for performing CBIR.

Texture Feature

Texture is observed in the structural patterns of objects such as wood, sand, grass, grain, and others. Intuitively this descriptor can provide and measure properties such as coarseness, regularity, smoothness and others. Regarding to process a texture descriptor, a certain region of the image must be considered with its particular properties. According to Belongie et al. [1998], while color is a point property feature, the texture involves a local neighborhood property or set of properties. Many region texture descriptors have been proposed and the tree main approaches are: statistical, structural and spectral [Gonzalez and Woods, 2006].

Statistical approaches yield features that can describe textures like smooth, coarse, grainy and so on. The most common statistical measures in a neighborhood set of pixels are: average entropy, average gray level, standard deviation, histogram moments and uniformity.

Structural analysis deals with the arrangement of image primitives such as the regularity of occurrence of some pattern.

Spectral techniques are based on Fourier Transform, Discrete Cosine Transform (DCT), wavelets or other spectrum analysis that can identify global periodicity or high-energy peaks.

Among different approaches, the works of Flickner et al. [1995]; Ma and Manjunath [1997]; Carson et al. [1999]; Liu et al. [2004]; Datta et al. [2007] use texture as an important feature for performing CBIR.

Shape Feature

The shape of an object describes its physical structure and plays an important role on object recognition and CBIR approaches. Among other advantages, one of the main reasons for the success of shape on CBIR and specially SBIR is its invariance to lighting conditions [Zheng et al., 2011].

For human perception, shape can be enough information for a successful object recognition and/or categorization [Lee and Grauman, 2009a]. Large datasets of object shapes already exist and have application in several areas. Select and retrieve shapes on image datasets that satisfy certain specific constraints is a central problem on shape retrieval and management [Mehrotra and Gary, 1995].

Many strategies can be used to represent shape, among them we have boundary, region, moment and structural representations. Shape can also yield geometric and moment features. From geometric features, we can obtain: perimeter; area; consecutive boundaries; corners; aspect ratio; roundness; number of holes and symmetry. For moment features, we can express shape center mass; orientation; bounding rectangle; best-fit ellipse and eccentricity. Also, shape can be expressed on frequency domain analysis, like Fourier descriptors [Jain, 1989; Mehrotra and Gary, 1995].

In [Mehrotra and Gary, 1995], the authors addressed the problem of similar-shape retrieval. They defined this problem as: "*retrieve or select all shapes or images that are visually similar to the query shape or the query image's shape.*" We can use this definition to describe the objective of the present dissertation approach, where, finding similar shapes described in the sketch image is our goal.

In addition to the mentioned works, we can detach [Flickner et al., 1995; Del Bimbo and Pala, 1997; Ma and Manjunath, 1997; Mezaris et al., 2003] as important approaches that use shape feature to perform CBIR.

Spatial Location

Spatial relationship among objects (layout) plays an important role on image discrimination and understanding [Del Bimbo, 1999]. Relative spatial relationship is more important than absolute spatial location aiming to bring semantic characteristics [Liu et al., 2007]. For example, clouds are expected to be found surrounded by sky, and sea to be under clouds and sky no matter its absolute position.

The spatial relations are categorized in two main groups: **object-based** and **relational-based**. On the first group, the spatial relations are analyzed by an algorithm yielding the objects coordinates and defining their relations. On the second group, a model of the objects position is created and the visual information itself is not used in the model. Objects are symbolically represented and the uninteresting or unneeded spatial relationships are discarded. In relational-based approaches the storage space is improved and the spatial query time is faster due to the use of symbols instead of visual information. Further, the representation of objects is simple in relational models. On the object-based model, the representation can be a simple regular tree [Nievergelt et al., 1984] or some hierarchical data structures like quadtrees, region quadtrees, R-Trees, R^+ Trees and R^* Trees [Del Bimbo, 1999; Samet, 2006].

For the relational-based representation, the relative position of the objects can be represented by a string. An example of spatial content representation is the 2-D string structure described in [Chang et al., 1987]. In this work, the authors presented an algorithm for encoding symbolic pictures in 2-D string representation. This string can also be matched to one each another with the objective of finding relational spatial equivalence. Also, the work [Lee and Chiu, 2003] extended the idea of Chang et al. [1987] and improved memory storage and efficiency in terms of execution time due to their cutting mechanisms.

Smith and fu Chang [1996] presented an approach for querying images by regions and their spatial features attributes. The idea is to find images that present similar spatial arrangements of regions to those diagrammed by the user in a kind of sketch. The images of the dataset are indexed by features like regions, objects size, location and other visual features. Also, we can cite [Smith and Chang, 1996; Carson et al., 1999; Mezaris et al., 2003] as classical works that use spatial location to perform or improve CBIR.

2.1.2 Semantic Gap

Depending on the user needs, visual similarity may in fact be crucial, while for other applications visual similarity may have no importance where the semantic is imperative [Datta et al., 2006].

The semantic gap is the lack between the visual features of the image and their interpretation or understanding of what the features represent, *i.e.*, their meaning. The big issue of the semantic gap is how to link visual feature data to real meaning, which is not an easy task [Datta et al., 2006; Zhang et al., 2012]. Humans are much better than computers on image understanding and description, while computers are better

on measuring properties and retaining this information in long-term memory [Flickner et al., 1995]. In this way, semantics and visual features complement one each other.

Concerning the two main ways of image retrieval (text-based and content-based), in general, there is no direct link between the high semantic level of the text-based approaches and the low level of content-based visual features [Zhou, 2000]. Although we have a lack on a direct link, some approaches using ontologies can help on bridging this gap. In [Mezaris et al., 2003], the authors proposed an approach that uses image segmentation to split it into regions and extract low level features from them describing color, position, size and shape. Regarding to reduce the semantic gap, these features are automatically mapped into an appropriate intermediate level descriptor composing and associating them with an *object ontology*. In this approach, the low level features are hidden to the ordinary user, that deals just with the middle level information modeled as an ontology [Chandrasekaran et al., 1999].

In [Town and Sinclair, 2001], the authors proposed a CBIR system founded on semantically meaningful labeling of images. The image labeling is achieved by training visual features with neural network classifiers [Haykin, 2007], which map segmented image region descriptors to semantic trained classes. In this system, the query is composed by placing blocks of textures, named *semantic regions*, in the desired position. The query is sensitive to the semantic region and their positions.

Another approach is to link web images to their key-words or hash-tags bringing some semantic meaning. The authors [Cao et al., 2010] addressed this semantic problem in a query platform. In this platform, the users place visual object segments on desired position and tag it formulating the query. In [Hu and Collomosse, 2013], the authors inserted keywords on the similarity measure of sketch images to bridge the semantic gap in the SBIR domain.

This dissertation concerns on Sketch-Based Image Retrieval (SBIR), which is a search modality of CBIR as described in the following.

2.1.3 Query Modalities

For a CBIR system, an important issue is how intuitive and easy the user interaction with the system is. The importance of building a human-centered multimedia system has been discussed by Jaimes et al. [2004]. The user interface provides a bridge between the system and the end user. Thus, an easy interface query formulation is important for the success of the retrieval tool. To start the search, the user provides an existing visual model or creates the visual representation of his needs by drawing a sketch or composing a set of visual features selected in a pallet of options. In the context of the

query system input, the user may perform the query using a visual example (Query by Visual Example (QVE)), or text, for text-based queries. Within the query modalities for image retrieval, Del Bimbo [1999] and Venters et al. [2005] distinguish among six different methods: (*i*) query-by-sketch; (*ii*) query-by-painting; (*iii*) query-by-example or query-by-image; (*iv*) query-by-icon; (*v*) query-by-text; and (*vi*) query-by-browsing. Each one of these categories is illustrated in Figure 2.1 and described in the following.

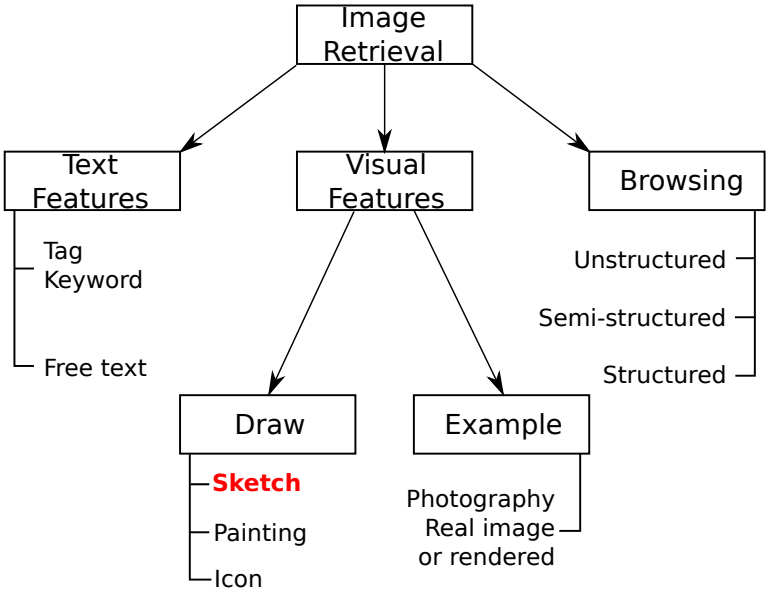

Figure 2.1. Image Retrieval Modalities: in the query-by-text, the user provides keywords, tags or a free text; in the query by visual features, the user offers a real image or draws a sketch, a color painting or even composes an iconic image; and in the query-by-browsing, the user navigates through image thumbnails on a structured or semi-structured classification of images by visual content or semantic labels.

Query-by-sketch: In this technique, the user produces a sketch that consists on a set of contour lines of either an object or a completed scene. The user sketch must represent the visual content of the real image that he/she is expecting to retrieve, considering or not affine transforms, depending on the approach used. This technique presents two main advantages: the first one is the independence of color and illumination, once the sketch has not these information, but just the contours; the second one comes from the simplicity of the sketch picture design, it is fast and easy to produce. Actually, this technique is the only one readily available to all humans due to its facility to draw sketches with simple strokes. Figure 2.2 (a) presents the idea of the query-by-sketch. Kurita et al. [1992] were the fist authors to address this problem, but other examples of this approach

are described by Del Bimbo and Pala [1997]; Flickner et al. [1995]; Müller and Rigoll [1999]; Sciascio et al. [1999]; Chalechale et al. [2005]; Saavedra and Bustos [2010]; Eitz et al. [2011]; Cao et al. [2011]; Tseng et al. [2012]; Hu and Collomosse [2013]. Section 2.1.5 explores this approach with more details, the focus of this dissertation.

Query-by-painting: This type of query is usually employed on color-based image retrieval. It is also a hand-drawn image shaped by the user, however, instead of simple set of lines representing the objects, like on the query-by-sketch, the user produces a painting with color information representing the scene, this painting is similar to a segmented image. Although the painting is also a hand-drawn picture, this kind of picture is a more sophisticated and richer representation than a simple set of sketch lines. In this sort of approach, the user has an idea about the scene that he/she wants to retrieve in terms of similar position, scale and also some approximated chromatic knowledge of the image that he/she is looking for. Figure 2.2 (b) presents the idea of the query-by-painting. Some works that use this type of query are [Jacobs et al., 1995; Flickner et al., 1995; Bimbo et al., 1998; Wang et al., 2011].

Query-by-example or query-by-image: Differently of the query-by-sketch and query-by-painting, on this kind of query, the user applies an existing image instead of drawing one picture. The example image can be either an internal dataset sample or an external image obtained anywhere else. This scenario is indicated for queries based on color, texture or structure properties of the example image. Also, the query-by-example is indicated when the user has some similar image on his/her hands or when the visual content can not be easily reproduced. Sample images can either be provided by the answers of previous queries performed by query-by-sketch or any other query method. The image can be a real photography, a realistic rendered image or any realistic image representing the real world. In this type of query, the same visual features used to index the image dataset are extracted from the example image aiming to perform the search and similarity comparison of the images within the dataset. Figure 2.2 (c) presents the idea of the query-by-example. Some approaches that perform query-by-example are: [Faloutsos et al., 1994; Jacobs et al., 1995; Flickner et al., 1995; Ma and Manjunath, 1997; Carson et al., 1999; Zhou et al., 2010].

Query-by-icon: The iconic querying is based on a high-level image retrieval concept. On this query method, the user composes a pictorial image by selecting and

placing icons at the desired position. The icons are predefined object/person categories selected on a pallet of options. The icons relative spatial position, orientation and scale are analyzed by a parser checking their correctness and translating the visual specifications into a formal symbolic query sentence [Del Bimbo, 1999; Venters et al., 2005; Gupta and Jain, 1997]. Several approaches in the last years have been proposed for iconic query like: [Chang et al., 1987; Angelaccio et al., 1990; Del Bimbo et al., 1993; Papantonakis and King, 1995; Aslandogan et al., 1996; Smith and fu Chang, 1996; Chavda and Wood, 1997; Lee and Whang, 2001]. Figure 2.2 (d) presents the idea of the query-by-icon.

Query-by-text: On this sort of image retrieval, the user offers as input a textual description of the desired content. The image dataset must have an index, a priory annotated by human intervention or even machine. The human annotation provides a rich high level feature (concepts), such as keywords and tags. This is the most popular kind of image retrieval procedure and it is widely used on web search engines like *Google* and *Yahoo*. Instead of human annotation, an web image search engine uses the surrounding text of the picture to relate image and semantic context [Datta et al., 2007]. Automatic image annotation can also be used in order to bridge the gap between the low level of visual features and the high level image concepts for the task of image retrieval by text. The latest development on automatic image annotation extracts semantic features using machine learning techniques [Zhang et al., 2012]. Figure 2.2 (e) presents the idea of the query-by-text.

Query-by-browsing: This is the simplest method of image query. The query-by-browsing can be performed on three ways: (*i*) unstructured browsing, where the user can scroll through a complete view of image thumbnails; (*ii*) semi-structured, where the user can scroll and select an example image to perform a query-by-example, and then, iteratively scroll again on best classified results, until the desired image be found; and (*iii*) structured, where image clusters are previously defined. The image cluster can be defined by text or by visual information and these clusters can be hierarchically organized or not. On the structured browsing, the user can navigate through hierarchical classes of image features or labels until find the desired images. One example of structured browsing is the ImageNet[1] dataset [Deng et al., 2009] where the images are organized according to the WordNet [Miller, 1995] hierarchy. In [Paiva et al., 2011], the authors presented a

[1]ImageNet – `http://www.imagenet.org/`

hierarchical visualization of images by separating groups and subgroups of visual similarity. An illustration of structured browsing of the ImageNet dataset is presented in Figure 2.3. The viability of browsing is strongly dependent on the size of the image dataset, *i.e.*, it is very costly to find images in big datasets. Although most users perform unstructured browsing to find personal images in folders of small collections, there is no meaning on browsing unstructured datasets of millions of images. Examples of semi-structured browsing manners are presented in [Sclaroff et al., 1997] and [Laaksonen et al., 2000].

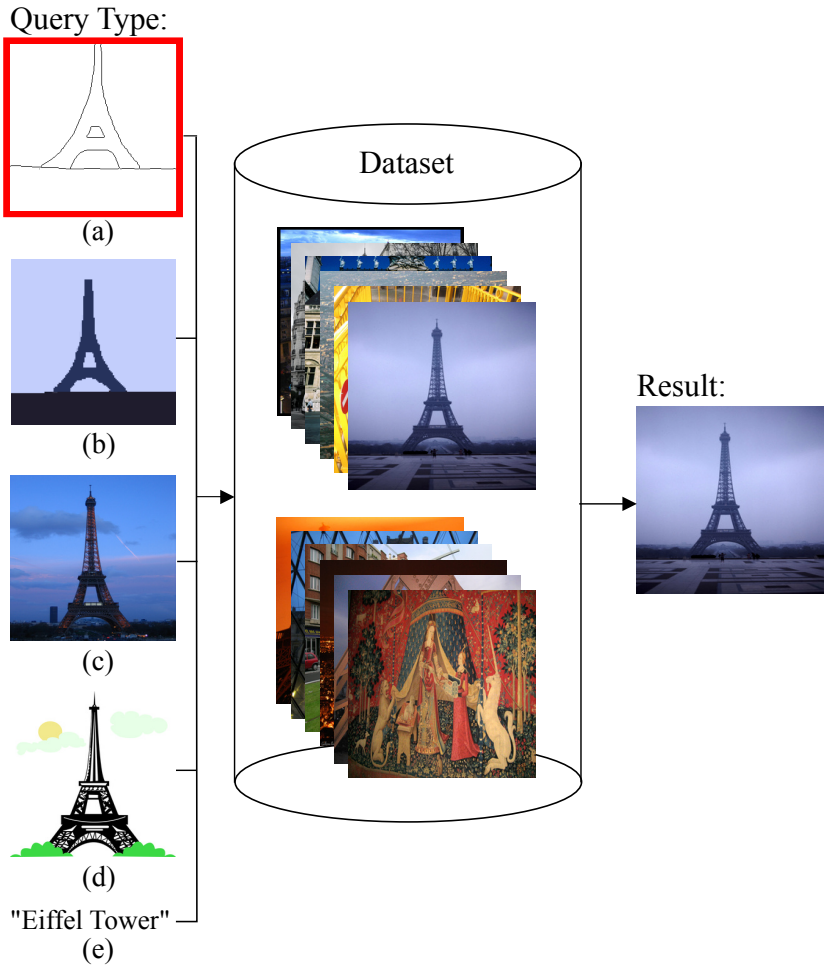

Figure 2.2. Types of Image Retrieval: (a) query-by-sketch – the user draws a set of strokes describing the shape of the objects or scene; (b) query-by-painting – the user draws an image with chromatic information; (c) query-by-example – the user offers an existing image as input query; (d) query-by-icon – the user composes a query definition by selecting and placing predefined icons; and (e) query-by-text – the user defines keywords, tags or gives a free text description for image retrieval.

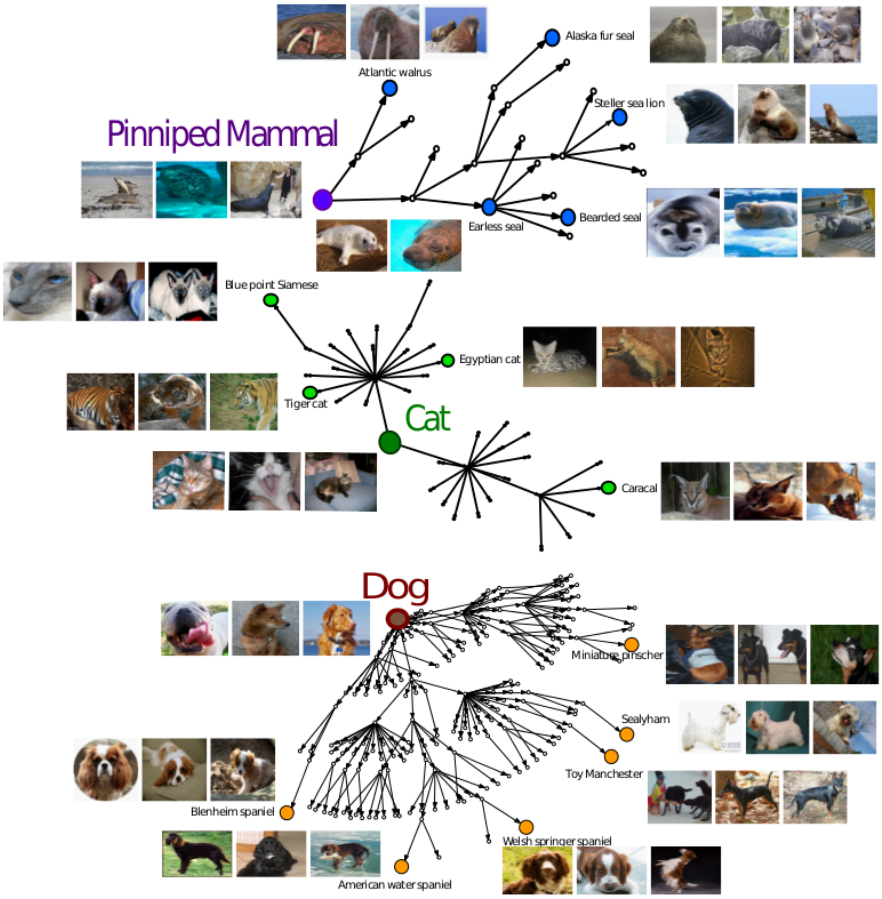

Figure 2.3. Structured browsing using the WordNet ontology for the ImageNet dataset hierarchy structure. The objects are organized by the Wordnet ontology on semantic terms. For example, to find tiger images, the user browses by selecting "animal", then "cat", and finally "Tiger cat". – Source: ImageNet.

In order to improve the effectiveness of the queries and bring more facilities to the user, the input can be composed by two or more query modalities, simultaneously or not. An example is the work presented in [Cao et al., 2010], where the user can compose the query with example image fragments and tags or keywords to better express his expectation. Also, Hu and Collomosse [2013] complemented the query-by-sketch with semantic tags to enhance the results and bridge the semantic gap. In [Datta et al., 2007], the authors presented four different scenarios for querying image associating visual features and image tags. Additionally, they presented an approach for tagging images in near-real-time in order to help meaningful retrieval. The authors affirmed that their approach ensures high precision and recall annotations and outperforms traditional approaches on the query scenarios proposed by them.

2.1.4 User Objective

According to the objective, the user may perform a query with one of the three objectives: (*i*) *search by association* (*ii*) *aimed search*; and (*iii*) *category search* [Smeulders et al., 2000; Datta et al., 2006]. The objectives are detailed on following:

Search by association: in this category, the user has no exact target image, but he/she can define some feature like a range of color and/or other features to query and then browse on the resulting images. After the first query, if necessary, the user can refine it and repeat the process until get satisfied.

Aimed search: this query is used when a specific image is sought, for example when it is necessary to find wholly or partially image copies or when the user has a very specific image on mind, like an image he/she knows exists in the dataset and wants to retrieve it [Zhou et al., 2010].

Category search: this query is used when an image representing a semantic category class is sought, for example, a sketch or an example image of an aircraft is used in order to retrieve any image containing at least one aircraft.

The query-by-sketch can fall into any one of these three user objectives, however, depending on the objective, one or other approach can be more adequate. The present dissertation proposal and the works [Cao et al., 2011; Tseng et al., 2012; Jacobs et al., 1995; Sun et al., 2013], are mostly congruous for "search by association" and/or "aimed search" objective, while the approaches [Eitz et al., 2011; Hu and Collomosse, 2013] better falls in the "category search". It is important to distinguish the user objective for different approaches in order to better compare and evaluate them on their respective groups.

2.1.5 Sketch-Based Image Retrieval (SBIR)

Humans have been using sketches to represent the real world since prehistoric times, like on the use of petroglyph, or rock art as illustrated in Figure 2.4.

Finding images by sketch is not a trial task. Several factors make this problem difficult to solve. The "query" image is typically very different from the real "target" image in the dataset, specially when the user is not very skillful or patient on drawing a digital sketch image, a problem known as **sketch uncertainty** [Sun et al., 2012]. Figure 2.5 presents some examples of lions sketches obtained from the [Eitz et al., 2012] dataset. It shows that the human perception of the real world may not be very close to the reality in terms of shape depiction.

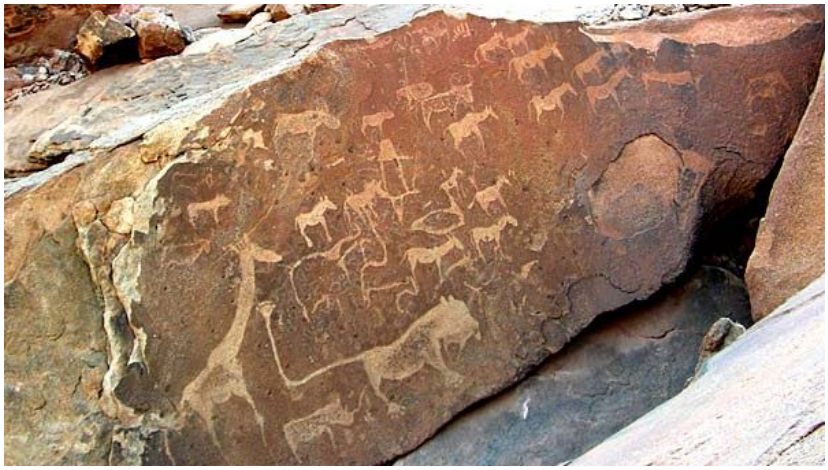

Figure 2.4. Petroglyph sketch example. Sketches have been used since prehistoric times to represent objects.

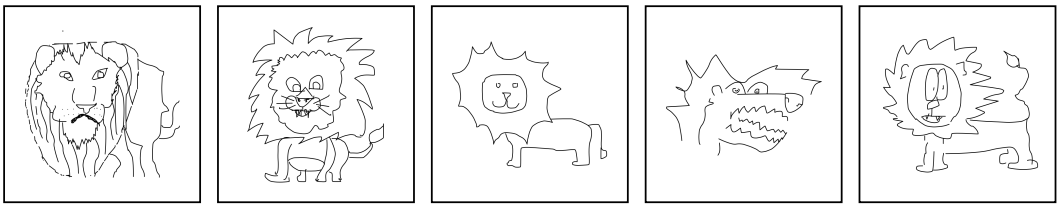

Figure 2.5. Lion Sketch examples. These examples show that the human perception of the real world may not be very close to the reality in terms of shape depiction. These sketches were obtained from the sketch dataset of Eitz et al. [2012].

In [Bird et al., 1999], the authors affirmed that the users blamed their own sketch when the query result was not successful. Also, according to Hu and Collomosse [2013], the user limitation on depictive representation of objects limits the accuracy of the query-by-sketch. Further, **intra-class shape variation** and specially **inter-class shape ambiguity** can disrupt a query-by-sketch [Sun et al., 2012]. Intra-class shape variation is defined as the variety of shapes that the same object may have, depending on several issues like, position, rotation of the object and even the shape variety of the object nature. Two class examples that varies a lot are chair and bridges. Figure 2.6 presents some examples of intra-class shape variation. Inter-class shape ambiguity is related to the same, or similar shape, that different objects may have. This is a really difficult problem for SBIR because it is a semantic issue rather than shape. Figure 2.7 show some examples of inter-class shape ambiguity. Finally, digital input devices, like mouse, do not help performing good sketches.

In order to outline this problem, the SBIR approach must be tolerant to some

Figure 2.6. Examples of intra-class shape variation. The shapes of the same object may present a big diversity.

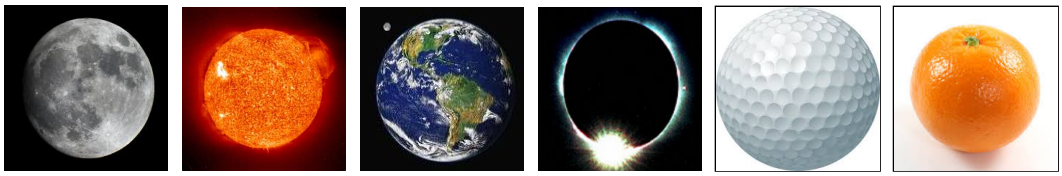

Figure 2.7. Examples of inter-class shape ambiguity. Different objects might present quite similar shapes.

distortions on the sketch, such as position of the object, scale and rotation. If the sketch is color based, *i.e.*, a painting sketch, the approach must be tolerant to some chromatic shift once that the user does not know the exact color of the objects, although he/she has a general idea of what is looking for.

On SBIR approaches, where the query is a simple black and white hand-draw draft, texture and color loose their original ability to serve as content keys, remaining from the traditional CBIR approaches, the importance of shape and spatial features to represent the natural image contours.

2.1.6 Sketch Datasets

In order to evaluate a SBIR approach it is necessary to have a dataset of sketches and a ground-truth for it. Aiming to evaluate the SBIR approach proposed by this dissertation, we created a sketch dataset for the Paris dataset as described in Section 5.3.

Besides Paris sketch dataset collection, other datasets were built for SBIR evaluation, like the sketches for the *Flickr15K* dataset with 330 samples in 33 different categories [Hu and Collomosse, 2013] and the work of Eitz et al. [2012]. The sketches of Eitz et al. [2012] were collected using the Amazon Mechanical Turk (AMT) with 1,350 unique workers spending a summed time of 741 hours. The average time to draw each sketch was 86 seconds and the initial collection of 22,500 sketches was reduced, after a data verification, to 20,000. The authors categorized the sketches in 250 groups where each one has exactly 80 sketch samples. As far as we know, this is the biggest collection of sketches available in this domain.

2.2 Contour Detection

Contour detection and image segmentation are not identical but related problems on image processing and computer vision. Among several approaches, contour detection plays an important role on image shape detection. In order to perform sketch-based image retrieval, the detection of natural image contours is crucial in the visual feature description for indexing an image dataset, which is the main core of this work. If the image dataset is not well described by its relevant contours, the retrieval task based on this entity becomes ineffective. Also, the contours can not be over detected because the excess of contours may act like noise.

Looking for a good contour detector, we found in literature a relevant approach, state of the art on image segmentation and contour detection as far as we know [Arbelaez et al., 2011]. This approach presents a high performance on contour detection combining local and global information on its method. The contour signal is then transformed into a hierarchical representation of closed regions or segments that preserves the original contour quality.

The basic principle of the approach described in [Arbelaez et al., 2011] is based on the previous work of [Martin et al., 2004], who defined a function $Pb(x, y, \theta)$ whose objective is to predict the probability of a boundary at each pixel (x, y), with orientation θ, by measuring the difference in local image features of brightness, color and texture channels.

For each channel feature, the function $Pb(x, y, \theta)$ is computed placing a circular disc centered at pixel (x, y) and split by a diameter at angle θ. On each half-disc, an histogram of intensity is computed, generating two histograms g and h. On the following, the X^2 distance between the histograms g and h are computed as a function $Pb(x, y, \theta)$. Figure 2.8 shows an example.

This Ultrametric Contour Map (UCM) algorithm, from the work [Arbelaez et al., 2011], is used in our approach to estimate the contours of our image datasets described in Section 4.1.1. Some examples of final contour detection using the UCM algorithm are presented in Figure 4.2(b), obtained from the natural images shown in Figure 4.2(a).

2.3 Conclusion

This chapter presented a review on the most important concepts used in this dissertation. Among these concepts, we presented the concept of content-based image retrieval and sketch-based image retrieval. This review also presented some basic concepts used in our proposed approach. The visual features are the basis of image description for

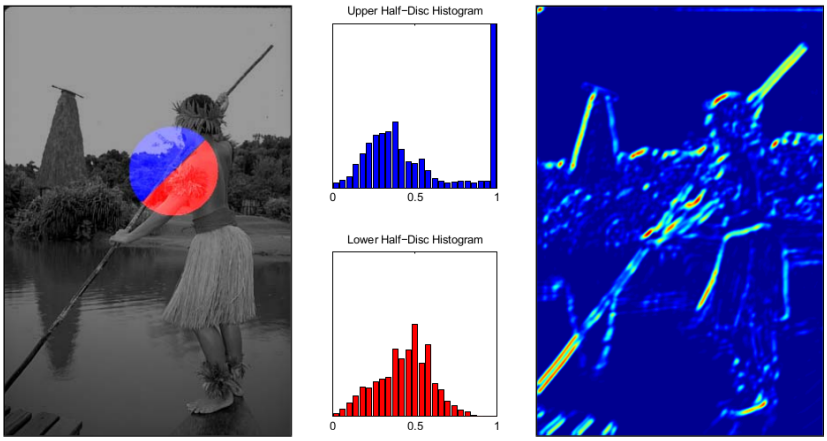

Figure 2.8. Contour detection with oriented gradient histograms. On left, the image with a centered disc at pixel (x, y), where the contour existence probability is been estimated. The disc is split on 45° by a diameter in half blue and half red segments. On each half part of the disc (blue and red) is estimated a histogram of the analyzed feature (brightness, color or texture). On the middle of this figure is shown the histogram obtained by each half disc of the left image and on right, the contours probability obtained on the orientation $\theta = 45°$ are presented, where red mean high probability of contours and blue represents low probability. Source: Arbelaez et al. [2011]

CBIR and SBIR approaches and the main low level features are color, texture and shape. Using the visual features, a visual image retrieval system can perform image retrieval in four main ways: by sketch; by example; by painting; and by icon. For sketch-based image retrieval, the most important low level feature is based on shape descriptors. The low semantic discriminating power of the low level features is known as semantic gap, and the use of textual information can help on bridging this gap.

A good algorithm for contour detection plays an important role on sketch-based image retrieval, once that this is the main feature used to describe and compare images and the input sketch. As shown in the experiments on Section 6.2.1, the threshold used on the image contours presents a relevant impact on the effectiveness of these approaches.

Chapter 3

Literature Review

This chapter provides a general review of the main approaches for content-based and sketch-based image retrieval. This review goes from the first approaches and extends until the most recent works as far as we know. We present an overview of each approach presenting its strong and weak features. Two other important approaches that guided the development of our proposal are described in details: Mind-Finder [Cao et al., 2011] and Fast Multiresolution Image Querying [Jacobs et al., 1995]. Within the description of these approaches, we also present their similarity measures, image indexing and inverted lists concepts, as well as, the wavelet transform for image indexing in the compressed-domain.

Inside the general modalities of CBIR (see Section 2.1.3), the SBIR domain is a less stated field comparing to the others. Among the existing SBIR approaches, we stand out the works of Del Bimbo and Pala [1997]; Chalechale et al. [2005]; Saavedra and Bustos [2010]; Eitz et al. [2011]; Cao et al. [2011]; Tseng et al. [2012]; Hu and Collomosse [2013]. Here, we briefly discuss the main CBIR approaches with emphasis on SBIR domain.

3.1 Related Works

One of the first approaches, using the idea of sketch as input to search image collections, is the work described in [Kurita et al., 1992]. In this paper, the authors addressed the problem of SBIR with an approach named Query-by-Visual Example (QVE). For this work, the authors used a dataset of art gallery paintings. The query and the images of the dataset are transformed into an abstract representation of the image edges maps in 64×64 pixels. Aiming to compare the images, the similarity measure estimates a local correlation of the sketch edges with the corresponding neighborhood of the target

images. However, the low resolution images lead to low effectiveness, and the use of only art galleries images restricts the scope of this proposal.

Probably, one of the most known works in CBIR and SBIR using low level features is the IBM's Query by Image Content (QBIC) system described in [Flickner et al., 1995]. Developed in the middle of the nineties, the QBIC is a tool that allows users to query visual content not only in images and but also in videos. Regarding to visual features on which the users can search images, it is possible to explore: color, texture and shape. In QBIC system, the query can be performed by example or by sketch.

In the query-by-painting domain, we stand out the work of Jacobs et al. [1995]. This work presented a general approach for query-by-painting and query-by-example (see Section 2.1.3). A color hand drawn painting or a scanned image can be used to search in the image dataset using the same algorithm, however with different configuration parameters. The approach used wavelet decomposition, aiming to obtain a set of coefficients that composes the **image signature**, as the authors named. This image signature is composed by the n most significant negative and the n most significant positive wavelet coefficients. On the query time, these image signatures are compared in the compressed domain, by an image query metric that operates on this domain. Essentially, the similarity measure compares how many high energetic wavelet coefficients, provided by the query image, are matched to their similar coefficients on the image dataset index. The metric includes weights of parameters according to the coefficient spatial position that can be tuned using statistical analysis. Algorithms are simple, and the resulting image signature requires just a little storage overhead. Because in this dissertation the technique of using the most significant wavelet coefficients is also applied, a detailed description of this approach is presented in Section 3.3.

A hybrid system named *VisualSeek* is presented in [Smith and Chang, 1996]. Indexed visual features and spatial query methods were integrated in this work. Regarding to spatial relations, the user composes the query by specifying the spatial arrangement of color regions. The search engine considers not only the absolute position of the objects, but also, the relative position among them. In such work, the authors affirmed that the spatial information improves the performance of retrieval over non-spatial approaches.

An elastic shape similarity for sketch-based image retrieval was shown in [Del Bimbo and Pala, 1997]. The proposal is based on elastic deformation of the user sketch regarding to match objects in the images of the dataset. This approach is an attempt to approximate how human perception works in shape similarity perception of objects. Spatial relationships among objects in multi-object queries play an important role on this work. Furthermore, the elastic matching is integrated with arrangements

to provide scale and partial rotation invariant, however, the main drawback of this work, is the expensive computational cost.

A content-based image retrieval using three dimensional models is presented by Assfalg et al. [2002]. The authors described an approach to create 2D images from 3D models and apply them to query-by-example. Image is constructed by the user in a 3D environment where he/she can place the desired objects and define their colors and textures, then, the search engine retrieves the images based on the color and texture similarity.

A sketch-based image retrieval method is shown in [Chalechale et al., 2005]. In this approach, two abstract edge representations are obtained, one from the strong edges of the natural images, and another from morphological thinned outline of the sketch image. Angular-spatial edge distributions obtained from the images are, then, employed to extract a compact set of features using the Fourier Transform. Extracted features are invariant to rotation and scale, and robust against translation as well.

In [Saavedra and Bustos, 2010], the authors described a method based on histogram of edge local orientations named HELLO (Histogram of Edge Local Orientations). Local orientations are computed based on directional fields of fingerprints, in the context of biometric processing. The proposed description is invariant to scale and translation transforms. Aiming to achieve rotation invariance, the authors applied two different normalization processes, one using Principal Component Analysis (PCA), and other using polar coordinates. Also, in [Saavedra and Bustos, 2010], the user sketches do not need to be performed using continuous strokes.

A recent method for sketch-based image retrieval that inspired the approach of the present dissertation is shown in [Cao et al., 2011]. In such work, named Mind-Finder, the authors presented a contour-based matching algorithm to assess the similarity of sketches and natural image contours. In this approach, the authors converted each image of the dataset into a set of oriented *edgel* descriptors. The *edgels* are contour image pixels or edges, plus their line orientation. Following, this visual word is indexed using inverted lists. In the query step, the edgels of the query sketch are matched to the dataset index using a variation of the *Chamfer Matching* (CM) [Brogefors, 1988] algorithm. In this work, the variation of the CM is named *Oriented Chamfer Matching* (OCM) [Stenger, 2004], because of the addition of the orientation information. Two million image composes the dataset of this approach, indexed in main memory with 6.5 GB. However, indexing large datasets is limited to the available memory space, and the amount of data to process a query makes the approach very expensive on very large datasets. More details about Mind-Finder approach are presented in Section 3.2.

Germane to the evaluation performance on large scale SBIR approaches, the

authors Eitz et al. [2011] designed a benchmark and built a dataset of more than 30,000 ratings, according to human criteria of how much a pair sketch/image is similar. The evaluation study was performed in a controlled environment where the users were instructed on how to perform the task. This analysis demonstrates that humans match sketch and image pairs in a similar way. Also, the approach presents a new local descriptor based on shape and Scale-Invariant Feature Transform (SIFT) descriptors [Lowe, 2004]. Those descriptors are then adapted to a "bag-of-features" [Sivic and Zisserman, 2003] approach for SBIR domain.

Lee et al. [2011] presented a work for guiding the drawing of objects through a shadows projection on the canvas drawing area. This shadow is iteratively projected while the user draws his/her sketch. The shadows are projected based on the average of dataset image contours similar to already drawn strokes. The objective of this work is to help people to draw more realistic sketches and can be potentially used on SBIR systems to improve the pictorial user skills, and consequently, improving the efficacy of the results.

Concerning to a collection of sketches, Eitz et al. [2012] presented a study with more than 20,000 sketches distributed over 250 categories. Authors described a sketch representation approach in a form of "Bag-of-Visual-Words" for training a multi-class Support Vector Machine (SVM) for classifying sketches. Human *vs.* machine sketch classification is confronted and the results showed that humans can correctly identify the object category 73% of the time, while computers presented an accuracy of 56%. Furthermore, in this work, the authors affirmed that automatic recognition of the sketch category can be potentially employed to improve the SBIR task.

Similar to [Eitz et al., 2012] with respect to sketch recognition, is the work published by Sun et al. [2012]. In such work, the authors developed a general sketch recognition system that used one million web clipart images to compose a dataset. This dataset is the knowledge base of the recognition mechanism. The system used an adapted SBIR approach to retrieve cliparts based on the input sketch and relate the web text surrounding the clipart results to infer the sketch class.

A similar work to this dissertation, where the objective is to save memory space and computational cost is presented in [Tseng et al., 2012]. This SBIR application is also based on the approach of Cao et al. [2011], and the authors focused on the idea of SBIR for mobile devices. This approach also splits the image edges on six quantized orientations, and then, the authors apply the Distance Transform (DT) on each oriented channel. Following, high dimensional DT features are projected in a compact hash bits. Retrieval performance is competitive to Cao et al. [2011], requiring only 3% of the memory storage, according to authors. However, this work presented

just a few results and did not use a common dataset and sketches to compare their approach with others.

Sun et al. [2013] used a similar hash bits idea to the one presented in [Tseng et al., 2012]. The objective of this work is to perform a big data sketch-based image retrieval, indexing more than 1.5 billion images. The basis of this work is the same of Cao et al. [2011], and its goal is to build a compact index that represents the oriented image contours as in Tseng et al. [2012].

Hu and Collomosse [2013] presented another approach adopting "Bag-of-Visual-Words". Histogram of Oriented Gradients (HOG) descriptor is adapted to *Gradient Field HOG* (GF-HOG). Authors tested eight common distance measures frequently used in text ("Bag-of-Words") retrieval. The approach is evaluated using 33 shape categories search patterns. Further, the authors incorporated semantic keywords aiming to enable the use of annotated sketches for image search. As shown in the paper, the keywords relevance inserted in the similarity measure increased the results in semantic terms.

Other approaches for content-based and sketch-based image retrieval are described in [Faloutsos et al., 1994; Matusiak et al., 1998; Carson et al., 1999; Engel et al., 2011]. In next section, we describe in more details two important works that introduce some basis for the present dissertation SBIR proposal.

3.2 Mind-Finder

The work presented by Cao et al. [2011], named Mind-Finder, plays an important role in this dissertation once that it introduces an important idea. The idea is to represent the image edges in six quantized orientations and then use this information on SBIR task. In Mind-Finder, the authors present the *edgel*, a short name for **edge pixel**. The edgel is a triple (x, y, θ) containing not only the spatial pixel coordinate (x, y), but also, its quantized orientation (θ). Mind-Finder similarity measure is based on how many similar edgels between a sketch and the target image are computed, the more similar edgels found, the more similar the sketch and the image are. The similarity comparison considers the edge coordinate and orientation. Following sections describe Mind-Finder approach in more details.

3.2.1 Similarity measure using Oriented Chamfer Matching

The main idea of Mind-Finder to measure the similarity of a **query sketch** $\mathcal{Q}$ and the natural image contours of some **target image** $\mathcal{T}$, is based on computing the number

of edgels that matches a similar spatial position $X_p = (x_p, y_p)$ and orientation (θ). In such work, the authors propose the idea of **Hit Map** to compare the edgels. The hit map is an extended spatial area, or neighborhood of the edgels inside a radius r, given in pixels. It is used to match the neighborhood edgels of $\mathcal{Q}$ to the edgels of $\mathcal{T}$, and vice versa. Figure 3.1 (b) presents some examples of hit maps obtained from the sketch 3.1 (c).

The **Oriented Chamfer Matching** [Stenger, 2004] is an extension of the **Chamfer Matching** (CM) [Borgefors, 1983; Brogefors, 1988] algorithm, where the image edges are matched considering its spatial position. To improve performance, the authors use an inverted list index of image IDs stored in main memory. In MindFinder, the memory space and query time are linear to the dataset size. Figure 3.1 presents the idea of edgel Oriented Chamfer Matching in the Hit Map. Natural image contours are presented in Figure 3.1 (a). These contours are segmented in six quantized orientations, aiming to match its edgels to the respective oriented hit map of the sketch (Figure 3.1 (b)).

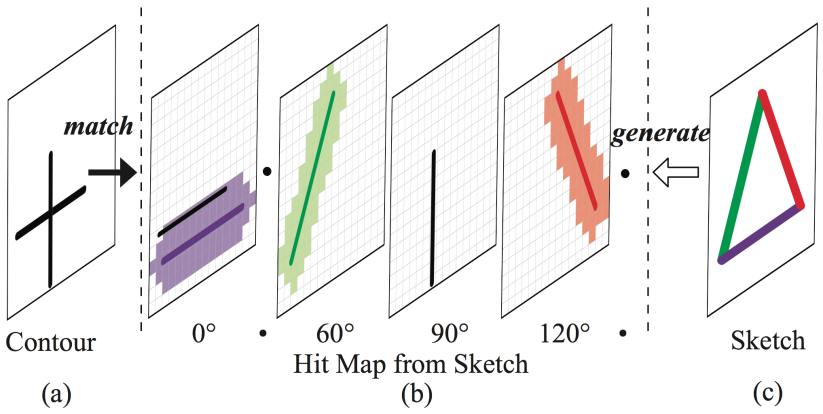

Figure 3.1. Oriented Chamfer Matching Diagram. In (a) the natural image contour is presented in black lines. In (b), four samples of Hit Maps with radius $r = 3$ pixels are shown. The oriented chamfer matching between the natural contour and the sketch is presented in black, where matches occur when a black segment intersects a colored area. The sketch Hit Map is presented in 4 parts based on their orientation in different colors, *i.e.*, purple (0°), green (60°) and red (120°). Empty hit map channels (90° and 150°) are hidden in this figure. In (c), the query sketch is represented in color lines just to distinguish the stroke orientatin, although in fact the sketch is a binary image. Source: [Cao et al., 2011].

The OCM algorithm computes the similarity in the number of edgels at similar position inside a given radius r and at same orientation θ. Considering the set of edgels $\mathcal{L}_\mathcal{Q}$, representing the oriented sketch contours, in which the position of the

edgels $p \in \mathcal{L}_\mathcal{Q}$ is denoted by $X_p = (x_p, y_p)$ and its gradient orientation is denoted by θ_p. Also, considering the oriented hit map represented by $\mathcal{M}_\theta^\mathcal{Q}$ of a query sketch $\mathcal{Q}$ with N_Θ channels ($\theta \in \Theta$), r is the tolerance radius and $|\mathcal{L}_\mathcal{Q}|$ the number of edgels of $\mathcal{Q}$. Equation 3.1 gives the matching of edgels and the similarity measure from the query sketch $\mathcal{Q}$ to the target image $\mathcal{T}$ is denoted by Equation 3.2.

$$Hit_\mathcal{Q}(p) = \begin{cases} 1 & \exists_p \in \mathcal{L}_\mathcal{Q}(\parallel X_\mathcal{Q} - X_\mathcal{T} \parallel_2 \leqslant r \ \& \ \theta_\mathcal{Q} = \theta_\mathcal{T}) \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

$$Sim_{\mathcal{Q}\to\mathcal{T}} = \frac{1}{|\mathcal{L}_\mathcal{Q}|} \sum_{p \in \mathcal{L}} Hit_\mathcal{Q}(p) \tag{3.2}$$

In fact, the OCM is performed in two ways, from the sketch to the target image $Sim_{\mathcal{Q}\to\mathcal{T}}$, and vice versa $Sim_{\mathcal{T}\to\mathcal{Q}}$ (Equation 3.2).

Finally, the similarity measure considering the two ways of OCM is given by Equation 3.3.

$$Sim_{\mathcal{Q},\mathcal{T}} = \left(Sim_{\mathcal{T}\to\mathcal{Q}} \cdot Sim_{\mathcal{Q}\to\mathcal{T}}\right)^{1/2} \tag{3.3}$$

The reason for the comparison be performed in two ways is that the similarity $\mathcal{T} \to \mathcal{Q}$ is different from $\mathcal{Q} \to \mathcal{T}$ in Mind-Finder, *i.e.*, all edgels of $\mathcal{Q}$ can be contained inside the hit maps of $\mathcal{T}$, but it is possible that all edgels of $\mathcal{T}$ may not be contained in the hip maps of $\mathcal{Q}$, and vice versa. Figure 3.2 illustrates this idea, in such figure, all edgels of the query sketch are contained in the hit maps of the first image, middle top in the figure, using the similarity measure $Sim_{\mathcal{Q}\to\mathcal{T}}$, but all edgels of the first image are not contained in the sketch using $Sim_{\mathcal{T}\to\mathcal{Q}}$. For the second image, right top in the figure, both comparisons ($Sim_{\mathcal{Q}\to\mathcal{T}}$ and $Sim_{\mathcal{T}\to\mathcal{Q}}$) match all edgels.

3.2.2 Image indexing with inverted lists

An **inverted list** or inverted file is an index data structure for storing a map of content. Inside an inverted list, the roles of records and attributes are reversed, *i.e.*, instead of listing the attributes for a given record, also known as **forward list**, the inverted list presents a list of records for a given attribute [Knuth, 1998].

The use of inverted lists has been applied with success for indexing data for text retrieval [Brin and Page, 1998; Badue et al., 2001] and several other applications. Multimedia data can also be benefited by the use of inverted lists, as on Mind-Finder and the present dissertation. This strategy speeds up the query and works quite well because it acts like a shortcut to go directly on the desired attribute or feature, therefore,

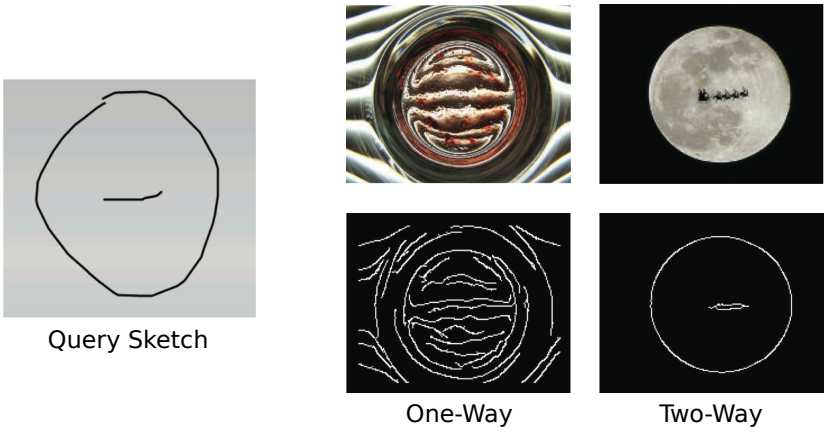

Figure 3.2. Mind-Finder query in one-way and two-way similarity measure. On the left, the query sketch; on the middle, the image query result (top) and its natural contour (bottom) using one-way OCM similarity; and on the right, the query result (top) and its natural contour (bottom) using two-way OCM similarity.

accessing all documents where the feature is present, without having to read all index.

On Mind-Finder, each inverted list represents an edgel word feature. This list is composed by Identification code (ID) of the image where the edgel is present. Figure 3.3 presents the scheme of inverted lists for the edgel indexing.

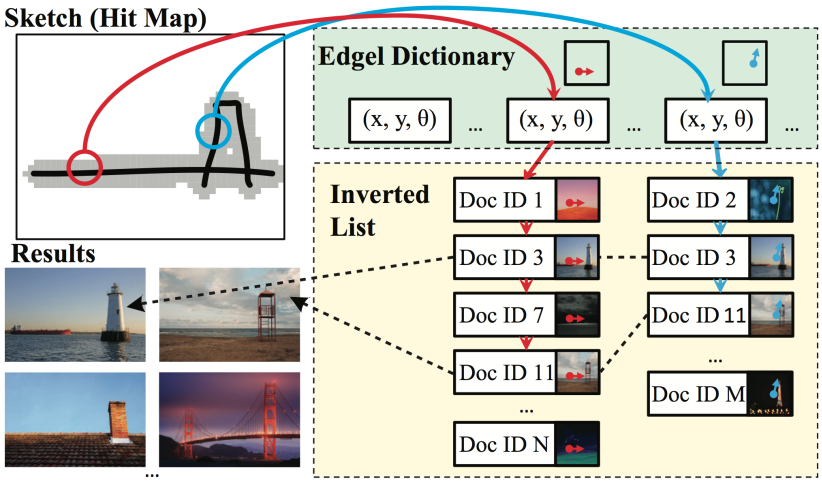

Figure 3.3. Edgel Index using inverted lists. In this figure, the Hit Map of a sketch (top left box) and the mapping of each edgel hit to the edgel dictionary (top right box) are presented. In the edgel dictionary, each word is associated to its respective occurrences in the image dataset by an inverted list of image IDs (right bottom box). Source: [Cao et al., 2011].

In Figure 3.3, the **Edgel Dictionary** represents all edgel words on the approach configuration, and each edgel word has one inverted list of image IDs, where that specific edgel is present. The configuration of Mind-Finder is presented on the following.

Image Preprocessing in Mind-Finder

In the practical work, Mind-Finder indexed a dataset of two million images. These images were downsampled to 200×200 pixels, which the authors considered a good balance between preserving structure information and storage cost. To obtain the natural image contours, they used the algorithm described in [Martin et al., 2004], and the index for two million images consumed 6.5 GB of memory.

3.3 Fast Multiresolution Image Querying

In this section, we describe the work named Fast Multiresolution Image Querying (FMIQ) [Jacobs et al., 1995]. Although this work is built for query-by-paint and query-by-example, which are not a sketch, this work is important because it presents a relevant contribution using the technique of compressed-domain index, like the index proposed on this dissertation. The compressed-domain index of Jacobs et al. [1995] is based on the wavelet image decomposition, and the basic idea of this work for image querying is to directly compare matches of wavelet coefficients in the compressed-domain. Basically, the more matches between two images, the more similar they are. Subsequent sections describe this work in more details.

3.3.1 The Compressed-Domain Index

The voluminous nature of visual information, like image and video, requires the use of compression techniques for storage, transmission and indexing for visual retrieval. In particular, visual information compression works very well with lossy methods, once that the visual information in image and video usually has a lot of redundant data to represent them. Thus, this kind of data with strong visual redundancy on space and time (for videos) is a key point where the compression techniques work eliminating them.

A straightforward approach of **Compressed-Domain Index** (CDI) [Castelli and Bergman, 2004], is to apply existing compression techniques, and to use derived compressed information to build indexes for visual information retrieval [Benois-Pineau et al., 2012]. The main advantage is its inherent efficiency of compression, reducing the complexity of visual data processing.

Typically, the features used for indexing visual data are directly extracted from the original image pixels. However, it is possible to use the compressed data to extract visual features to index the visual content. As advantage, the Compressed-Domain

Index has the inherent quality of efficiency and reduced complexity, due to its smaller amount of data to process. This strength is used as the basis for saving memory and CPU cost in order to speed up the retrieval of the approach presented in this dissertation.

The CDI is classified in two main categories: transform-domain and spatial-domain methods. Transform domain methods are generally based on Discrete Fourier Transform (DFT) [Villasenor, 1993], Karhunen-Loève Transform (KLT) [Jain, 1989], Discrete Cosine Transform (DCT) [Ahmed et al., 1974; Rao and Yip, 1990], subband or Discrete Wavelet Transform (DWT) [Antonini et al., 1992; Daubechies, 1992; Mallat, 2008; Stollnitz et al., 1995a,b]. Spatial-domain techniques are based on fractal image compression [Fisher, 1995] and vector quantization methods [Idris and Panchanathan, 1995]. Table 3.1, obtained from [Castelli and Bergman, 2004], presents a comparison among several compression techniques.

Although Table 3.1 presents the DFT translation invariant of the coefficients as an advantage, for our propose of sketch-based image retrieval, this feature is considered a disadvantage, once that our user objective retrieval (Section2.1.4) and goal intend to retrieve images with objects without or just a few translation.

In the work of Jacobs et al. [1995], the compressed information of the image is obtained from the wavelet transform domain and represented by its main quantized coefficients. A little introduction to wavelets as a lossy image compression tool is presented on the following.

3.3.2 The Wavelet Transform

Wavelet is a mathematical tool for hierarchical function decomposition. Function representation in the wavelet domain has a coarse information plus the details in different levels of hierarchy [Antonini et al., 1992; Daubechies, 1992; Mallat, 2008; Stollnitz et al., 1995a,b]. Some authors say that the wavelet decomposition is like seeing the forest, the trees, and the leaves, depending on the detail level you are looking at the function.

Among the wide spectrum usage of the wavelet decomposition, we can cite diverse applications to image processing and computer vision. In [Loupias and Sebe, 2000; Tsai, 2012], the authors used the wavelet decomposition, in order to obtain salient points used for CBIR. Wavelets are also applied with success to image compression [Christopoulos et al., 2000], surface reconstruction from contours [Meyers, 1994], and others. On image compression, the wavelet is used as a tool to encode the image using the most significant coefficients, like in JPEG2000 image format [Christopoulos et al., 2000]. In

Table 3.1. Compressed-Domain Indexing Approaches. Source: Castelli and Bergman [2004].

Technique	Advantages	Disadvantages	References
DFT	Uses complex exponentials as basis function. The magnitudes of the coefficients are translation invariant. Spatial domain correlation can be computed by the product of the transforms.	Lower compression efficiency	[Villasenor, 1993]
DCT	Uses real sinusoidal basis function. Has energy compaction efficiency close to optimal KLT	Block DCT produces blocking artifacts	[Ahmed et al., 1974; Rao and Yip, 1990]
KLT	Employs the 2nd order statistical properties of an image for coding. Provides maximum energy compaction among linear transformations	Is data dependent: basis images for each subimage has to be obtained, and hence has high computational cost	[Jain, 1989]
DWT	Numerous Basis functions exist. There is no blocking of data as in DCT. Yields, as by-product, a multiresolution pyramid. Better adaptation to nonstationary signals. High decorrelation and energy compaction. Low computational cost for fast *Haar* DWT	Chip-sets for real-time implementation is not readily available	[Antonini et al., 1992; Daubechies, 1992; Mallat, 2008; Stollnitz et al., 1995a,b]
Vector Quantization	Fast decoding. Reduced hardware requeirements makes is attractive for low power applications. Asymptotically optimum for stationary signals.	A codebook has to be available at both the encoder and decoder, or has to be transmitted along with image. Encoding and codebook generation are highly complex.	[Idris and Panchanathan, 1995]
Fractals	Exploits self-similarity to achieve compression. Potential for high compression	Computationally intensive, hence hinders real-time implementation.	[Fisher, 1995]

general lines, compression consists on just using the most significant coefficients and discarding the less significant ones, what usually, are majority. Therefore, just a few coefficients are enough to represent the main structure of the image. Although this technique presents loss of information, this loss can be controlled, and even not be noticed by most part of the users. Figure 3.4 presents four image reconstructions using the *Haar* wavelet using the most n significant coefficients. As shown in Figure 3.4, the more coefficients are used, the more the details of the original image are preserved.

In [Jacobs et al., 1995], the authors applied the *Haar* wavelet decomposition not for compression, but for indexing the image dataset with the most significant coefficients for CBIR. These coefficients represent a few percentage of the raw image

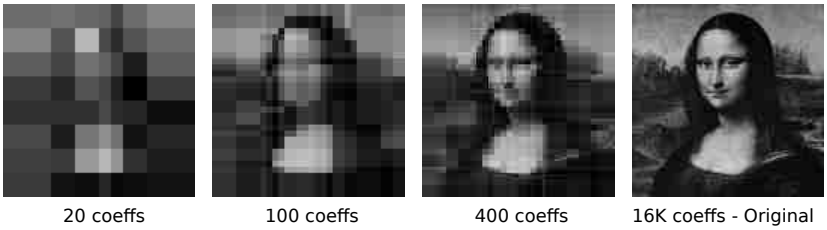

Figure 3.4. Image reconstruction using *Haar* wavelet transform. From left to right, the reconstruction using the n most representative coefficients with $n = \{20, 100, 400, 16K\}$.

pixel data. This compression characteristic brings the first advantage of using the compressed-domain index, once that we can build very small indexes with this approach.

Theoretically, the majority of the most significant coefficients use to be the same in similar images, what is not true when the images are different. Figure 3.5 presents three images on top, and its respective most significant coefficients map on the wavelet domain, at bottom. All three images are dissimilar, but the second and third ones are very similar to one each other, thus, this similarity is reflected on their respective most significant coefficients. White dots on the coefficient map of Figure 3.5 represent negative coefficients, while the white, represent positive ones. As shown, almost all significant coefficients on the second and third images are the same ones, while not on the coefficient maps among the first and the other images. The compressed domain of wavelets presents two main advantages for comparing two images: first, the representation of the image on the compressed domain is much smaller; and second, the comparison of a few set of coefficients data is computationally cheaper than comparing all image pixels.

On the work presented by Jacobs et al. [1995], the wavelet coefficients are used to index an image dataset in order to support query-by-painting and query-by-example. The authors performed experiments in two datasets, one with 1,000 images and another with 20,000.

To perform a retrieval, the query image, either an example or a painting is transformed to the wavelet domain. Then, the comparison of the query image to the compressed-domain index is performed basically comparing the number of wavelet coefficients matched between the query image and the coefficients of each indexed image. The more matches of coefficients, the more similar two images are. Moreover, some weights for the coefficients are considered. The weights of the coefficients are applied into Equation 3.4 and these weights are presented in Table 3.2.

The *Haar* wavelet decomposition is simple and fast to compute. Algorithms pre-

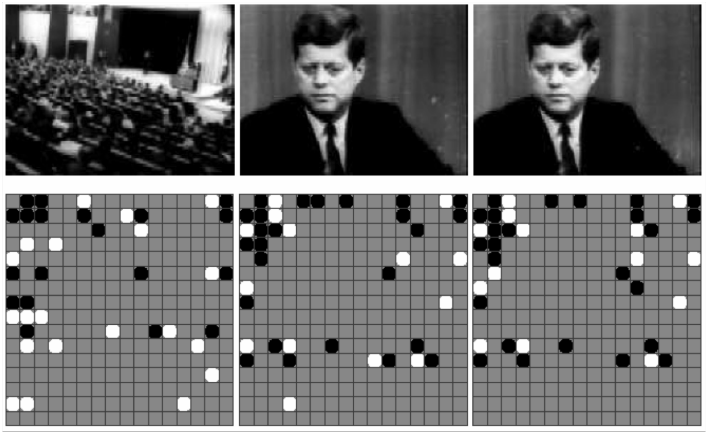

Figure 3.5. Image examples, on top, and its respective most significant quantized coefficients, at bottom. White dots on the coefficient map represent the most significant positive coefficients, while the black ones, the most negative. The three images are different, however, the second and third images are similar to one each other, and this similarity is reflected on their most significant quantized coefficients. This characteristic does not happen between the first image and the others, where most part of the quantized coefficients is different.

sented in Algorithm 1 and 2 are used for the image wavelet transform, and generate as results, the average value of gray level pixels, plus a set wavelet coefficients. The first algorithm is employed to one-dimensional decomposition of each image row/column. Algorithm 2 performs the image decomposition using the one-dimensional Algorithm 1. On the first step, Algorithm 1 decomposes all rows of the image, following, it decomposes all columns over the already decomposed rows.

Algorithm 1 Compute 1-D wavelet decomposition of a image row/column

```
1:  procedure DecomposeArray(A : array[0..h − 1])
2:      A ← A/√h
3:      while h > 1 do
4:          h ← h/2
5:          for i ← 0 to h − 1 do
6:              A′[i] ← (A[2i] + A[2i + 1])/√2
7:              A′[h + 1] ← (A[2i] − A[2i + 1])/√2
8:          end for
9:      end while
10:     A ← A′
11: end procedure
```

In Algorithm 1 (DecomposeArray) for row/column decomposition, A represents the image row or column and h (must be power of two in this algorithm) represents the number of A elements.

Algorithm 2 Compute 2-D image wavelet decomposition

```
1: procedure DecomposeImage(T : array[0..r − 1, 0..r − 1])
2:     for row ← to r do
3:         DecomposeArray(T[row, 0..r − 1])
4:     end for
5:     for col ← to r do
6:         DecomposeArray(T[0..r − 1, col])
7:     end for
8: end procedure
```

In Algorithm 2 (DecomposeImage), T represents the image matrix and r the number of rows and columns of T. In practice, DecomposeImage is better implemented by decomposing each row, transposing the matrix T, decomposing each row again, and finally, transposing back the matrix T.

To build the dataset index, Jacobs et al. [1995] used the *Haar* Wavelet, YIQ color space, non-standard wavelet decomposition [Beylkin et al., 1991; Stollnitz et al., 1995a] and image resolution of 128×128 pixels. Each channel of the YIQ color space is independently processed with Algorithms 1 and 2 aiming to obtain the main wavelet coefficients and index the image dataset.

Using the image resolution of 128×128, there are $128^2 = 16,384$ different wavelet coefficients for each color channel. Rather than using all coefficients to compare images, the authors kept just the coefficients with largest magnitude. This is the key for reducing the index size, and according to the authors, truncating the coefficients improved the discriminatory power of the metric. Also, the authors suggested using the 60 largest magnitude coefficients of each color channel for query-by-painting and 40 for query-by-example.

In the FMIQ approach, a quantization process is applied to the selected largest-magnitude coefficients. The quantization turns −1 for negative, and +1 for positive coefficients. Like truncation, the quantization of each wavelet coefficient speeds up the querying processes, reduces the storage, and according to the authors, improves the discriminatory power of the metric, *i.e.*, the use of quantized value is more discriminatory than its real value, even retaining little or no data about the precise magnitudes of major features of the images. Section 3.3.3 describes the distance measure for image retrieval and how the quantized coefficients are employed in the metric.

3.3.3 Query Distance Measure

Jacobs et al. [1995] compared two images, either a painting or an example query image

$\mathcal{Q}$, to the target image $\mathcal{T}$, using a distance measure based on two main parts. First part corresponds to the average gray level difference between the images. The second corresponds to the number of wavelet coefficient matches with same spatial position (x, y) and quantized sign ($s = \{+1\ or\ -1\}$) between the compared images. The more matches between the two images $\mathcal{Q}$ and $\mathcal{T}$, the less distant, or more similar, they are.

Formally, for the distance measure between $\mathcal{Q}$ and $\mathcal{T}$, let $\widetilde{\mathcal{Q}}[x, y]$ and $\widetilde{\mathcal{T}}[x, y]$ represent the $[x, y] - th$ truncated quantized wavelet coefficient, with sign $s = \{+1$ or $-1\}$, respectively obtained from $\mathcal{Q}$ and $\mathcal{T}$. Also, let $\mathcal{Q}[0, 0]$ and $\mathcal{T}[0, 0]$ represent the average intensity of the color channel of the images $\mathcal{Q}$ and $\mathcal{T}$. For convenience, $\widetilde{\mathcal{Q}}[0, 0]$ and $\widetilde{\mathcal{T}}[0, 0]$ which do not correspond to any wavelet coefficient, are defined as 0.

Furthermore, let us consider the equality operator, or matching *hit* function:

$$Hit_{\mathcal{Q}}(c) = \begin{cases} 1 & \exists_c \in \mathcal{A}_{\mathcal{Q}}(\widetilde{\mathcal{Q}}[x, y] = \widetilde{\mathcal{T}}[x, y], \\ 0 & \text{otherwise} \end{cases}$$

that evaluates to 1 on each match of wavelet coefficient between $\mathcal{Q}$ and $\mathcal{T}$. Where $\mathcal{A}_{\mathcal{Q}}$ represents the set of quantized largest-magnitude coefficients of $\mathcal{Q}$.

The formulation for the distance measure $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$, between $\mathcal{Q}$ and $\mathcal{T}$, at each color channel (YIQ), is presented in Equation 3.4.

$$\mathcal{W}_{\mathcal{Q},\mathcal{T}} = w_0 \left|\mathcal{Q}[0, 0] - \mathcal{T}[0, 0]\right| - \sum_{i=1}^{N_{\mathcal{Q}}} w_{bin(x,y)} Hit_{\mathcal{Q}}(c_i) \tag{3.4}$$

In Equation 3.4, $N_{\mathcal{Q}}$ is the number of coefficients of $\mathcal{Q}$; and w_b is the weight factor based on Table 3.2. The $bin(x, y)$ function, presented in Equation 3.5, is used to address the weight line b of Table 3.2, according to the coefficient position (x, y). Where, $min(x, y)$ is a function that returns the minimum value between x and y, and $max(x, y)$ returns the maximum value.

$$bin(x, y) \coloneqq min\{max\{x, y\}, 5\} \tag{3.5}$$

While the lines of Table 3.2 address the weights based on the coefficient position, columns address the weight according to the kind of query, by painting or by example, as well as, the color channel (YIQ).

The practical application of Fast Multiresolution Image Querying is shown in Figure 3.6. In this figure, we have the painting image query on left and the best classified results on a grid on the right.

Table 3.2. Weights for the wavelet coefficients.

	Painting			Example		
b	$w^Y[b]$	$w^I[b]$	$w^Q[b]$	$w^Y[b]$	$w^I[b]$	$w^Q[b]$
0	4.04	15.14	22.62	5.00	19.21	34.37
1	0.78	0.92	0.40	0.83	1.26	0.36
2	0.46	0.53	0.63	1.01	0.44	0.45
3	0.42	0.26	0.25	0.52	0.53	0.14
4	0.41	0.14	0.15	0.47	0.28	0.18
5	0.32	0.07	0.38	0.30	0.14	0.27

Figure 3.6. Query-by-painting in the Fast Multiresolution Image Querying application.

3.4 Conclusion

This chapter presented a wide review on SBIR, from the first known works to the most recent publications. This review also described two main works that brought impact in the development of the present approach. First approach, Mind-Finder, inspired the comparison and representation of image contours in quantized orientations. The second, Fast Multiresolution Image Querying, inspired the use of the wavelet-domain to represent the dataset index in the compressed-domain. The algorithms presented for the wavelet transform are also used in this dissertation approach. From both works presented in this chapter, we also based the similarity measure of our proposal. Concerning to inverted lists, the present dissertation also takes the advantages of such strategy to speed up the queries, as described in Section 4.1.4.

Next chapter describes our proposal, the methods and the processing steps in details.

Chapter 4

Sketch-Finder Approach

In this chapter, we introduce our approach for SBIR, named Sketch-Finder. First, we resume the main features of the approach, and then, we present the technical details for indexing the dataset and querying an image. On the technical description, we present the feature extraction steps, our similarity measure proposal using the features, and the index structure.

Sketch-Finder is an approach for black and white sketches. Those sketches consist of a stroke set that describes important contours of the images that the user is looking for. Here, we consider that, the more a query and a target image have strokes/contours at the same or near position and orientation, the more similar they are.

This proposal is presented in two versions: Sketch-Finder 1.0 and 2.0. Although the newer version presents better effectiveness, the presentation of the Sketch-Finder 1.0 is important to show how Sketch-Finder 2.0 achieved its properties. Further, Sketch-Finder 1.0 is faster than Sketch-Finder 2.0, making the first approach a good choice when efficiency is imperative, like in the prototype for mobile devices shown in Chapter 7.

Sketch-Finder 1.0 and 2.0 have two main differences: the first version uses only the compressed domain of wavelets to encode the index of the dataset, also, the number of features to represent the image contours is fixed, while on the second version, the number of wavelet coefficients is variable. Further, besides the compressed domain, Sketch-Finder 2.0 uses the pixel domain to verify the contour consistency between the sketch strokes and the image contours, what improves the effectiveness of the approach. Our proposed approach uses an index stored on disc with efficient data structures, granting the growth of the image dataset without memory limitations dependence, thus, allowing big data sketch-based image retrieval.

According to the users objective (Section 2.1.4), he/she can retrieve images by

sketch in two main scenarios. In the first one, he/she may wish to retrieve the desired object, no matter its scale and/or position. Usually, approaches using image descriptors and "Bag-of-Visual-Words" (BoVW) fall in this category, like the works of Eitz et al. [2011]; Hu and Collomosse [2013]. By the other hand, the user may want to retrieve not just the object, but he/she also has in mind its similar position, rotation and object scale. Our approach, and others like [Cao et al., 2011; Del Bimbo and Pala, 1997; Tseng et al., 2012; Jacobs et al., 1995; Sun et al., 2013] lies inside this objective. These approaches do not use the BoVW, but a descriptor that preserves, in some way, the spatial information. Therefore, for the proposed approach in this thesis work, we consider the similarity between a sketch and some image using two main criteria: (*i*) *shape sensitive*, which means that the contour shape of the target image objects must be as close as possible to the sketch, and (*ii*) *position sensitive*, which means that image contours should be as close as possible to the sketch strokes in terms of position and scale.

The present approach aims to improve the retrieval efficiency comparing to Mind-Finder [Cao et al., 2011], while preserving effectiveness. High memory and computational cost for holding and processing a big amount of data in the pixel domain is one problem of Mind-Finder, *i.e.*, keeping the index and processing a huge amount of edgels in big datasets. Not only us noticed this problem on Mind-Finder, but other works like those described in [Tseng et al., 2012; Sun et al., 2013], also did. These works were actually developed in parallel to ours, without knowing one each other. Further, the present thesis and the works [Tseng et al., 2012; Sun et al., 2013] have in common Mind-Finder as base for comparison of effectiveness and efficiency. These works also share the same user objective and basic root methods for SBIR, *e.g.*, the orientation of the image contours. Although those works followed other ways to solve the problem, they reached a similar objective about compressing index data, like we did in the present thesis. While in [Tseng et al., 2012; Sun et al., 2013], the authors used a compact hash of bits, in this thesis we used the compressed-domain index based on the wavelet decomposition. The pixel domain is also used on an improved version of the present approach, in order to verify the spatial consistency of the edgels and to increase effectiveness. As shown in Section 3.2, an edgel is an edge pixel with its spatial position (x, y) plus the orientation (θ) of the stroke/contour that the pixel remains. This segmentation in quantized orientations is important to better describe the image contours or the sketch strokes without merging the same area around the contour in different orientations. Next section describes the technical details of Sketch-Finder 1.0 and some properties for Sketch-Finder 2.0. Section 4.2 presents the main characteristics introduced on the approach for Sketch-Finder 2.0.

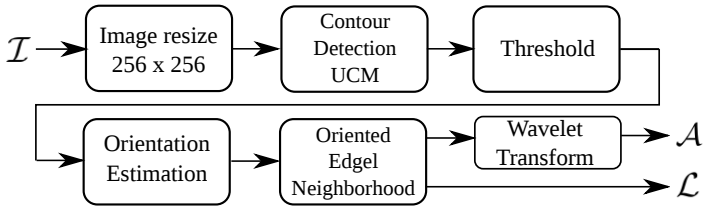

Figure 4.1. Indexing workflow of Sketch-Finder 1.0 and 2.0 – in this sequence: image input $\mathcal{I}$ resize; contour detection using the Ultrametric Contour Map; contours thresholding; edge orientation estimation; oriented edgel neighborhood estimation inside a radius r, given in pixels (set of edgels $\mathcal{L}$ only for Sketch-Finder 2.0); standard *Haar* wavelet decomposition; set of wedgels $\mathcal{A}$, or, wavelet most significant coefficients.

4.1 Sketch-Finder 1.0

Sketch-Finder 1.0 is an approach for SBIR based in two main works: Mind-Finder [Cao et al., 2011] presented in Section 3.2 and Fast Multiresolution Image Querying presented in Section 3.3. Even as the approaches [Tseng et al., 2012; Sun et al., 2013], Sketch-Finder 1.0 inherits the idea of representing and comparing sketch strokes and image contours in six quantized orientations. From the approach FMIQ, Sketch-Finder inherits the idea of using the compressed-domain index, where image contours and sketch strokes are represented by wavelet coefficients. Roughly, the similarity measure of Sketch-Finder compares the number of largest magnitude coefficients at the same position, signal and orientation map. Details of preprocessing, indexing and querying images are presented in the following.

4.1.1 Feature Extraction

On a SBIR approach, it is necessary to index the image dataset. First, we need to preprocess the images and extract the features that represent the contour in the index. On Sketch-Finder, the feature extraction process consists on six main steps: image resize; contour detection using the Ultrametric Contour Map; threshold of the contours; orientation; oriented edgel neighborhood estimation inside a radius r, given in pixels; and wavelet analysis. This process is illustrated in Figure 4.1. At the end of the process, we get the wavelet coefficients obtained from different oriented contours.

The coefficients resultant of the image processing are the basic visual features of the Sketch-Finder, and we named this feature as *wedgel*, a short name for wavelet coefficient obtained from edgel information. This *visual word* is based on the *Haar* wavelet transform using Algorithms 1 and 2 (shown in Section 3.3.2) plus its oriented

edgel map source. In our approach, the wedgel is represented as a quadruple (x, y, s, θ), where (x, y) represent the wavelet coefficient spatial position, s the coefficient quantized sign (+1 or -1) and θ the edge map orientation used before the wavelet transform.

Each step of the image processing, represented in Figure 4.1, to obtain the set of wedgels is described with details on following.

Image resize. Aiming to have all images with the same size, important for matching the spatial position (x, y) of the *wedgels*, all images are resized to the same pattern, 256×256 pixels. Although the image aspect ratio is lost, all images, likewise the sketch, use the same aspect and the distortion in position is naturally compensated by the spatial approximation of the visual features obtained on the next steps. Some examples of resized images are shown in Figure 4.2 (a).

Contour detection. Because our approach uses black and white line-based sketches, we need to detect the contours of the dataset images. As such, a good algorithm of contour detection is fundamental for the success of this approach. To obtain the image contours, we use the hierarchical Ultrametric Contour Map (UCM) approach, with the default parameters, described in [Arbelaez et al., 2011]. As far as we know, this high level contour detection is state-of-the-art in literature. Some examples of the UCM are shown in Figure 4.2 (b), respectively obtained from the images shown in line (a) of the same figure. Also, in the figure, we present the contours in black with white background for better printing of the present text, however, the real contours are white lines in a black background. This information is important because the parameter value used in the next step, the threshold, depends on how the contour is represented.

Threshold. To obtain the most important object contours, the result given by the UCM algorithm is then thresholded. The threshold choice and other parameters were obtained on an optimization process with a genetic algorithm, as described in Section 6.2.2.

Orientation estimation. The natural contours of each image $\mathcal{I}$ and the query sketch $\mathcal{Q}$ are quantified in N_Θ oriented Θ channels, where Θ is a set of quantified orientations: $\Theta = \{\theta_1, \theta_2, ..., \theta_n\}$. In this work, the edge orientation is estimated and quantified in six intervals, *i.e.*, $N_\Theta = 6$: $-15° \sim 15°, 15° \sim 45°, ..., 135° \sim 165°$. Each edge pixel (x, y) plus its orientation information θ, named edgel, is defined as $p_i = (x, y, \theta)$ [Cao et al., 2011]. Figure 4.3 (b) represents the edge orientation estimation obtained from Figure 4.3 (a). In this figure, we show just three

orientations to simplify the idea and illustration, although we actually use six orientations.

Oriented edgel neighborhood. For each orientation θ, the pixel neighborhood of each edge element in a radius r, given in pixels, is integrated to the set of edgels, making this neighborhood part of the set of edgels $\mathcal{L} = \{p_1, p_2, ..., p_n\}$. In this dissertation, the neighborhood of edgels is simply named "*edgel neighborhood map*" $\mathcal{G}_\theta^\mathcal{I}$, of an Image $\mathcal{I}$ and orientation θ. Figure 4.3 (c) presents an illustration of three $(\mathcal{G}_{\theta_1}^\mathcal{I}, \mathcal{G}_{\theta_2}^\mathcal{I}, \mathcal{G}_{\theta_3}^\mathcal{I})$ edgel neighborhood, with radius $r = 2$ pixels, obtained from Figure 4.3 (b). More details on defining r are given in Section 6.2.2.

Wavelet transform. For each edgel neighborhood map $\mathcal{G}_\theta^\mathcal{I}$, we apply the standard *Haar* wavelet transform [Chui, 1992; Stollnitz et al., 1995a], following Algorithms 1 and 2 presented in Section 3.3.2. The n largest magnitude coefficients are used to encode each transformed $\mathcal{G}_\theta^\mathcal{I}$, composing the set of *wedgels* $\mathcal{A} = \{w_1, w_2, ..., w_n\}$, where a wedgel is a quadruple $w_i = (x, y, s, \theta)$, with its coefficient at spatial position represented by (x, y), its quantized sign s, and its edgel neighborhood map orientation represented by θ. In Sketch-Finder 2.0, instead of using a fixed number of coefficients, we select the largest magnitude coefficients greater than a given threshold ω, Section 4.2 explains why this change is taken. Figure 4.3 (d) represents the positive wedgels in white dots, and the negative ones in black. Those coefficients were obtained from the *Haar* wavelet transform of $\mathcal{G}_\theta^\mathcal{I}$ represented in Figure 4.3 (c). Selected coefficients or wedgels are quantized to +1 if the coefficient is positive, or -1 otherwise; in our approach, the real value of the selected coefficients is not important. The number of words of the dictionary, or different wedgels, can be calculated by using $N_R \times N_C \times N_S \times N_\Theta$; where N_R represents the number of image rows; N_C the number of columns; N_S the number of quantized signs for the wavelet coefficient, always two; and N_Θ the number of quantized orientations. In our proposal, the index dictionary has $256 \times 256 \times 2 \times 6 = 786432$ possible wedgels.

Some methods and parameters of this work are based either on literature, either on the best balance between precision of the desired results and efficiency based on experiments. The image size of 256×256 pixels is an approximation of the size used in the work Cao et al. [2011] which is 200×200. This resolution presents compatibility with the fast wavelet decomposition algorithm presented in Jacobs et al. [1995] which must have the lines and columns equals 2^k, further this resolution presents a good relation in terms of object shape representation and acceptable computational time on the contour

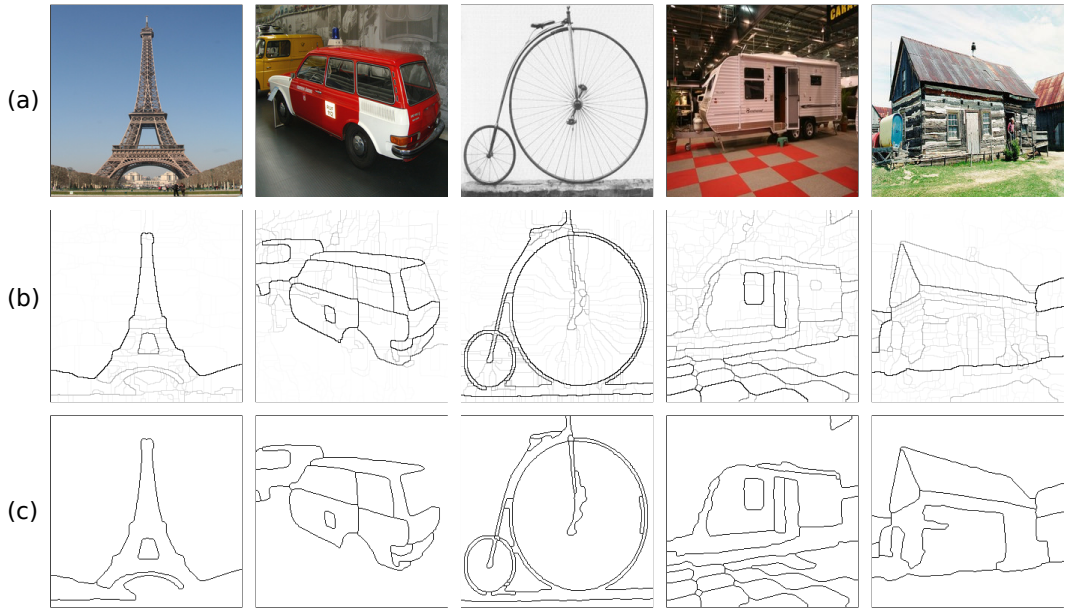

Figure 4.2. Contour detection examples from natural images using the Ultrametric Contour Map (UCM) algorithm and their respective thresholded images with $threshold = 0.16$ in $[0, ..., 1]$. We have in (a) natural images; in (b) hierarchical contours; and in (c) the thresholded contours.

detection of the algorithm Ultrametric Contour Map (UCM) Arbelaez et al. [2011]. Smaller resolution might bring lost of important information while bigger resolution must be prohibitive in terms of computational costs. The choice of the UCM for contour detection comes from literature review, that says, as far as we know, that this contour detection is the state of the art on its domain, further, this algorithm is available for research proposals. The contours threshold and the radius size of the neighborhood are experimentally obtained as described in Section 6.2.2 while the number of orientation for the contours comes from the Mind-Finder approach. The choice of *Haar* wavelet comes from the work Jacobs et al. [1995], this wavelet algorithm is fast and simple to implement.

Figure 4.3 presents the natural image contours or the sketch strokes processing until the wavelet domain representation.

Algorithms 1 and 2 presented in Section 3.3.2, were used in our approach to obtain Haar wavelet coefficients. Further details on the wavelet transform used in this thesis can be found in [Antonini et al., 1992; Daubechies, 1992; Mallat, 2008; Jacobs et al., 1995; Stollnitz et al., 1995a,b].

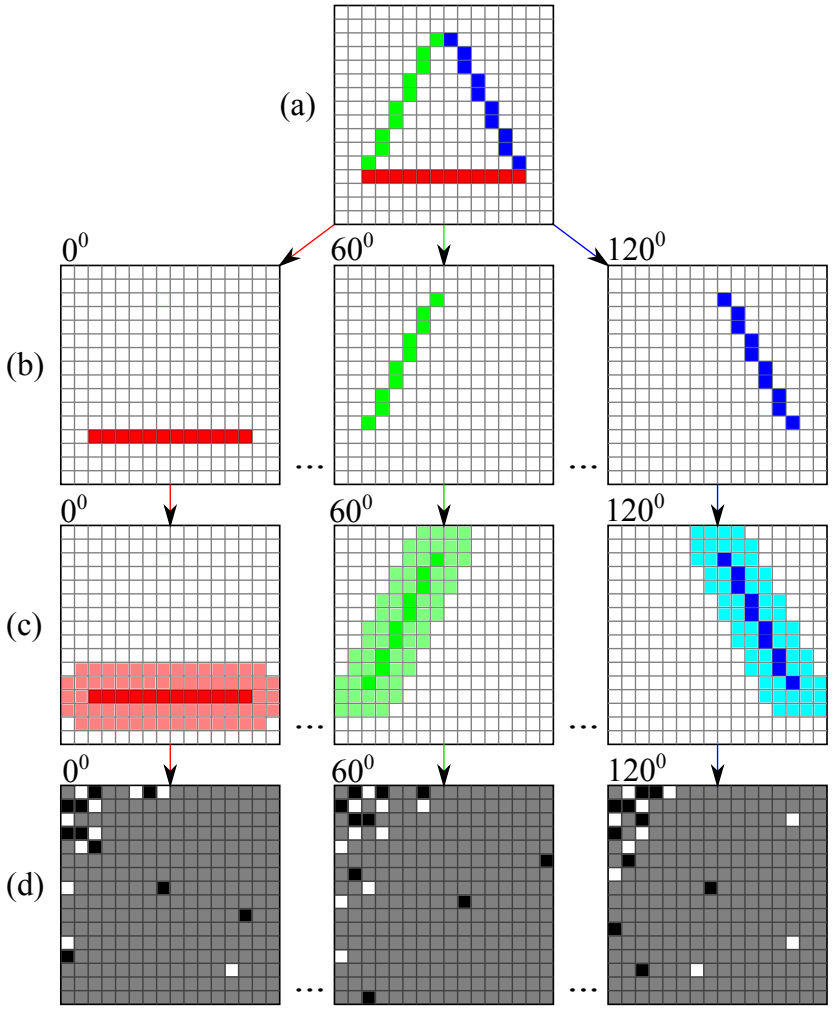

Figure 4.3. Visual features extraction: Preprocessing: (a) image contours map; (b) estimation of edges orientation and segmentation of their contour maps; (c) neighborhood edgels estimation; (d) wavelet domain representation of the ten most positive coefficients (white dots) and the ten most negative coefficients (black dots).

4.1.2 Similarity Measure

The use of the compressed-domain of wavelet is an efficient strategy for SBIR, as shown by the experiments presented in Chapter 6. The similarity measure $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$ between the query sketch $\mathcal{Q}$ and some target image $\mathcal{T}$ of the dataset is based on computing the number of similar wedgels matched between $\mathcal{Q}$ and $\mathcal{T}$, times some weight. On the following, we present the similarity measure equation; the appropriate weight for the coefficients were chosen based on the experiments presented in Chapter 6.

Wavelet Comparison

Consider the problem of computing the similarity, on the wavelet domain, between a query sketch $\mathcal{A}_\mathcal{Q}$ and a potential target image with set of wedgels $\mathcal{A}_\mathcal{T}$. Let $\widetilde{\mathcal{A}_\mathcal{Q}}[x, y, s, \theta, \mathcal{G}_r]$ and $\widetilde{\mathcal{A}_\mathcal{T}}[x, y, s, \theta, \mathcal{G}_r]$ represent the $[x, y] - th$ truncated quantized wavelet coefficient, respectively of $\mathcal{Q}$ and $\mathcal{T}$, with sign $s = (+1 \text{ or } -1)$, orientation θ and neighborhood edgel map $\mathcal{G}_r$, with radius r given in pixels. Also let N_Θ represent the number of orientations, and $N_\mathcal{G}$ the number of different $\mathcal{G}_r$ used in the index.

Furthermore, let us consider the equality operator, or *hit* (Equation 4.1) wedgel function:

$$Hit_\mathcal{Q}(w) = \begin{cases} 1 & \exists_w \in \mathcal{A}_\mathcal{Q}(\widetilde{\mathcal{A}_\mathcal{Q}}[x, y, s, \theta, \mathcal{G}_r] = \widetilde{\mathcal{A}_\mathcal{T}}[x, y, s, \theta, \mathcal{G}_r]), \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

that evaluates to 1 on each match of wedgel between $\mathcal{Q}$ and $\mathcal{T}$. For the wavelet coefficients comparison $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$, the more similar $\mathcal{Q}$ and $\mathcal{T}$ are, the more matches of wavelet coefficients at the same spatial position, sign and orientation are computed. The comparison is taken in several parts, one for each combination of the set $\mathcal{K} = \{\mathcal{G}_r, \theta, s\}$. Let $| \mathcal{A}_{\mathcal{Q}(\mathcal{G}_r,\theta,s)} |$ be the number of coefficients of the query sketch for the set $\{\mathcal{G}_r, \theta, s\}$. The equation for the sum of match weighted wedgels $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$ between $\mathcal{Q}$ and $\mathcal{T}$ is presented in Equation 4.2.

$$\mathcal{W}_{\mathcal{Q},\mathcal{T}} = \sum_{\mathcal{G}_r=1}^{N_\mathcal{G}} \sum_{\theta=1}^{N_\Theta} \sum_{s=1}^{2} \sum_{i=1}^{|\mathcal{A}_{\mathcal{Q}(\mathcal{G}_r,\theta,s)}|} \left(\alpha \cdot bin(x, y) + (1 - \alpha) \cdot \mathcal{X}\right) \cdot Hit_\mathcal{Q}(w_i) \tag{4.2}$$

Where $\alpha \in [0 - 1]$ and is used to perform a linear interpolation between two kind of weights. First weight, determined by $bin(x, y)$ function (shown in Equation 3.5) is based on the hypothesis that the weight for the wavelet coefficients can be based on the coefficient position (x, y) as in [Jacobs et al., 1995]. For *bin* function, we considered the neighborhood edgel map as a luminance channel Y from YIQ color model [Gonzalez and Woods, 2006]. Thus, we used the weights for *bin* function given in Table 3.2, for painting image and color channel Y, *i.e.*, column $w^Y[b]$.

Experiments considering only *bin* function with $\alpha = 1$ did not presented good results in some sketch examples. These examples were images where the distribution of edgels within the six oriented maps were not uniform, *i.e.*, when some orientation was much more dense in strokes rather than others. For example, the Pantheon sketch (e.g. Figure 5.14 (h)) presents much more vertical strokes than horizontal and the other intermediate orientations. Using only *bin* function for The Pantheon sketch, brings, among true positive images, lots of pyramids due to the pyramid above the vertical

strokes on the Pantheon. This occurs because each oriented map represented with the same number of wavelet coefficients plays the same importance weight. However, to bring more true positive pantheons, it is necessary to give more importance to dense orientations. This drives us to a second hypothesis for coefficient weights.

On this second hypothesis, the weight $\mathcal{X}$ (Equation 4.3) is determined by the estimation of the number of edges in Q, given by the function t.

$$\mathcal{X} = \frac{t(\theta, \mathcal{G}_r)}{r} \quad _{w \epsilon \mathcal{A}_Q} \tag{4.3}$$

Function $\mathcal{X}$ models orientation maps with higher importance related to large number of edges, *i.e.*, the more edgels at orientation θ, the higher is the weight for its wavelet coefficients. The radius size of $\mathcal{G}_r$, represented by r in pixels, is used in Equation 4.3 to balance the average gray level of the oriented maps according to its neighborhood edgel map, otherwise, bigger radius would bring higher weights. Thus, the division by r aims to provide a good balance of the coefficient weight when combining more than one neighborhood edgel map. Otherwise, larger neighborhood edgel maps, without the division by r, should give higher relevance in relation to edgel maps with small radius, which is not desired.

A simple way to measure the amount of edges in order to know the importance of the orientation, thus the weights of the wedgels, is to use the average value of gray level of $\mathcal{G}_\theta^Q$. The higher the average value is, the more edges are present. Also, this strategy approximates our approach to the idea of the similarity measure presented in Cao et al. [2011]. The average value of gray level of $\mathcal{G}_\theta^Q$ at r is given within the interval $[0.1, ..., 1]$, so as the minimum value of the weight is 0.1, empty $\mathcal{G}_\theta^Q$ or with just a few number of edges are also considered, but with small similarity contribution. Given the average value of gray level within $[0, ..., 1]$, the weight for t is given by Equation 4.4:

$$t = AVG(\mathcal{G}_\theta^Q) - \frac{AVG(\mathcal{G}_\theta^Q)}{10} + 0.1 \tag{4.4}$$

where $AVG(\mathcal{G}_\theta^Q)$ represents the average value of gray level of $\mathcal{G}_\theta^Q$, within the interval $[0, ..., 1]$.

As shown in the experiments of Section 6.1.1, using more than one version of neighborhood edgel map on the wavelet transform improves the precision of the approach. Thus, using more than one radius r neighborhood edgel map version, produces different versions of wedgels. In fact, it is one set of wedegel for each configuration of radius r.

We performed several experiments to evaluate the importance of the coefficient

weight according to our first and second hypothesis of weights, using *bin* function and edgel density $\mathcal{X}$, thus, varying α between 0 and 1 on Equation 4.2. Results presented in Section 6.1.1 prove that second hypothesis, where the coefficient weight is based on the edgel density of each oriented map $\mathcal{X}$, presents better precision.

4.1.3 Query Process

The query process has some common steps to the indexing workflow described in Section 4.1.1: image resize; orientation estimation; neighborhood edgel map; and wavelet transform. Contour detection is not necessary on the query time because the sketch is already a contour entity, and the threshold is not applied because we consider that the input is already binary. Therefore, we describe on the following just the processes not in common to those explained in Section 4.1.1 for indexing the dataset. Figure 4.4 presents the query workflow of the proposed approach.

Similarity measure. In this step, the set of wedgels $\mathcal{A}_\mathcal{Q}$ obtained from the sketch is used to load in main memory the correspondent inverted file list of each wedgel $w_i \in \mathcal{A}_\mathcal{Q}$. Following, we measure the similarity with Equation 4.2. In Sketch-Finder 2.0, the set of edgels $\mathcal{L}_\mathcal{Q}$ is also retrieved in inverted files for the OCM precision refinement. More details about inverted list concepts and our practical implementation can be found respectively on Sections 3.2.2 and 4.1.4.

Sorting the similarity. Once we have the similarity of the sketch to the images of the dataset, obtained on the last step, we sort the similarity using the quick sort algorithm [Cormen et al., 2001] and present the z most relevant images for the user. In our prototype, we present the results in a (HTML) page format. A text file is also created with all results for precision×recall evaluation of effectiveness.

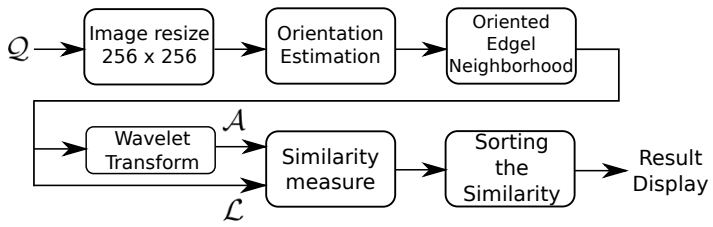

Figure 4.4. Query workflow – in this sequence: the sketch input $\mathcal{Q}$; sketch resize; edges orientation estimation; edges orientation estimation and set of edgels $\mathcal{L}$ (only for Sketch-Finder 2.0); standard *Haar* wavelet decomposition and set of wedgels $\mathcal{A}$; similarity measure of the sketch strokes; sorting the similarity results; and display of the classified results.

4.1.4 Index Structure

The proposed approach of this thesis presents several advantages. Its compact index was designed to support efficient and effective query. Efficiency is mainly achieved because the compressed-domain has only a few number of wavelet coefficients to process, rather than on uncompressed index.

In lossy image compression formats like JPEG [Ansari and Memon, 2000] and JPEG2000 [Christopoulos et al., 2000], the user may choose between compression rate and image size, accepting lower image quality to gain memory space. We can use this very same idea in our index size. Theoretically, the more coefficients, the bigger is the index size, gaining better effectiveness of retrieval by paying less efficiency, and vice-versa. Our experiments presented in Section 6.2.1, respectively for Sketch-Finder 1.0 and 2.0 demonstrated in both that, the impact of the index size on the effectiveness is small, among several parameters. Thus, in our approach, it is possible to control the index size, thereafter, choosing the most desirable characteristic between effectiveness and efficiency. Furthermore, it is possible to have different index versions on disk, and load the most desirable one according to the machine memory and CPU load, if other processes are concurring. Our compact index also allows big data image indexing for SBIR.

Finally, we can mention the small variation on the query speed of our approach as shows the standard deviation in Table 6.7, Section 6.2. This statistical measure presents a small variation of CPU usage on the query time, when compared to Mind-Finder. The reason for less variation time in the query speed comes from fixed number of *wedgels* to compose the contour signature, on Sketch-Finder 1.0. Thus, Sketch-Finder always processes the same number of *wedgels* on the similarity measure, an advantage that using our approach is possible to have a better estimation of the time of each query. On the following topic, we present the structure of the proposed index.

4.1.5 Wedgel Index

Aiming to improve performance, the "matching" of *wedgels* is performed by retrieving each inverted file list of image (IDs), associated to each *wedgel* (x, y, s, θ) of the sketch. This strategy acts like a shortcut to go directly on the important information to process, without having to read all index. Each image ID in the inverted lists represents one *hit* or match in the similarity measure. After processing the corresponding lists of each sketch wedgel, the similarity of the images of the indexed dataset, is then, sorted using quick sort algorithm.

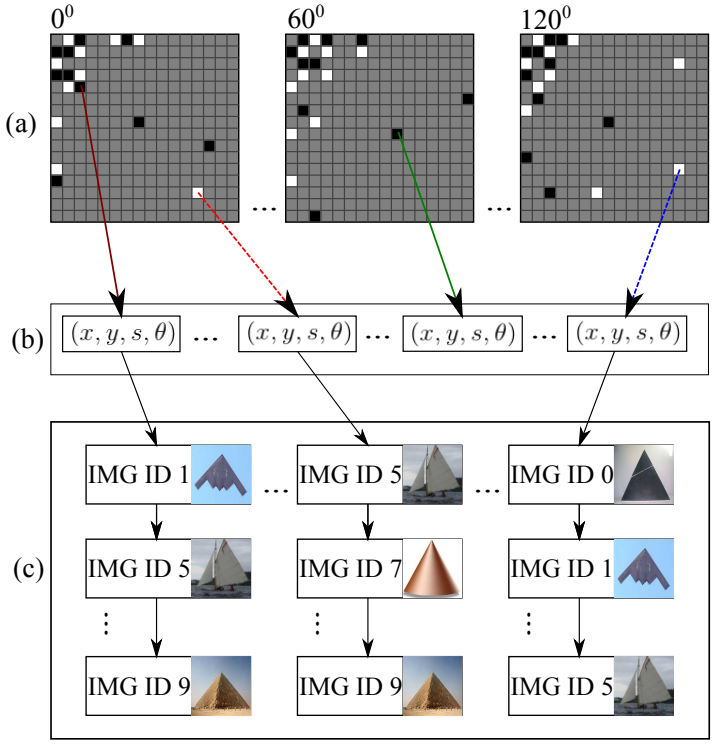

Figure 4.5. Indexing of wedgels using inverted lists: (a) wavelet domain representation of coefficients (wedgels) with positive coefficients represented in white and negative in black; (b) wedgels dictionary; and (c) inverted list of image IDs for each wedgel.

Figure 4.5 presents the index structure for the compressed-domain of wavelet coefficients. The wavelet domain with its most significant coefficients, already quantized, are shown in Figure 4.5 (a). Negative coefficients are represented by black dots, while the positive ones are represented on white. To simplify the idea, here, we illustrate just three orientations, although, actually we have six. Figure 4.5 (b) presents the wedgel dictionary (x, y, s, θ), and Figure 4.5 (c), presents the list of image IDs associated with each wedgel of the dictionary.

4.2 Sketch-Finder 2.0

Sketch-Finder 2.0 presents two main differences over its antecessor, the number of wavelet coefficients is no more constant, furthermore the similarity of the compressed-domain index is refined in the pixel domain, improving effectiveness.

First version of Sketch-Finder presented two ways for weighting for the wavelet coefficients. The fist one is based on the coefficient position and the second is based on the density of edgels on each orientation. As shown in the experiment varying

the weight function presented in Section 6.1.1, the use of edgel density information impacts in retrieval precision better than the wavelet coefficient position. Following this characteristic, instead of considering weights based on edgels density, that is directly related to the number of coefficients, Sketch-Finder 2.0 simply computes the number of coefficients. This change makes the approach more simple and effective as presented in Section 6.2.3.

The second important change on the new approach of Sketch-Finder is the use of the pixel domain besides the wavelet-domain index, aiming to improve even more the effectiveness of the approach. Although the use of only wavelet-domain presents an equivalent effectiveness with Mind-Finder using natural image contours as query input, the use of real sketches presents an effectiveness inferior to Mind-Finder. Further, it was verified in our experiments, that some contours are not very well matched in the compressed-domain, mainly, due to the loss of information. In order to better match the spatial pixel consistency, between the sketch and the image contours of the dataset, we use the Oriented Chamfer Matching as in Mind-Finder. Nevertheless, we do not perform two hand similarity verification as Mind-Finder (see Equation 3.3), we just need to apply the verification from the query sketch to the target image, which is the cheapest computational part of Mind-Finder similarity.

These two improvements, mentioned before, for Sketch-Finder 2.0, are modeled and presented in the following.

4.2.1 Similarity Measure

In Sketch-Finder 2.0, the similarity measure $Sim_{\mathcal{Q},\mathcal{T}}$ between the query sketch $\mathcal{Q}$ and some target image $\mathcal{T}$ of the dataset is given by the wavelet coefficients similarity $1/(\mathcal{W}_{\mathcal{Q},\mathcal{T}} + 1)$ times the edgels similarity $\mathcal{P}_{\mathcal{T}\rightarrow\mathcal{Q}}$ performed by the Oriented Chamfer Matching (OCM). The similarity measure $Sim_{\mathcal{Q},\mathcal{T}}$ between $\mathcal{Q}$ and $\mathcal{T}$ is presented in Equation 4.5 and the estimation of $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$ and $\mathcal{P}_{\mathcal{T}\rightarrow\mathcal{Q}}$ are presented in Equations 4.7 and 4.9, respectively.

For the wavelet similarity, we use the same *hit* function of Sketch-Finder 1.0 presented in Equation 4.1. Further, for the wavelet similarity of this version, let $\mathcal{C}_{\mathcal{Q}_{\mathcal{K}_i}}$ be the number of wedgels of the sketch image and $\mathcal{C}_{\mathcal{T}_{\mathcal{K}_i}}$ the number of wedgels of the target image for $\mathcal{K}_i$. The comparison is taken in several parts, one for each combination of the set $\mathcal{K} = \{\mathcal{G}_r, \theta, s\}$. Let $N_{\mathcal{K}}$ be the number of combination for the set $\mathcal{K}$ and $\mathcal{K}_i$ the ith combination of $\mathcal{K}$. The equation for summing of matched wedgels $\mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})}$ at each combination of $\mathcal{K}$ between $\mathcal{Q}$ and $\mathcal{T}$ is presented in Equation 4.6. The similarity between the set of wedgels from the query sketch $\mathcal{Q}_{\mathcal{K}_i}$ and the set of wedgels for some

target image $\mathcal{T}_{\mathcal{K}_i}$ is presented in Equation 4.7.

$$Sim_{\mathcal{Q},\mathcal{T}} = \frac{1}{\mathcal{W}_{\mathcal{Q},\mathcal{T}} + 1} \cdot \mathcal{P}_{\mathcal{T}\to\mathcal{Q}} \tag{4.5}$$

$$\mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})} = \sum_{q=1}^{|\mathcal{A}_{\mathcal{K}i}|} Hit_{\mathcal{Q}}(w_{\mathcal{K}_q}) \tag{4.6}$$

$$\mathcal{W}_{\mathcal{Q},\mathcal{T}} = \sum_{i=1}^{N_{\mathcal{K}}} \left|\mathcal{C}_{\mathcal{Q}_{\mathcal{K}_i}} - \mathcal{C}_{\mathcal{T}_{\mathcal{K}_i}}\right| + \left(\mathcal{C}_{\mathcal{Q}_{\mathcal{K}_i}} - \mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})}\right) + \left(\mathcal{C}_{\mathcal{T}_{\mathcal{K}_i}} - \mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})}\right) \tag{4.7}$$

Where, $|\ \mathcal{A}_{\mathcal{K}i}\ |$ is the number of wedgels for the combination of $\mathcal{K}_q$. Thus, the comparison in the compressed-domain $\mathcal{W}_{\mathcal{Q},\mathcal{T}}$ shown in Equation 4.7 considers the distance between the sketch $\mathcal{Q}$ and the target image $\mathcal{T}$ in three parts. First, the difference between the number of coefficients of $\mathcal{Q}$ and $\mathcal{T}$ $\left|\mathcal{C}_{\mathcal{Q}_{\mathcal{K}_i}} - \mathcal{C}_{\mathcal{T}_{\mathcal{K}_i}}\right|$; second, the difference between the number of sketch coefficients $\mathcal{C}_{\mathcal{Q}_{\mathcal{K}_i}}$ and the number of matched coefficients $\mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})}$, which is zero if all sketch coefficients match; and third, the difference between the number of coefficients in the target image $\mathcal{C}_{\mathcal{T}_{\mathcal{K}_i}}$ and the number of matched coefficients $\mathcal{B}_{(\mathcal{Q}_{\mathcal{K}_i},\mathcal{T}_{\mathcal{K}_i})}$, also zero if all target image coefficients match. This distance in three parts is important because similar images must have a similar number of coefficients, measured in the first part of the equation. Further, matching all coefficients of the sketch is not enough to affirm that the sketch is similar to the target image (second part of Equation 4.7) if the target image has not all coefficients also matched (third part of Equation 4.7).

4.2.2 Oriented Chamfer Matching

The proposed compressed-domain index of Sketch-Finder 1.0 presents a very compact index and fast query. However, the effectiveness is just equivalent for natural contours input, but not superior to the one presented in Cao et al. [2011] for real sketches. Thus, by only analyzing the results of the wavelet comparison, we observed that this method is robust to some spatial variation between the position of the sketch strokes and the image contours. Nevertheless, the consistency on the quantity of strokes at similar position is not very well matched in the wavelet domain due to the loss of information. To overcome this problem, a verification in terms of how much the number of edgels are similar between the query sketch and the image contour, at similar spatial position, becomes necessary. One strategy that can be applied to solve this problem is the Chamfer Matching [Borgefors, 1988] or, more specifically, the Oriented Chamfer Matching [Cao et al., 2011; Lee and Grauman, 2009b; Liu et al., 2010].

Although the use of OCM decrease the efficiency of the method, this second comparison does not present high computational cost for two reasons: first, it does not process two similarity measures of OCM as in Cao et al. [2011], *i.e.*, the similarity between the sketch $\mathcal{Q}$ and the target image $\mathcal{T}$ ($\mathcal{Q} \rightarrow \mathcal{T}$) and the similarity ($\mathcal{T} \rightarrow \mathcal{Q}$). The present approach just performs the OCM of ($\mathcal{T} \rightarrow \mathcal{Q}$). Second, all oriented edgel maps of the dataset are indexed already with their neighbors in a radius r given in pixels, saving computational cost to compute the edgels inside a radius r during the query time, as defined in *Hit* function $\mathcal{P}_{\mathcal{T}\rightarrow\mathcal{Q}}$, presented in Equation 4.9. Another possibility, is to process the OCM only for the z best ranked images on the wavelet comparison, saving computational cost and index size.

Figure 4.6 illustrates the pixel consistency idea between the query sketch $\mathcal{Q}$ and some target image $\mathcal{T}$. Although in Figure 4.6 the pixel consistency is presented without considering the edge orientation, for simplification of the scheme, the comparison takes the orientation into account.

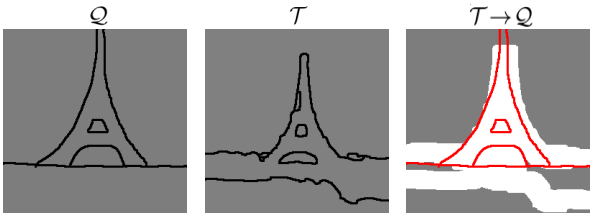

Figure 4.6. Pixel consistency comparison between the sketch and some target image. In this figure, form left to right; the sketch strokes $\mathcal{Q}$; the target image contours $\mathcal{T}$ and the representation of sketch stroke pixels inside a neighborhood radius r (given in pixels) of $\mathcal{T}$ ($\mathcal{T} \rightarrow \mathcal{Q}$).

Formally, consider the set of edgels $\mathcal{L}_\mathcal{Q}$, representing the oriented sketch contours, whose position of the edgels $p \in \mathcal{L}_\mathcal{Q}$ is denoted by $X_p = (x_p, y_p)$, and its gradient orientation is denoted by θ_p. In this pixel consistency, the Hit map $\mathcal{M}^\mathcal{T}$ of a target image contour $\mathcal{Q}$ with N_Θ oriented quantized channels, and each channel is a binary map $\mathcal{M}^\mathcal{T}_\theta$, $\theta \in \Theta$, where r is the tolerance radius, given in pixels, and $|L_\mathcal{Q}|$ is the number of edgels of $\mathcal{Q}$. The *hit* of edgel between $\mathcal{Q}$ and $\mathcal{T}$ is defined in Equation 4.8.

$$Hit_\mathcal{Q}(p) = \begin{cases} 1 & \exists_p \in \mathcal{L}_\mathcal{Q}(\parallel X_\mathcal{Q} - X_\mathcal{T} \parallel_2 \leqslant r \;\&\; \theta_\mathcal{Q} = \theta_\mathcal{T}) \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

$$P_{\mathcal{T}\rightarrow\mathcal{Q}} = \frac{1}{|\mathcal{L}_\mathcal{Q}|} \sum_{p\in\mathcal{L}} Hit_\mathcal{Q}(p) \tag{4.9}$$

In Sketch-Finder 2.0, the index can be visualized as two indexes, both are based on the inverted list strategy. The first index is used for the compressed-domain of the

wavelet coefficients (wedgels) and the second is used for the pixel domain of edgels. The first index structure was presented in Section4.1.5 for Sketch-Finder 1.0 and the second index, used to compute the OCM on Sketch-Finder 2.0, is presented on the following.

4.2.3 Neighborhood Edgel Map Index

In order to save computational cost, instead of computing the neighborhood edgel map in the query time, we can compute the neighborhood inside a radius r for all images of the dataset in the indexing time, thus, saving this precomputed data in a form of inverted file lists to compose an index of neighborhood of edgels. These inverted lists are similar to those representing wedgels, however, in such case, these lists of image IDs represent, therefore, the neighborhood of edgels used in the *Hit* function (Equation 4.9).

The neighborhood edgels index is responsible for the pixel consistency verification using the OCM. For each neighborhood edgel (x, y, θ) of the dictionary, we store the list of image IDs where there is an edgel in the neighborhood of each image. Instead of comparing the radius r of all dataset edgels to the strokes of the sketch, we just have to retrieve the set of sketch edgel $\mathcal{L}_\mathcal{Q}$ lists, giving a computational cost of $O(\mathcal{L}_\mathcal{Q})$ lists to process. In a real time application, we can compute the neighborhood of the edgel maps in the query time only for the z best ranked images of the compressed-domain measure, *i.e.*, the wavelet similarity ($\mathcal{W}_{\mathcal{Q},\mathcal{T}}$), thus, exchanging memory space cost by a small computational cost.

Figure 4.7 presents the neighborhood edgel map index structure. Figure 4.7 (a) illustrates the three neighborhood edgel maps, Figure 4.7 (b) represents the neighborhood edgel dictionary (x, y, θ), and Figure 4.7 (c) represents the list of image IDs associated to each edgel of the dictionary. In this figure, we illustrate just three orientations, aiming simplification of the scheme, although, actually we have six orientations.

Although the computational cost is saved in query time, the only drawback of this strategy is storage, once that it changes computational cost by memory space. However, a second option that can be applied to a industrial product is, to only compute the neighborhood edgels for the best classified images on the wavelet similarity measure. We did not use this second scheme aiming to have an accurate precision for the experiments of this work.

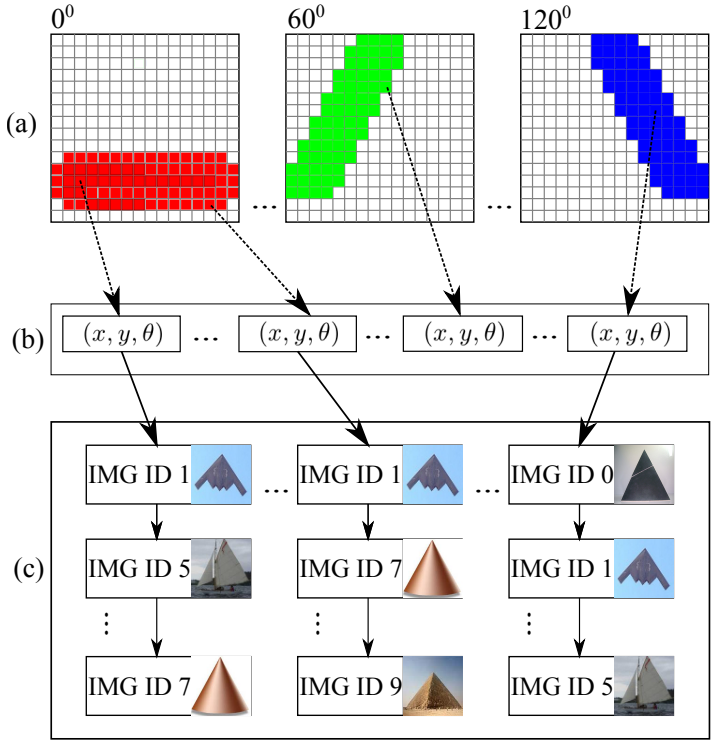

Figure 4.7. Neighborhood edgel indexing using inverted lists: (a) neighborhood edgel map $\mathcal{G}_\theta^{\mathcal{I}}$; (b) edgels dictionary; and (c) inverted list of image IDs.

4.3 Conclusion

In this chapter, we presented our approach for efficient and effective sketch-based image retrieval in two versions. The first one, presents better efficiency due to the use of the compressed-domain index. The second one also uses, besides the compressed-domain, the pixel domain index in order to effectiveness improvement.

Further, we presented in details in this chapter, the similarity measure for comparing sketches in the proposed compressed-domain and pixel domain index, likewise, how to build the index and compare images with the sketch. Inverted lists of image IDs are employed to build the dataset index and this strategy improves the efficiency of our method.

Although Sketch-Finder also uses the pixel domain, it is still more efficient than Mind-Finder by two reasons. First, the most computationally expensive comparison on the pixel domain of Mind-Finder is changed by our faster comparison on the compressed domain, and second, the other pixel domain verification performed in both approaches is not expensive as the first performed by Mind-Finder. Also, only the compressed-domain retrieval presents an effectiveness quite good and near to Mind-Finder effectiveness.We can always allow the use of this technique when the query time

answer is imperative, like in the practical application that we present in Chapter 7.

In next chapter, we present three datasets used in our experiments to measure efficiency and effectiveness of our method and compare with other approaches. We present the Paris dataset, a small collection of images mostly used to measure effectiveness; a big dataset, obtained from *ImageNet*, mostly used to measure efficiency; and a third dataset, *Flickr15K*, is used to compare our approach to several others.

Chapter 5

Experimental Setup and Image Dataset Analysis

In this chapter, we describe three image datasets used in our experiments. The first dataset, named **Paris dataset**[1], is a collection of more than six thousand images. This is a relatively small dataset used to set the parameters of our approach. The huge volume of experiments to find the right parameters requires a small dataset because each evaluation, even on a small set of images, demands lots of CPU time to index and perform all the queries. The second dataset, with more than 535 thousand images issued from ImageNet[2] is used to evaluate both, effectiveness and efficiency. This big dataset is important to evaluate the behavior of our approach with big data, what is also one of the goals of the present dissertation. A third dataset with 15 thousand images named *Flickr15K* is used in our experiments to compare our approach with others found in the literature.

Additionally, in this chapter we describe the methodology used to evaluate our approach, as well as, the sketches, the ground-truths, and the metrics used in our experiments.

5.1 Image Datasets

The **Paris Dataset** is a homogeneous collection of 6,412 images collected by Visual Geometry Group (VGG) on *Flickr*. The dataset is grouped by 11 particular famous landmarks of Paris (*La Defense, Tour Eiffel, Hotel des Invalides, Musée du Louvre,*

[1]Visual Geometry Group – `http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/index.html`

[2]ImageNet – `http://www.imagenet.org/`

Moulin Rouge, Musée d'Orsay, Notre Dame, Panthéon, Centre Pompidou, Sacré Cœur and *Arc de Triomphe*). Also, there is one general category for all kinds of images from Paris. Some image examples of the Paris dataset are presented in Figure 5.1.

Figure 5.1. Image examples from Paris dataset. From left to right: *La Defense, Tour Eiffel, Hotel des Invalides, Musée du Louvre, Moulin Rouge, Musée d'Orsay, Notre Dame, Panthéon, Centre Pompidou, Sacré Cœur* and *Arc de Triomphe*.

The **Building and Vehicle ImageNet Dataset** is a subset of *ImageNet2011* fall release. The *ImageNet2011* fall release is a dataset with more than 14 million images organized according to the WordNet [Miller, 1995] hierarchy, where each node of the hierarchy is depicted by hundred and thousands of images. Inside this hierarchy, with more than 21 thousand categories, two main nodes and all their subcategories were selected: **Building Edifice** with code node `n02913152` and **Vehicle** with code node `n04524313`. These two categories sum a total of more than 535,000 images. Some images of this subset are shown in Figure 5.2. From this point up, we just refer as ImageNet, the subset of 535 thousand images collected from *ImageNet2011* fall release, as described in this paragraph.

A third dataset, *Flickr15K*, from Hu and Collomosse [2013], was used to compare our approach with the one using the Gradient-Field HOG descriptor and the other descriptors for BoVW, all presented in Hu and Collomosse [2013]. *Flickr15K* is a dataset with 14,660 images issued from *Flickr*. Hu and Collomosse [2013] also present a set of 330 sketches partitioned into 33 categories that we used in our queries. The categories of the *Flickr15K* dataset contain shapes, *e.g.*, hart, etc; landmarks, *e.g.*, Eiffel Tower, Big Ben, etc and objects, *e.g.*, bike, airplane and etc.

It is important to note that the ground-truths we employed for these datasets were not designed specifically for SBIR, but rather corresponds to a higher semantic level. The images inside each class may have the same object at different viewpoint, shape, rotation, scale and/or position. It follows that the performance measure we obtained

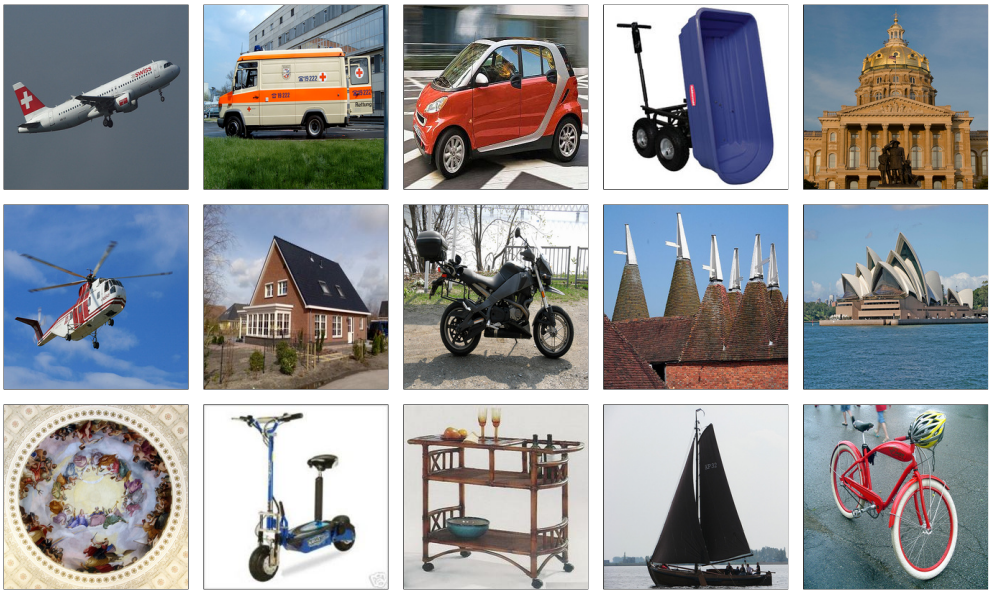

Figure 5.2. Image examples from ImageNet Subset. In this figure, each image represents one of the following classes: airplane, ambulance, automobile, dump-cart, governmental building, helicopter, house, motorcycle, oast-house, opera house, rotunda, scooter, serving-cart, ship and velocipede cart.

with these ground-truths are, naturally, lower than should be expected on a ground-truth specifically designed for SBIR. There are, however, many images showing very similar items from similar viewpoints and having similar positions, that are relevant for the evaluation of SBIR. Furthermore, since our approach and the method in [Cao et al., 2011] make similar assumptions regarding the queries and used the very same ground-truths and evaluation methods, we believe the comparison is fair on these ground-truths.

5.2 Dataset Analysis

To better know and understand the image datasets, we performed a statistical analysis of contours on the Paris and ImageNet datasets. The analysis estimates the distribution of the edgels (x, y, θ) in six orientations, as well as, the density of edges (x, y) in rows and columns.

5.2.1 Paris Dataset Analysis

The contours of the Paris dataset are analyzed in three histograms: one histogram for the edgel distribution, which represents density of the contours in six quantized orientations, and two other histograms for edge analysis respectively on rows and columns.

Figure 5.3 presents the histogram of probability for each one of the six orientations. The fist orientation is the horizontal and the others follow anticlockwise orientation. This distribution used the contours of the dataset obtained from the UCM algorithm [Arbelaez et al., 2011], using images with resolution of 256×256 pixels.

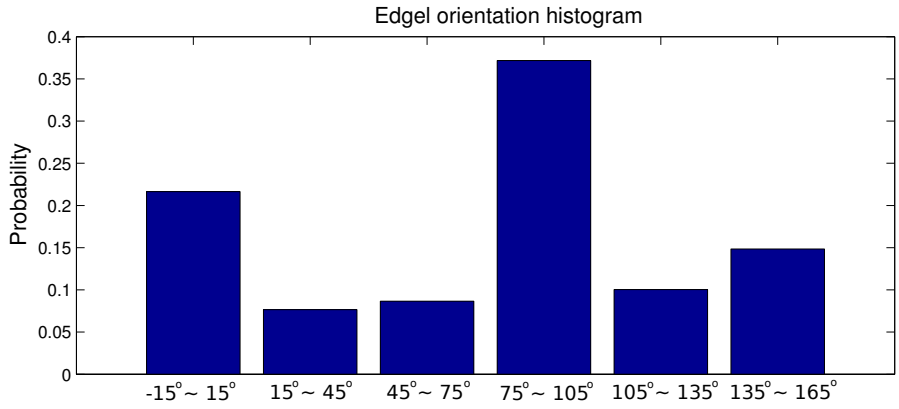

Figure 5.3. Paris dataset edgel orientation histogram. Horizontal ($-15° \sim 15°$) and vertical ($75° \sim 105°$) contours are more dense, probably because the dataset has several buildings where these orientations are more common.

As shown in Figure 5.3, the vertical orientation ($75° \sim 105°$) is the most dense, with more than 35% of the edgels. We believe that the Paris buildings, full of horizontal and vertical lines, contributed to this concentration. Further, the high number of Eiffel Tower images contributed to the vertical lines dominance.

With the histogram presented in Figure 5.3 it is not possible to visualize the spatial distribution of the edgels. Additionally, the spatial distribution of the edgels in six orientations can be analyzed in Figure 5.4. As presented, some areas are more dense than others, where the more black the region, the more dense is the area in edges, and vice-versa. For example, in Figures 5.4 (c) and (e), some dense regions in the middle are self-asymmetrical but symmetrical to one each other between (c) and (e).

The histogram distribution of edges in rows for the Paris dataset is shown in Figure 5.5. As presented in this figure, the bottom is more dense in edges rather than the top in most part of images. We believe this occurred influenced by outdoor images with sky and clouds. These images present less contours on the top rather than bottom, full of buildings, cars and people. Figure 5.4 (g) also shows that top left and right regions of the images are less dense on edges in several images of the dataset. This density estimation was obtained by summing all the edges in each row of the dataset divided by the total number of edges.

The histogram distribution of edges in columns for the Paris dataset is shown in Figure 5.6. The edge distribution in columns is more homogeneous than in rows,

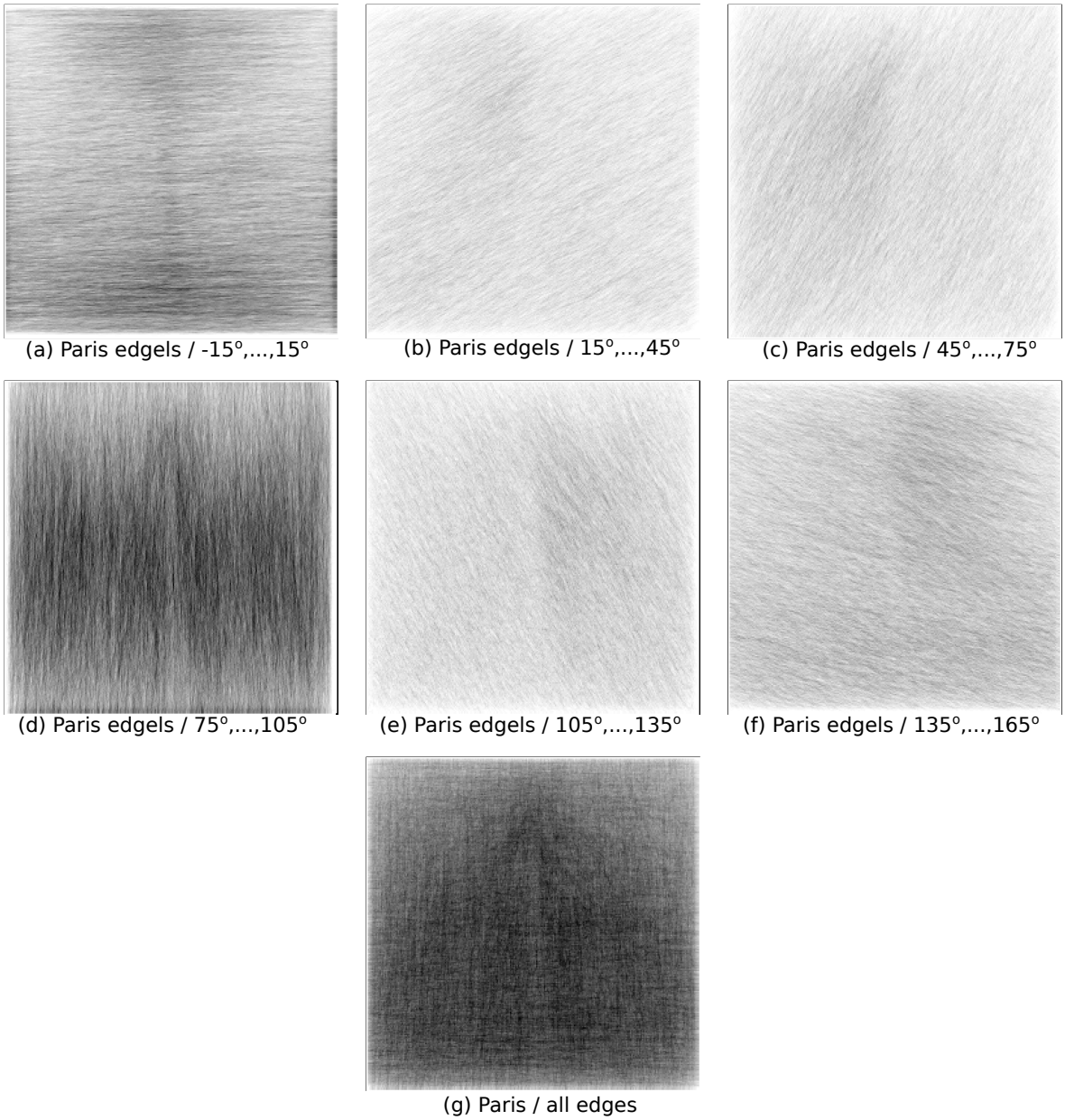

Figure 5.4. Paris dataset spatial edgel distribution in six orientations. In this figure, the average spatial distribution of edgels, from all images of the Paris dataset. Each image from (a) to (f) presents the density in one of the six quantized orientation. It is also presented in (g), the average of spatial edges distribution of all images, without considering orientation.

with a little concentration in the middle of the images, and few edges on the borders. The occurrence was obtained by summing all the edges in each column of the dataset divided by the total number of edges.

As presented in Figure 5.3, vertical and horizontal lines are the two most dense oriented edgels. We believe the reason is the presence of buildings and other architectural

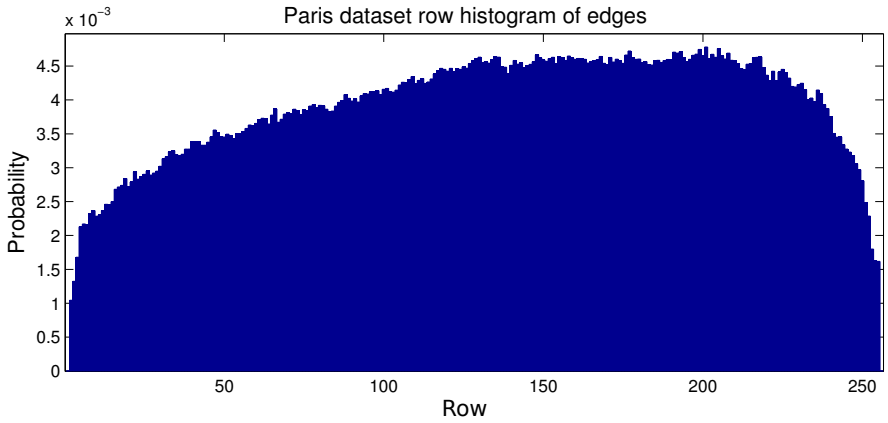

Figure 5.5. Paris dataset row histogram of edges. This distribution shows that the bottom of the images in this dataset tends to be more dense due to the presence of buildings, cars and people rather than top of images, frequently full of sky and clouds.

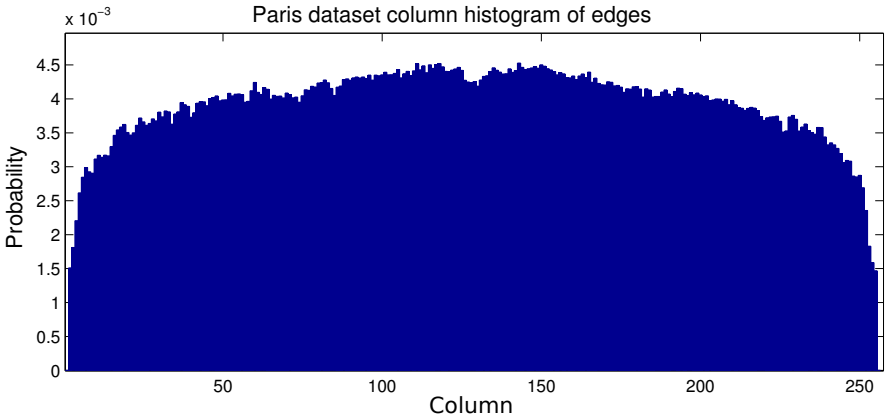

Figure 5.6. Paris dataset column histogram of edges. The edges are more concentrated in the middle of the images of this dataset.

landmarks of Paris, specially *Notre Dame, Centre Pompidou* and *Arc de Triomphe*. We know that these landmarks are mostly composed by vertical and horizontal lines. Once this dataset is not heterogeneous, some concentration may occur. Another reason for edges concentration in the middle of images, is that, photographers tend to centralize the object of interest. Row and column histograms (Figures 5.5 and 5.6) show that these image borders (top; bottom; right; and left) are less dense than central area.

5.2.2 ImageNet Dataset Analysis

As for Paris dataset, the contours of the ImageNet dataset are analyzed in three histograms: one histogram for edgel distribution, which represents the density of the

contours in six quantized orientations, and two other histograms for edges analysis respectively in rows and columns. Figure 5.7 presents the histogram of probability for each orientation. The fist orientation is the horizontal, the forth is vertical and the others are intermediate orientations. This distribution used the contours of ImageNet dataset obtained from the UCM algorithm [Arbelaez et al., 2011], using images with resolution of 256×256 pixels.

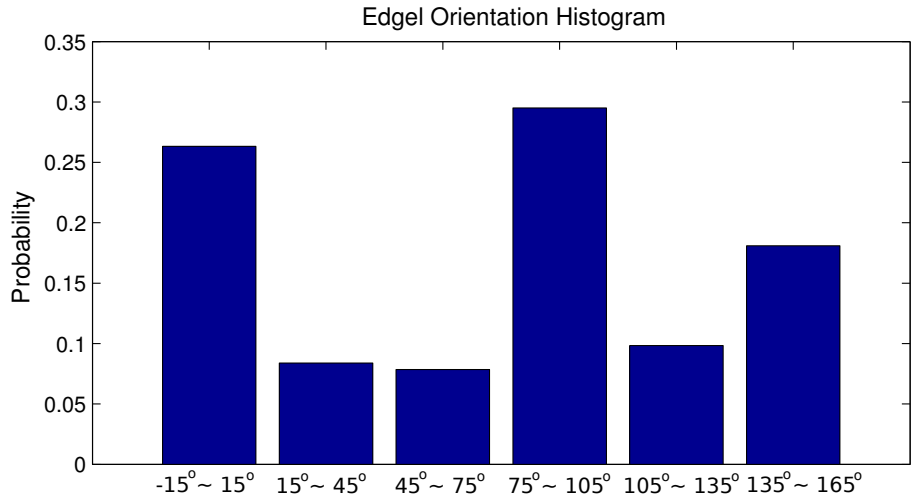

Figure 5.7. ImageNet dataset edgel orientation histogram. Horizontal ($-15° \sim 15°$) and vertical ($75° \sim 105°$) contours are more dense, probably because the dataset is mostly composed by buildings and vehicles, whose orientations are more common.

In the histogram presented in Figure 5.7, it is not possible to visualize the spatial distribution of the edgels. However, the spatial distribution of the edgels in the six orientations can be analyzed in Figure 5.8. As shown in this figure, some areas are more dense than others, where the more black the region, the more dense is the area in edges, and vice-versa. For example, in Figures 5.8 (a) the central area is dense in the middle for horizontal contours, as well as, in (d) for vertical contours. The other orientations present a homogeneous distribution with few concentration in some areas. In the same figure, in (g) we also present the density distribution of edges without considering orientation. In this figure, the center is denser rather than the borders. All the images are averaged contours obtained from all images of the dataset (535 thousand).

The histogram distribution of edges in rows for the ImageNet dataset is shown in Figure 5.9. This histogram presents a Gaussian shape and shows that on average, the middle of the image is the most dense part in edges inside, considering rows. Different of the row distribution in Paris dataset, this distribution is more symmetric, *i.e.*, the

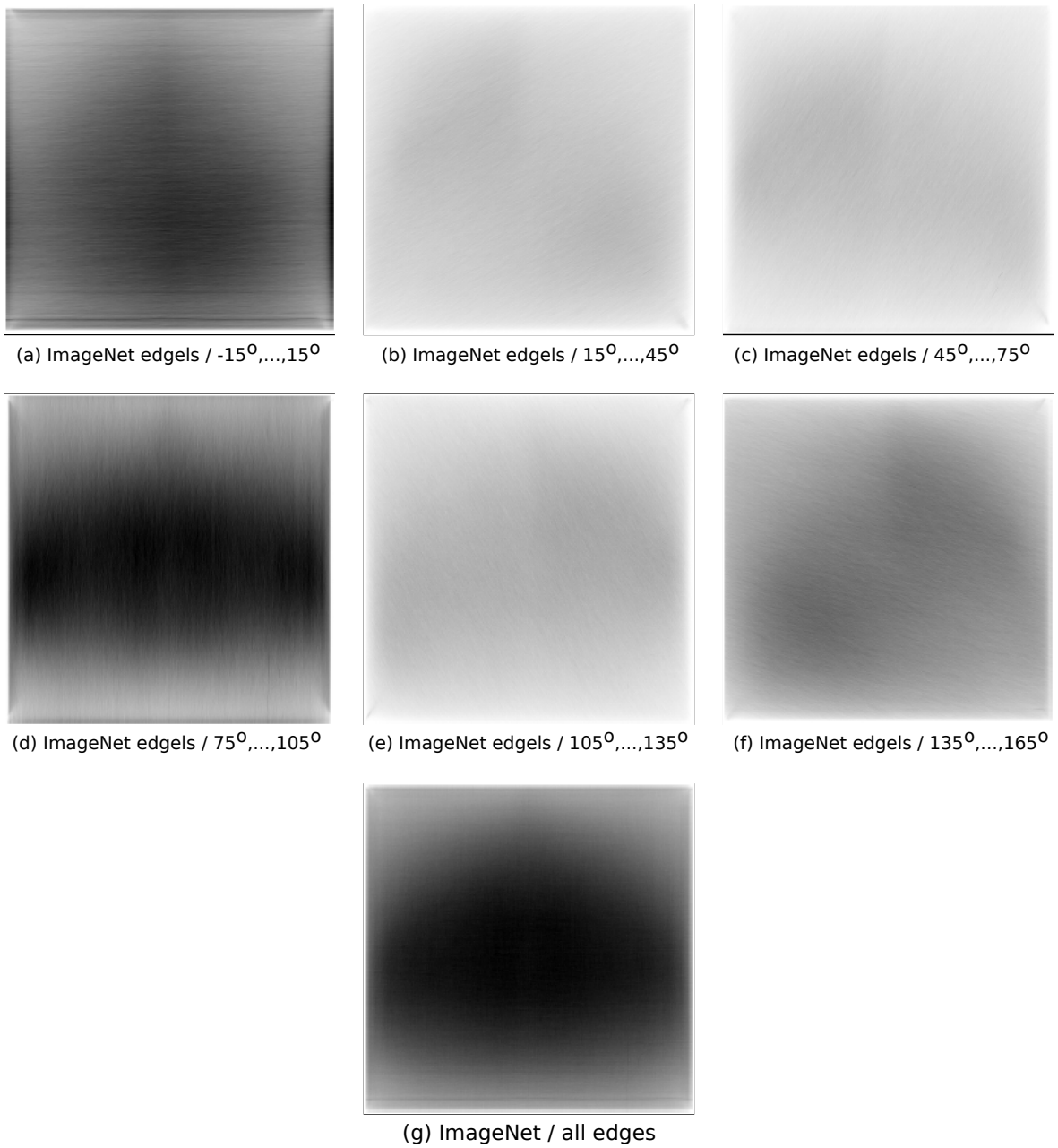

Figure 5.8. ImageNet dataset spatial edgel distribution in six orientation. In this figure, the average spatial distribution of edgels, from all images of the ImageNet dataset. Each image from (a) to (f) presents the density in one of the six quantized orientation. It is also presented in (g), the average of spatial edges distribution of all images, without considering orientation.

density of edges in rows increases from top and bottom of the images to the center, almost in symmetry. This dataset has more indoor images, what may explain this behavior.

The histogram distribution of edges in columns for the ImageNet dataset is shown

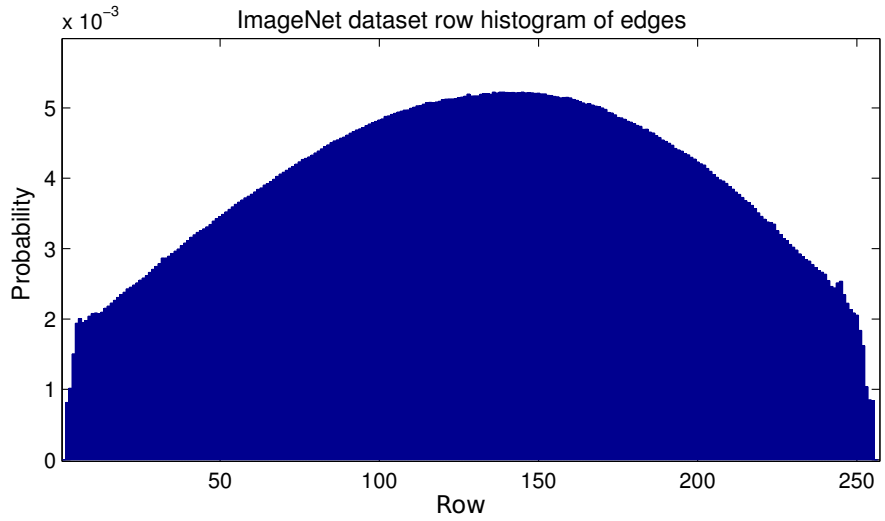

Figure 5.9. ImageNet dataset row histogram of edges. This distribution shows that, on average, edges are more concentrated in the middle rows of the images.

in Figure 5.10. The edge distribution in columns is more homogeneous than rows distribution, with a little concentration in the middle of the images and few edges on top and bottom borders. The center of the image is also the most dense contour region in columns, however the distribution is less concentrated than edges in rows. This occurrence was obtained by summing all the edges on each column of the dataset divided by the total number of edges.

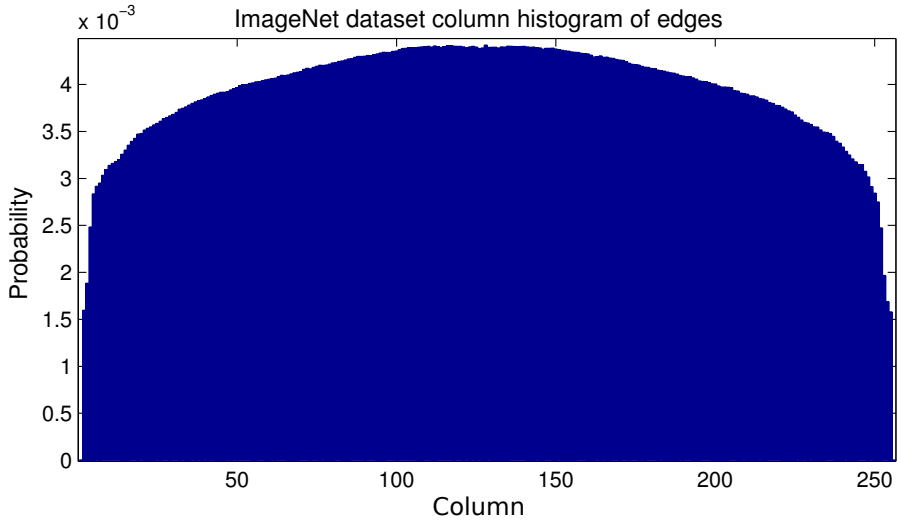

Figure 5.10. ImageNet dataset column histogram of edges. The edges are more dense in the middle of the images in this dataset, specially from the 100th to the 150th column, presenting in average, almost the same density.

As in the Paris dataset, the ImageNet orientations are more dense in horizontal

and vertical lines, see Figure 5.7. The histogram distribution in the orientations maps is quite similar to the Paris one. We believe it is because the subset of the ImageNet is not homogeneous and the occurrence of buildings in this dataset contributes with a large number of horizontal and vertical lines. Further, photographers tends to centralize the object of interest in his photo, that explains the concentration of vertical and horizontal contours in the middle of the images. In row and column histogram, we also perceived that the images borders: top; bottom; right; and left, are less dense parts of the images in edges of the ImageNet dataset.

5.2.3 Wedgel Distribution in the ImageNet Dataset

In this section, we present an evaluation on the probability of wavelet coefficient occurrence at spatial position (x, y) of the compressed domain. For this analysis, we used the ImageNet dataset, with neighborhood edgel map of radius r = 20 pixels, Haar wavelet transform, and 40 coefficients per neighborhood orientation map, *i.e.*, 20 most negative and 20 most positive coefficients of each orientated neighborhood map. In this analysis, we consider the occurrence in all six quantized orientations, and with both signs of the coefficient (positive and negative) for neighborhood edgel maps of 256×256 rows and columns. In this configuration, shown in Figure 5.11, 99.88% of the coefficients are concentrated in the first 64×64 rows and columns. Besides understanding the index distribution of the dataset, the wedgels distribution analysis is also used in some classical Information Retrieval rank functions like "*term frequency inverse document frequency*" and BM25 that we evaluated in our approach as presented in Section 6.2.4.

Figure 5.11 presents the histogram of probability of coefficient occurrence in the space of Haar wavelet transform. Although in our experiment this space is 256×256, we present just the first 64×64 rows and columns, since it concerns almost 100% of the coefficients. Therefore, this part of the histogram presents better scale for printing and visualization. The most probables concurrency of coefficients is concentrated near to coefficient position $(0, 0)$, *i.e.*, the near the coefficient is to the coordinate $(0, 0)$, the most probable it tends to exist. Further, these most significant coefficients are also related to low scales components in the wavelet domain.

An other visualization of the histogram, in more details, is presented in Figure 5.12. This analysis, in a segment of the first 16×16 rows and columns, represents 91.78%, of the coefficients for the indexed ImageNet dataset.

Although the proposed wedgel dictionary maximum size is $256 \times 256 \times 2 \times 6 = 786432$, as described in Section 4.1.1, in practice, the real number of visual words is much smaller due to the concentration of coefficients near to the coordinate $(0, 0)$.

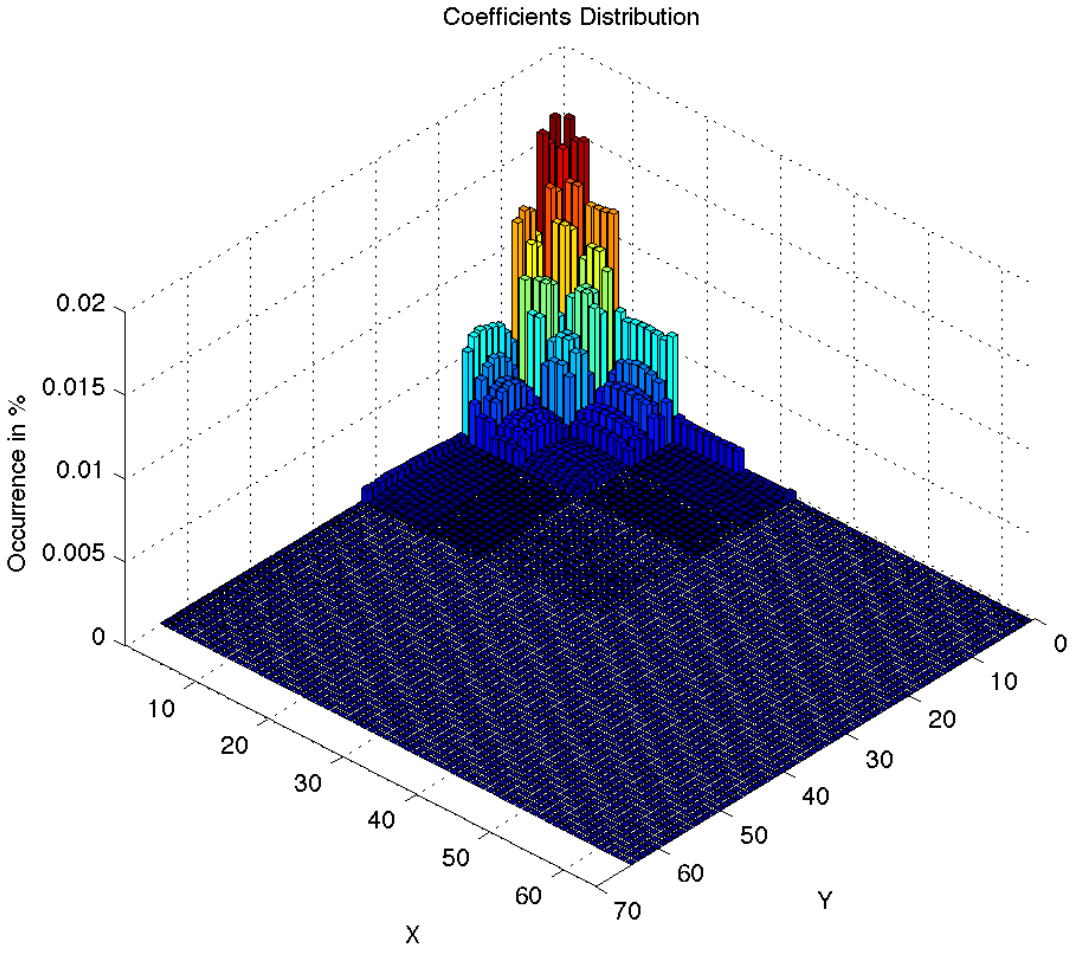

Figure 5.11. Histogram of coefficient occurrence in 64×64 rows and columns wavelet domain space. In this probability analyses, we joined all six quantized oriented neighborhood edgel maps, transformed into the wavelet domain, considering both signs of large coefficient magnitude, negative and positive at position (x, y). In this area, 99.88% of the coefficients are present, from the complete space of 256×256 rows and columns.

Informally, the farther is the coefficient from position $(0, 0)$, the less likely is to have a wedgel word containing a inverted list of image IDs.

There is a relation between the radius r of the neighborhood edgel map and the spatial distribution of coefficients. The grater is the radius, the more low frequency coefficients of large magnitude are present in the wavelet domain, thereafter more low frequency of large magnitude coefficients are present. Another relation is the radius size and the number of non null, or non empty, inverted lists of image IDs in the dataset index. Table 5.1 presents the relation between the radius size and the number of non

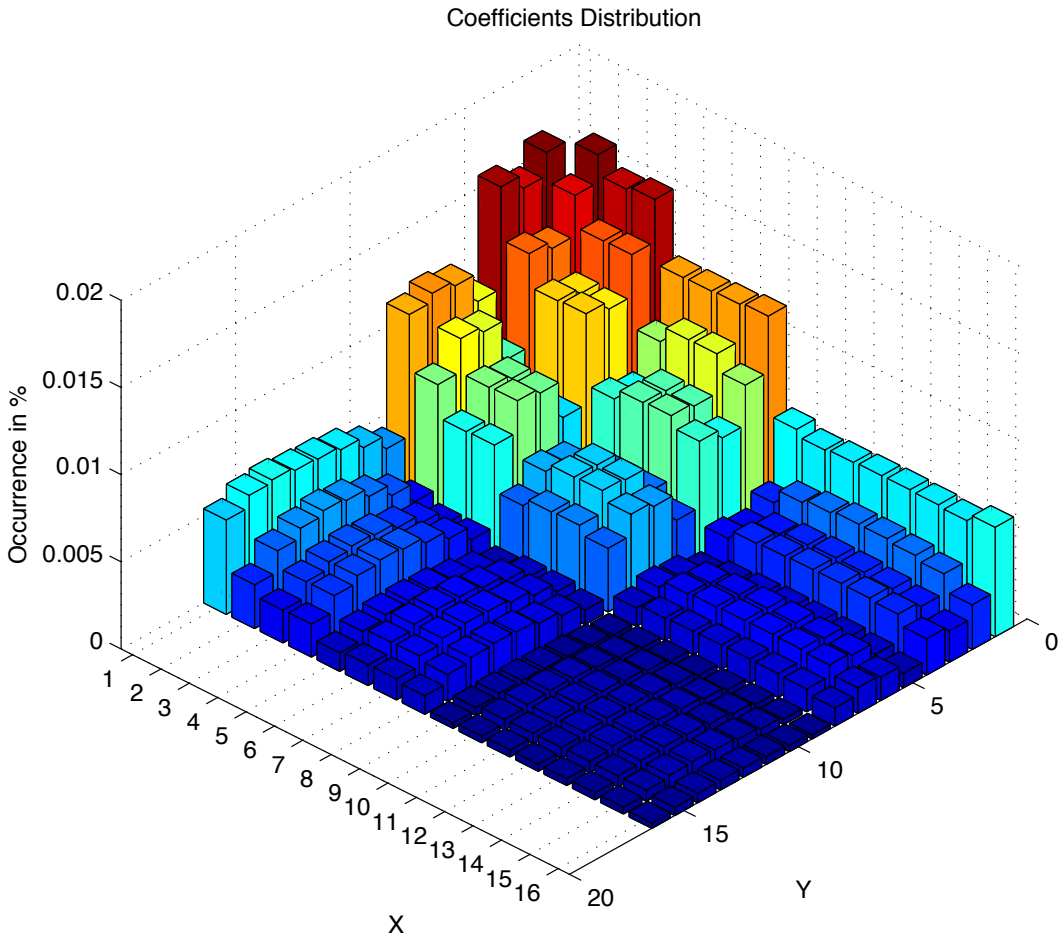

Figure 5.12. Histogram of coefficient occurrence in 16×16 rows and columns wavelet domain space. This figure is a segment of Figure 5.11 in the most probable coefficients, that represents in this subarea, 91.78% of the coefficients from the complete space of 256×256 rows and columns.

null inverted lists. This data was also obtained also from the ImageNet dataset. All the indexing experiments presented are taken from ImageNet dataset with 40 coefficients per orientation map, *i.e.*, 20 most negative and 20 most positive coefficients. We present the variation in the radius r size of the neighborhood edgels, before the wavelet transform, affecting the Number of Inverted Lists (NIL). We also present in Table 5.1 the percentage of words used of the dictionary.

Table 5.1. Number of Inverted Lists on the dataset index. In the first column, we present the variation on the radius r size (given in pixels) of the neighborhood edgel map. In the second, the Number of Inverted Lists (NIL), and in the third column, used words, the percentage of Inverted Lists according to all possibilities of the dictionary.

Radius r	**NIL**	**Used words** (%)
2	418783	53.25
5	149378	18.99
10	59808	7.60
15	43667	5.55
20	37102	4.72
25	31270	3.98
30	29905	3.80

5.3 Paris Sketch Dataset

Collecting sketches from different users is an important task to compose a set of hand drawn images to evaluate the performance of SBIR approaches.

In the beginning of this work, the lack of sketch datasets motivated us to build our own set. Once the VGG Paris dataset was not originally designed for sketch-based image retrieval, it has no sketches for SBIR evaluation. In order to fill this gap, we asked some voluntaries to draw some sketches representing the Paris landmarks. The voluntaries were shortly introduced about the objectives of the work, the main concept of SBIR, and what was the reason for producing those sketches. Aiming to light the idea of what kind of sketch we were expecting to have, four sketch examples were shown. For this task, voluntaries were instructed to draw sketches of 11 different landmarks of the dataset. Ten users, some of them with more than one example per landmark, outlined 132 sketches. The document instructing the voluntaries on how to produce their sketches is presented in Attachment A of this dissertation. That document instructed the voluntary to draw free hand line sketches, at any position, scale and/or rotation that he/she had in mind. For inspiration, we gave, attached to the drawn instructions, six example images for each Paris landmark at different rotation, angle and scale (see Attachment A). The creation of these sketches for the Paris dataset is one of the contributions of this dissertation work. Figures 5.13 and 5.14 present some sketches, produced by the voluntaries, for the Paris dataset SBIR evaluation.

As show in Figures 5.13 and 5.14, some users are not very skillful in drawing.

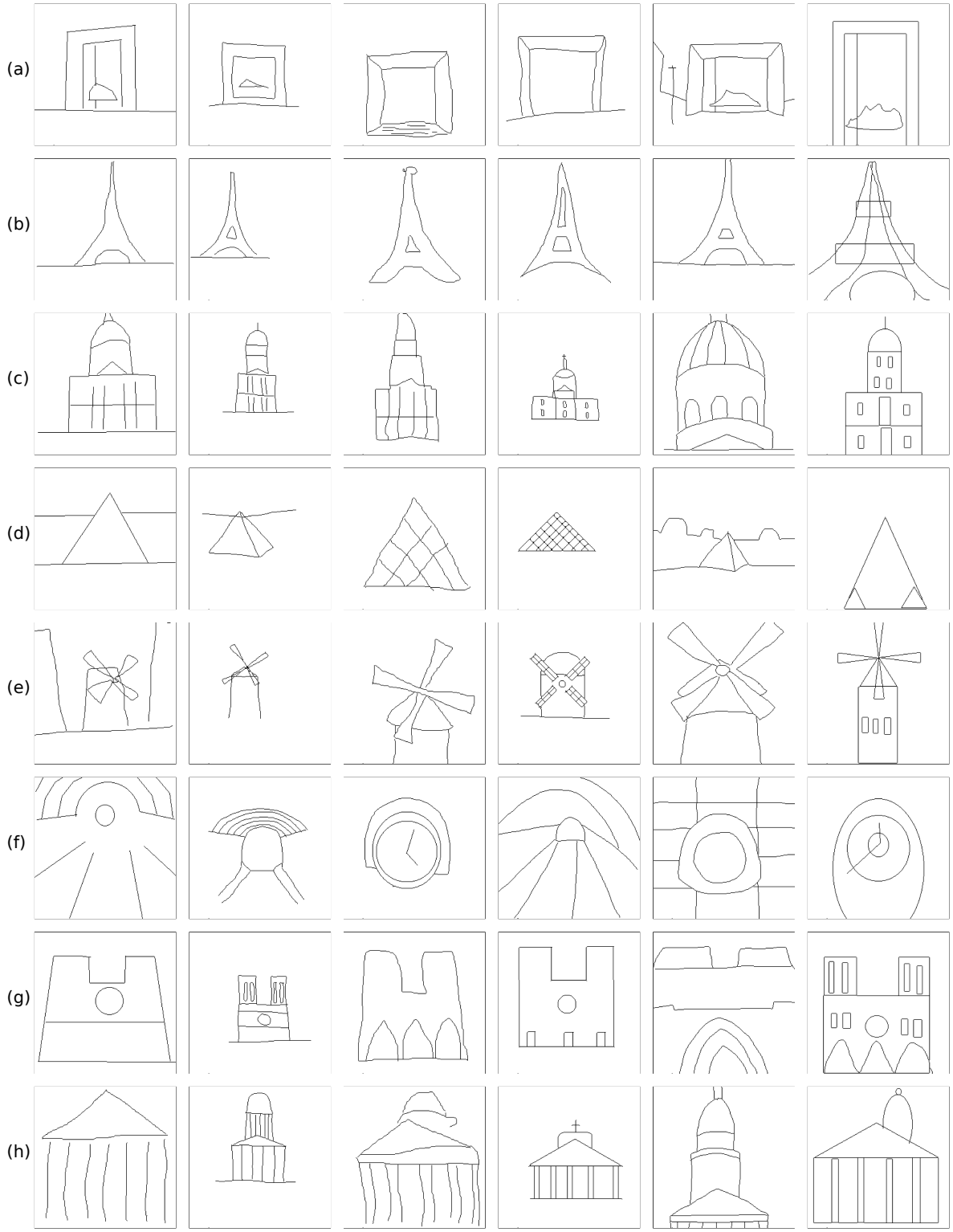

Figure 5.13. Real examples of Paris sketches: (a) *La Defense*; (b) *Tour Eiffel*; (c) *Hotel des Invalides*; (d) *Musée du Louvre*; (e) *Moulin Rouge*; (f) *Musée d'Orsay; (g) Notre Dame and (h) Panthéon.*

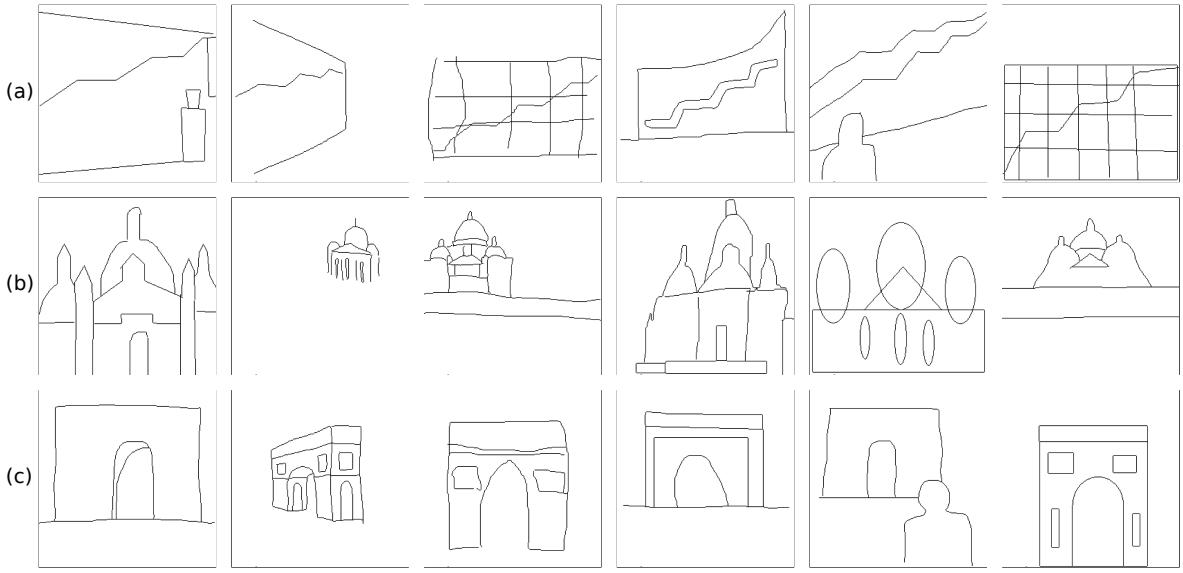

Figure 5.14. Real examples of Paris sketches: (a) *Centre Pompidou*; (b) *Sacré Cœur* and (c) *Arc de Triomphe*

Further, other users related some difficulty on using the mouse device to produce the pictures. Actually, bad sketches, far from the reality, are one of the difficulties of the query-by-sketch. As cited by [Bird et al., 1999], users blamed their own unreal sketches when the results were not good.

5.4 Ground-Truth

To gauge and measure the perceptual similarity of the queries, a ground-truth information created from the user perception is necessary [Eitz et al., 2011]. For the Paris dataset, a ground-truth mostly used for image classification is available in the VGG group web page. To overcome this problem, we used the same ground-truth of VGG for image classification of the Paris landmarks. For example, for the sketches of Eiffel Tower, we used the same ground-truth of Eiffel Tower already existent for the classification of this same landmark.

Using the VGG Paris dataset ground-truth, we constructed one general image contour shape for each landmark category. This general image was constructed by averaging the contours of each natural image contour of the ground-truth list for its respective landmark. These averaged images give us a visual pattern shape of the Paris landmarks revealing the homogeneity of each ground-truth category. The average of each landmark uses the contours obtained by the Ultrametric Contour Map [Arbelaez et al., 2011]. The images are shown in Figure 5.15. As we observe, some monuments

present a general and well defined shape like, *Tour Eiffel*, *Notre Dame*, *Centre Pompidou* and *Arc de Triomphe*. The *Moulin Rouge* landmark shape is more heterogeneous, however it is expected due to the moving blades of the windmill, which generated pictures with several views.

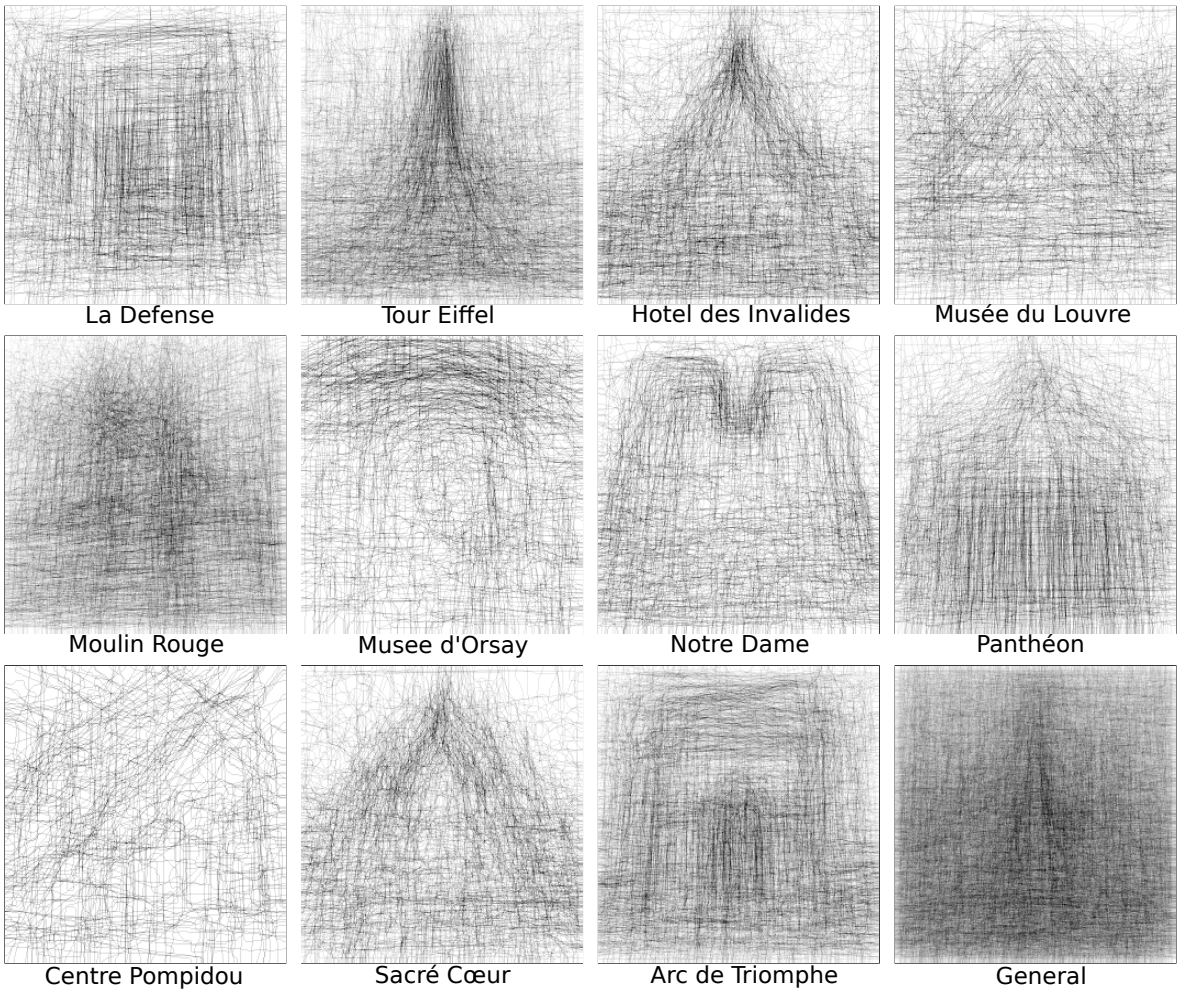

Figure 5.15. Average Ultrametric Contour Map (UCM) distribution by landmark category of the Paris dataset. These averaged contours used the Visual Geometry Group (VGG) ground-truth.

5.4.1 Building the ImageNet Ground-Truth

As described in Section 5.1, we collected the subset of images from ImageNet. Aiming to evaluate the SBIR approaches on this dataset, it makes necessary to build their respective ground-truths.

For each one of the ten selected vehicle categories of the ImageNet subset (*dump-cart, automobile, scooter, serving-cart, velocipede, airplane, helicopter, ship, motorcycle*

and *ambulance*), and five building categories (*governmental building, house, rotunda, oast house* and *opera house*) we built a ground-truth list containing the expected images.

One problem for building ground-truths is the subjective character of SBIR. It is difficult to set a ranking dataset of assorted images in terms of visual similarity once that different users may have different assumptions about the similarity evaluation of images [Gupta and Jain, 1997]. In this case, we assumed some criteria for the ground-truth construction.

Following the criteria of the Paris ground-truth, the ImageNet ground-truth contains all the objects of each category, independently of their scale, rotation, position or other visual information, like background and even other objects in the same correct picture. It makes the ImageNet ground-truth more semantic rather than structural, but this strategy was adopted in order to build more quickly the ground-truth. Further, as the compared approaches use the same ground-truth for evaluation of effectiveness, then, neither one neither other may have benefits, what makes this comparison fair.

The selection of the classes to build the ImageNet ground-truth followed the criteria of simplicity that we expect from the object in terms of easy drawing its sketch image, as well as, the homogeneity of the class and the simplicity of the background of most images. Some object category appear in several imageNet nodes, usually with the same position and visual similarity. These objects are repeated in several nodes because of their semantic classification, although the object is still the same on its general abstraction. On example is the airplane main class, that is present in several sub classes with the following denominations: *airbus, airliner, airplane, aircraft, fighter aircraft* and so on. Table 5.2 presents the categories and the selected images for the ground-truth of each class.

For the ImageNet dataset, we also averaged one image of each object class based on the image contours of the ground-truth image list. These images give us a visual pattern shape of each object class revealing the homogeneity of each ground-truth category. The averaged contour image of each landmark also uses the contours obtained by the Ultrametric Contour Map [Arbelaez et al., 2011] and these images are shown in Figure 5.16. As we observed, some object classes present a general homogeneous shape like, *dump cart, governmental building, motorcycle, opera house, rotunda, scooter, ship* and *velocipede cart*. The *airplane, automobile, helicopter, house, oast house and serving cart* classes are more heterogeneous on their shapes due to the bigger number of images in the class.

These ground-truths consider fifteen classes or subcategories of general *Edifice* and *Vehicle* categories. Even as in the Paris dataset, the ImageNet ground-truth

Table 5.2. ImageNet ground-truth. In this table we present the general class, the ImageNet labels of the selected images of the ground-truth, the ImageNet code nodes containing the image names, and the total of images belonging to the class.

Class	ImageNet Labels	Code Nodes	Images
Airplane	*Airbus; airliner; airplane, plane; amphibious aircraft; biplane; bomber; dive bomber; fighter aircraft; floatplane; flying boat; interceptor; jet plane; jetliner; jumbo jet; monoplane; multiengine plane; narrow-body aircraft; propeller plane; turbo-propeller plane; seaplane, hydroplane; single-propeller plane; bomber; stealth fighter; twinjet; warplane, military plane; widebody aircraft*	n02686121; n02690373; n02691156; n02704645; n02842573; n02867715; n03215191; n03335030; n03365231; n03373611; n03577672; n03595860; n03596543; n03604311; n03783873; n03798610; n03809312; n04012084; n04012482; n04160586; n04222723; n04308273; n04308397; n04503499; n04552348; n04583620	23947
Ambulance	*Ambulance; funny wagon;*	n02701002; n03404012	1750
Automobile	*Car, auto, automobile, motorcar; compact car; motor vehicle, automotive vehicle; roadster; sedan; shooting brake*	n02958343; n03079136; n03791235; n04097373; n04166281; n04201733	9235
Dump Cart	*Dumpcart*	n03255899	976
Gov. Building	*Capitol; chancellery; customshouse; diplomatic building; statehouse; town hall*	n02956699; n03005033; n03152303; n03203806; n04305210; n04461437	5845
Helicopter	*Helicopter, chopper, whirlybird, eggbeater; shuttle helicopter; single-rotor helicopter*	n03512147; n04212467; n04223066	1940
House	*Boarding house; cabin; chalet; country house; dacha; log cabin; manse; safe house; saltbox; shooting lodge, shooting box; sod house, soddy, adobe house; summer house; tract house*	n02857477; n02932400; n03002816; n03118969; n03158186; n03180865; n03322836; n03465605; n03544360; n03685486; n03686924; n03718935; n04125541; n04131368; n04202142; n04255899; n04354026; n04465050	16098
Motorcycle	*Moped; motorcycle, bike; trail bike, dirt bike, scrambler*	n03785016; n03790512; n04466871	3894
Oast House	*Oast house*	n03837698	826
Opera House	*Opera, opera house*	n03849814	1288
Rotunda	*Rotunda*	n04112654	1216
Scooter	*Scooter*	n04149374	485
Serving Cart	*Pastry cart; serving cart; tea cart, tea trolley, tea wagon*	n03897634; n04176068; n04397027	2340
Ship	*Carrack, carack; felucca; ketch; pirate ship; privateer; rigger; sailing vessel, sailing ship; school ship, training ship; schooner; smack; square-rigger; windjammer*	n02968210; n03327133; n03612010; n03947888; n04006067; n04091584; n04128837; n04146862; n04147183; n04242408; n04244847; n04291992; n04587327	14220
Velocipede Cart	*Boneshaker; Velocipede*	n02869563; n04524716	921

constructed is semantically based, *i.e.*, we consider the occurrence of the object, independent of its position, scale and rotation.

For the queries, each category has five natural contours examples. Table 5.3 presents the query image names used on each class of the ImageNet used to evaluate efficiency and effectiveness.

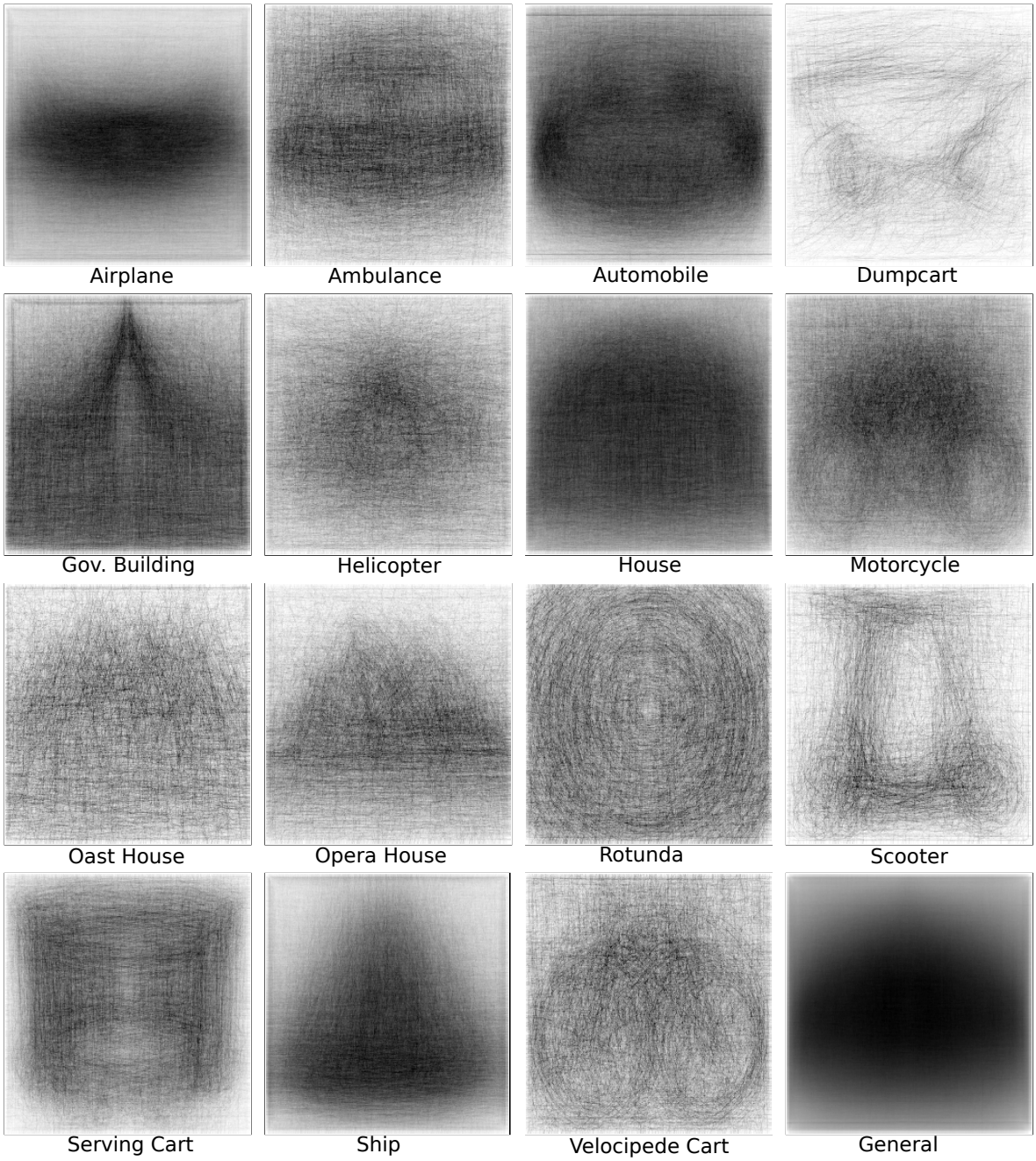

Figure 5.16. Average Ultrametric Contour Map (UCM) distribution by object class of the ImageNet ground-truth. In this figure, some object classes present a general homogeneous shape like, *dump cart, governmental building, motorcycle, Opera House, rotunda, scooter, ship* and *velocipede cart*. By the other hand, the classes *airplane, automobile, helicopter, house, oast house and serving cart* present a more heterogeneous contours shape. These heterogeneous classes are similar to the average of contours of all data set presented in the general averaged image on the bottom right of this figure.

Table 5.3. List of image queries used to evaluate efficiency and effectiveness on the ImageNet dataset. Each class has five image queries, column one presents the class and column two presents the image codes on ImageNet dataset.

Class	Images
Airplane	n02686121_{96; 256;} n03335030_11844; n03604311_7934; n02686121_684
Ambulance	n02701002_{1; 62; 171; 1499;} n03404012_4858
Automobile	n02958343_12; n03079136_1578; n03791235_4765; n04097373_23222; n04166281_8071
Dump Cart	n03255899_{6; 32; 3532; 3830; 3895}
Gov. Building	n02956699_62; n03005033_7819; n03093427_10687; n03152303_964; n04461437_3102
Helicopter	n04223066_{1051; 200;} n04212467_{2323; 119;} n03512147_16265
House	n02857477_18559; n02932400_8639; n03118969_10873; n03002816_2066; n03180865_13259
Motorcycle	n04466871_{24318; 15046; 12916;} n03790512_{20390; 1400}
Oast House	n03837698_{1200; 1793; 2336; 3791; 4948}
Opera House	n03849814_{2682; 2165; 4169; 5404; 9342}
Rotunda	n04112654_{203; 923; 1600; 5213; 11710}
Scooter	n04149374_{4; 74; 1118; 1762; 3673}
Serving Cart	n03897634_{33; 1203; 1541;} n04176068_427; n04397027_9071
Ship	n02793199_4738; n02901620_2917; n03045228_3887; n03186285_7575; n03612010_2784
Velocipede Cart	n02869563_{126; 1575; 2900; 3709;} n04524716_1727

5.5 Retrieval Performance Evaluation

In this section, we address the topic related to retrieval performance evaluation of CBIR and SBIR systems. The evaluation performance can be measured from two different perspectives: *efficiency* and *effectiveness*.

The efficiency is related to computational and memory costs, *i.e.*, the amount used of Central Process Unit (CPU), memory to keep the index and other metadada, and disk I/O, measured in bytes, to complete an image query. The resource allocation reflects on the query speed of retrieval. The meaning of speed is the time taken to perform a query. The faster the query is completed, the better the system is in terms of efficiency. The efficiency is also measured in memory and I/O reading, both in bytes. The less memory and I/O operations, the more efficient the system is. In our approach this relation of memory cost and CPU is direct because the less memory used to hold the dataset index, the less information the system has to process to complete a sketch query. Computational efficiency can be measured in time, CPU operations or algorithm complexity. Memory and I/O can be measured in bytes.

The effectiveness is related to the correctness or accuracy of the retrieved infor-

mation, in our case, the right classification of the desired images. The criteria employed to evaluate effectiveness of the image retrieval task must consider the number of correct images retrieved, and how well those images are classified. A good system must return not only the expected images, but also, in the first places in the classification order or rank.

Among several evaluation performance measures reviewed in [Müller et al., 2000; Del Bimbo, 1999] we detach and use in this dissertation, the classical Precision and Recall, inherited from Information Retrieval.

The precision measures the ability of the system to retrieve positive images relative to the number of retrieved images. The precision is presented in Equation 5.1.

$$Precision = \frac{\mathfrak{I} \cap \mathfrak{R}}{\mathfrak{R}} \tag{5.1}$$

By the other hand, recall measures the ability of the system to retrieve positive images relative to the total number of positives in the corpus. The recall is presented in Equation 5.2.

$$Recall = \frac{\mathfrak{I} \cap \mathfrak{R}}{\mathfrak{I}} \tag{5.2}$$

where $\mathfrak{I}$ represents the set of relevant images, and $\mathfrak{R}$ represents the set of retrieved images.

With precision and recall measures, we can obtain the precision×recall curve which is represented by a graph [Davis and Goadrich, 2006].

An other important measure based on precision×recall curve is the Average Precision (AP) and Mean Average Precision [Philbin et al., 2008]. The Average Precision is computed as the area under the precision-recall curve, and the bigger is the area, the better is the precision of the evaluated method. An ideal precision×recall curve has precision equal to 1, over all recall levels, which corresponds to an Average Precision of 1.

The Mean Average Precision is the mean of some set of AP computed respectively for some set of queries. This measure useful to summarize the precision×recall in a single value, helping the presentation and comparison of approaches. While the AP summarizes the precision×recall curve of one single query in a single value, the MAP summarizes the effectiveness of several queries.

5.6 Conclusion

This chapter introduced three datasets used in our experiments, the Paris dataset used for effectiveness evaluation and parametrization of our approach, the ImageNet dataset used to evaluate efficiency, and the Flickr15K used to evaluate the effectiveness and compare our approach with several others. For the Paris dataset, we also presented how we collected a set of sketches for the evaluation of our method, and for the Paris and ImageNet dataset, we presented a detailed description of how the ground-truths for effectiveness evaluation were built.

This chapter also presented an analysis of the Paris and ImageNet dataset according to their statistical low level features used to index them, as well as, presented some statistical analysis of the indexes. This analysis was important to guide the development of the approach and evaluate the homogeneity of the dataset and its ground-truths. Although the analysis of the two datasets presents some differences, we observed that in both, the distribution of contours are similar. For example, in both, the vertical and horizontal contours are predominant, as well as, they are more dense on edges in the middle of the images.

Finally, in the end of this chapter, we presented the concepts of precision and recall, as well as, average precision and mean average precision and how to estimate them. These measures are important on the evaluation of our approach experiments shown in next chapter.

Chapter 6

Experimental Evaluation

In this chapter, we present the experiments performed with our approach. First, we employ a relatively small image dataset to make the experiments and select the appropriate parameters for the content representation and for the similarity measure between the sketches and indexed images of the dataset. In our experiments, we compare both, effectiveness and efficiency of our proposal with Mind-Finder [Cao et al., 2011], on a dataset with more than 535K images issued from ImageNet. We use our own implementation of Mind-Finder, once that such code was not available. Our Mind-Finder implementation, as well as the Sketch-Finder is coded in C++ and uses the OpenCV[1] library [Bradski and Kaehler, 2008; Dawson-Howe, 2014].

We also present an evaluation of the impact of each parameter of our approach using the 2^k factorial analysis model. This analysis can show us the most important parameters to set, aiming to achieve the best optimization. For the optimization of the approach with parameters selection, we present a description of an evolutionary approach. This evolutionary algorithm automatically performed more than 5,000 indexing experiments and more than 550,000 queries on the Paris dataset.

Finally, we present an effectiveness comparison of our approach and several other methods based on Bag of Word using the *Flickr15K* dataset.

6.1 Experiments on the Paris Dataset

The experiments with the Paris dataset used Sketch-Finder 1.0, *i.e.*, the similarity measure used is the one presented in Equation 4.2, where number of coefficients to represent the neighborhood edgel map is always constant. The experiments were per-

[1]OpenCV – `http://http://opencv.org/`

formed with contours extracted from natural images (natural contours), as well as, real sketches drawn by several users, as described in Section 5.3.

The experiments on the Paris dataset were mostly employed to evaluate the similarity measure and improve the techniques that should be used in our method or discarded. The Paris dataset was also used to evaluate the importance of each parameter with the 2^k factorial analysis model, and to evaluate the effectiveness of our approach. The method was evaluated with real sketches for the Paris dataset presented in Section 5.3.

6.1.1 Evaluation of the method techniques

In order to develop a robust and efficient approach, several implementations and tecniques were evaluated until becoming the final version of our sketch-based image retrieval method. In this section, we describe the most important evaluations done on Sketch-Finder 1.0.

Some parameters have to be well chosen to have a good precision×recall curve on Sketch-Finder 1.0. Aiming to achieve this goal, we evaluated: (i) the neighborhood edgel map radius size r; (ii) the number of wedgels, *i.e.*, the number of most significant coefficients to encode the image contours; (iii) the viability of combining more than one version of neighborhood edgel map; and (iv) the impact evaluation for the use of different weighting functions. In this section, we show the results of these comparisons using the Paris dataset. For these experiments, we use our Paris sketch dataset described in Section 5.3 that contains 132 sketches, from which, we selected 110 samples.

In the following experiments, we consider the best rank precision. In this case, the first 20 classified images are used as a parameter for measuring the quality of queries. The evaluation takes into consideration the first 5 positions P_5, the first 10 positions P_{10}, and so on, until the 20th position P_{20}. The reason of choosing the z first positions for the evaluation of the parametrization is that we consider the best method must bring the best precision in the first z positions. Further, this is the measure used in some works similar to ours, like the work presented in [Sun et al., 2013].

The experimental setup of the following experiments, on the Paris dataset, is given by the variables presented on Section 4.1.2 for the similarity measure of Sketch-Finder 1.0. Where, $N_{\mathcal{G}}$ represents the number of neighborhood edgel map $\{1, 2 \text{ or } 3\}$ respectively with radius r_1, r_2, r_3, given in pixels; N_{Θ} the number orientations (always six); α the weighting function; UCM_t the threshold for the dataset image contours in the interval $[0, ..., 1]$; and N_w the number of coefficients to encode each natural image

Table 6.1. Experiments with a single neighborhood edgel map ranging its radius in pixels. Each column represents the radius size ranging from 2 to 45 pixels, and each line presents the average precision in percentage until the image ranking position P_z.

Pos	**2**	**5**	**10**	**15**	**20**	**25**	**30**	**35**	**40**	**45**
P_5	64	68	69	70	70	71	**72**	71	68	66
P_{10}	44	50	55	55	57	**58**	57	54	55	56
P_{15}	36	42	49	48	**51**	**51**	**51**	48	49	48
P_{20}	32	38	44	44	**47**	45	46	45	44	44

contour.

Evaluation of Neighborhood Edgel Map Radius Size

For single neighborhood edgel map, we experimented different radius sizes to find the best configuration. Some query sketches present better precision with smaller radius, while other queries work better with larger ones. This experiment used the following setup: $N_{\mathcal{G}} = 1$, $N_{\Theta} = 6$, $\alpha = 0$, $UCM_t = 0.27$ and $N_w = 120$. We varied r_1 between 2 and 45 pixels. Table 6.1 shows experimental results in average precision.

This experiment demonstrates that in average, the best neighborhood radius size lies around 20 and 30 pixels. The precision decreases for radius outside this range.

Varying the number of Wedgels

This experiment aims to evaluate the ideal number of large magnitude coefficients to represent each neighborhood edgel map in the wavelet domain. The number of coefficients directly impacts on the level of detail in the compressed representation of the neighborhood edgel map. In lossy image compression techniques, like JPEG2000 [Christopoulos et al., 2000], it is possible to choose the size of the image file accepting a lower quality. In JPEG2000, the image compression and its quality is basically determined by the number of large magnitude wavelet coefficients selected. In a similar way, the representation of the neighborhood edgel map approximation is determined by the number of *wedgels*, which is also based on large magnitude wavelet coefficient. When only a few number of coefficients are used, the contour signature does not represent the image edges accurately, by the other hand, using a significant number, can make the index too large and maybe redundant. To discover the best number of coefficients, we experimented a variation between 120 and 360 elements per contour signature, *i.e.*, each

Table 6.2. Experiments ranging the number of wavelet coefficients per image. Each column represents the number of coefficients per image ranging from 120 to 360. Each neighborhood edgel map of the image contour has 1/6 of the edgel. Half part for positive and half for negative coefficients. In this table, each line presents the average precision in percentage until the image ranking position P_z.

Pos	**120**	**144**	**168**	**192**	**216**	**240**	**264**	**288**	**312**	**336**	**360**
P_5	**74**	70	72	70	72	70	71	72	71	71	71
P_{10}	56	56	57	57	57	57	**58**	56	57	57	57
P_{15}	48	48	49	50	50	**51**	50	49	51	49	49
P_{20}	44	44	44	46	44	**47**	45	45	45	45	45

oriented neighborhood edgel map is represented by 1/6 of the coefficients, where half part are negative and half positive. For example, for 120 coefficients, each neighborhood edgel map has 10 positive quantized coefficients and 10 negative ones. Table 6.2 shows the best precision results using the same best rank evaluation of Table 6.1. This experiment used the following setup: $N_{\mathcal{G}} = 1$, $r_1 = 30$, $N_{\Theta} = 6$, $UCM_t = 0.27$ and $\alpha = 0$.

Although using more large magnitude coefficients increases the precision on the first 20 ranking positions, the differences are not very significant, and in the same queries, the use of more than 264 coefficients per images even decreases the precision. Further, the variation impact on the number of large magnitude wavelet coefficients is not very significant, as shown in the 2^k factorial analysis, presented in Section 6.2.1. In this experiment, the best number of coefficients lies between 240 and 264, although only 120 presents the best configuration for the first five images.

Using Simultaneous Versions of Neighborhood Edgel Map

As shown in the experiments evaluating the radius size for the neighborhood edgel map (see Table 6.1), the best precision was obtained with radius size between 20 and 30 pixels ($r = [20, ..., 30]$), in average of the 110 queries. However, we observed that some specific queries worked better with a smaller radius while other with bigger sizes. This observation raised the supposition that combination of more than one version of neighborhood edgel map, with different radius size could improve the precision of the approach.

In this way, we built experiments combining two and three neighborhood edgel maps with $r = [10, ..., 45]$ pixels, and the results are presented in Table 6.3. This experiment used 120 coefficients per radius size, *i.e.*, 240 coefficients for two radius per indexed image, and 360 using three radius. The other parameters are: $N_{\mathcal{G}} = 2$ or

Table 6.3. Experiments with multiple neighborhood edgel maps. The numbers of r in columns represent the radius size of each neighborhood edgel map. In this table, each line presents the average precision in percentage until the image ranking position P_z.

Pos	$r_{10,20}$	$r_{10,30}$	$r_{10,40}$	$r_{15,30}$	$r_{20,25,30}$	$r_{10,20,30}$	$r_{10,25,40}$	$r_{15,30,45}$
P_5	71	72	75	74	74	76	**79**	74
P_{10}	58	59	61	60	58	58	**62**	61
P_{15}	51	52	53	53	49	53	**54**	53
P_{20}	46	47	**49**	48	44	**49**	**49**	48

3, N_Θ = 6, α = 0, and UCM_t = 0.27. In Table 6.3, the first row presents the sizes of r for each neighborhood edgel map and, each line, presents the average precision in percentage until the image ranking position P_z.

Comparing the results of a single neighborhood edgel map (shown in Table 6.1) with these experiments, we observed that using multiple neighborhood edgel map versions improves the precision of the method. The most important conclusion is not the exact size of the radius for each neighborhood edgel map, but, the observation that using two radius is better than one and, worst than three, for the precision of the method.

Experimenting the Best Weight function for the Similarity Measure

The initial idea of Sketch-Finder was to use the coefficients weights inherited from the work of Jacobs et al. [1995] that are given by *bin* function presented in Equation 3.5. This function is based on the coefficient position and the weights possibly returned by *bin* function are shown in Table 3.2. Although this was a good start, even better than considering the weight always a constant, one for example, some results were not good, specially when some contour orientation was very dense in edgels. This observation leaded us to associate the edgels density to the weight of the coefficient, once that in Sketch-Finder 1.0 the number of coefficients that represent each edgel map is constant and do not depends on the edgels density.

This experiment evaluates both weights and in a linear interpolation among them. The interpolation is controlled by α ranges from 0, when we consider exclusively the weight associated to edgels density ($\mathcal{X}$), and 1, when consider only *bin* function in Equation 4.2. Intermediate levels among the weights are evaluated in the interval [0,...,1]. This experiment used the following setup: $N_\mathcal{G}$ = 1, r_1 = 30, N_Θ = 6, UCM_t =

Table 6.4. Experimenting the best weight for the similarity measure ranging α. Each column represents the α value ranging from 0 to 1, and each line presents the average precision in percentage until the image ranking position P_z.

Pos	$\alpha = 0$	$\alpha = 0.2$	$\alpha = 0.4$	$\alpha = 0.6$	$\alpha = 0.8$	$\alpha = 1$
P_5	**72**	68	68	67	66	66
P_{10}	**57**	53	52	51	50	50
P_{15}	**51**	48	46	45	45	45
P_{20}	**46**	43	42	42	41	41

0.27 and $N_w = 120$.

This experiment shows clearly that the weights based on the density of edgels on each quantized oriented edgel map presents a better precision rather than *bin* function, based on the coefficient position. It means that *bin* function is not important in our approach and can be eliminated in Equation 4.2. Also, guided by the results of this experiment, we developed a similarity measure for Sketch-Finder 2.0 (see Section4.2) based on a variable number of coefficients for each neighborhood edgel map, where the number of coefficients is related to the edgel density, thus simplifying the measure without considering these weights. Details for this approach are presented in Section 4.2.

Changing Wavelet by DCT

Aiming to experiment if the wavelet is the right choice to represent the images contours in the compressed-domain, we also experimented Discrete-Cosine Transform (DCT) as the compressed descriptor. The DCT is successfully used for JPEG compression format [Ansari and Memon, 2000]. However, here, instead of computing the DCT on image samples of size 8×8 pixels, as it is performed in JPEG format, we applied the DCT to the entire image with resolution of 256×256 pixels, as we did for wavelet analysis.

For this experiment, we followed the exact Sketch-Finder 1.0 same algorithm with the similarity measure presented in Equation 4.2. The only difference was the changing of wavelet coefficient by the DCT one. All the other assumptions made for Sketch-Finder with wavelet transform are rigorously followed by the DCT approach, like the use of the most significant positive and negative coefficients to encode the contours..

In this experiment, we compared the wavelet version and the DCT using the following setup for both: $N_{\mathcal{G}} = 1$, $r_1 = 20$, $N_\Theta = 6$, $UCM_t = 0.16$, $\alpha = 0$ and $N_w = 120$.

Table 6.5. Comparison of the compressed-domain feature (Wavelet *vs.* DCT). The presented value corresponds to MAP.

	Sketch-Finder 1.0
Wavelet	**0.073**
DCT	0.053

In this experiment, we abstracted the effectiveness of both approaches in a single value, the Mean Average Precision (MAP). The results obtained are presented in Table 6.5.

As shown, the results using the DCT presented a lower effectiveness comparing to the wavelet one. We believe the reason for the DCT poor result is the absence of spatial information on the DCT coefficient, rather than wavelet feature that preservers it.

6.2 Evaluation on the ImageNet Subset

To evaluate the ImageNet subset, we used the ground-truth presented in Section 5.4. Among the large number of object categories and subcategories, 15 were selected, on which the objects were most homogeneous in terms of position, rotation and scale. Within these 15 categories, 10 were selected for the vehicle subset (*dump-cart, automobile, scooter, serving-cart, velocipede, airplane, helicopter, ship, motorcycle* and *ambulance*) and five for the building subset (*governmental building, house, rotunda, oast house* and *opera house*). Like the evaluation of the Paris dataset, we considered the occurrence of the object in the subset label, even knowing that the subset has the same kind of object in different positions, scale and rotations.

Although the proposed SBIR approach is built for sketches, the difficulties of having a large number of different sketches encouraged us to use also the extracted contours from the images of the ImageNet dataset. Also, natural contours obviously better describe the shape of objects, once that the user, usually, does not know very well how to draw good sketches and lacks on perspective and scale drawing of objects. The experiments with natural contours are also important because the user can draw a sketch to make a first query, and then select one similar image, or true positive on the result, and remake the query using this image. This new query can extract the natural image contour and perform a new query with the very same method, improving the results over the first one. Finally, ImageNet, as our biggest dataset, was mostly used

Table 6.6. Best rank precision comparing Sketch-Finder 1.0 and Mind-Finder. Each line presents the average precision in percentage until the image ranking position P_z.

	Sketch-Finder 1.0	**Mind-Finder**
P_5	**59**	**59**
P_{10}	48	**49**
P_{15}	42	**43**
P_{20}	40	**41**

to evaluate efficiency rather than effectiveness. Thus, the nature of the input, being a sketch or a natural image contour, does not bring impact on efficiency evaluation.

The experiments with natural contours used five samples of each category that we defined. As such, we have 15 categories times 5 images contours, that sum a total of 75 queries. The list of query images contours is presented in Table 5.3. For each query, we evaluated the precision×recall curve, as well as, the average precision of all queries. This experiment used the following setup for Sketch-Finder 1.0: $N_{\mathcal{G}} = 3$, $r_1 = 10$, $r_2 = 25$, $r_3 = 40$, $N_{\Theta} = 6$, $\alpha = 0$, $UCM_t = 0.27$ and $N_w = 360$. For Mind-Finder, the experimental setup was: $r_1 = 45$, $N_{\Theta} = 6$, and $UCM_t = 0.27$.

To compare our approach with Mind-Finder [Cao et al., 2011], we considered the first 20 results, like we did for the experiments in the Paris dataset. In these experiments, nine, of fifteen categories, were best classified by our approach.

Table 6.6 shows the average precision in percentage until the image ranking position P_z of all 75 queries, for both approaches.

Figure 6.1 presents the average precision×recall curve for the first fifty ranked images. For each one of the fifteen object categories, we averaged the five respective queries. Also, blue line represents Mind-Finder while the red dashed line represents Sketch-Finder 1.0.

Figure 6.2 presents the average precision×recall curve for the best fifty results summarizing all 75 queries. The blue line represents Mind-Finder while the red dashed line represents Sketch-Finder 1.0.

According to the results, we consider that both approaches, Sketch-Finder 1.0 and Mind-Finder, are equivalent in their effectiveness in a large dataset using natural contours as input.

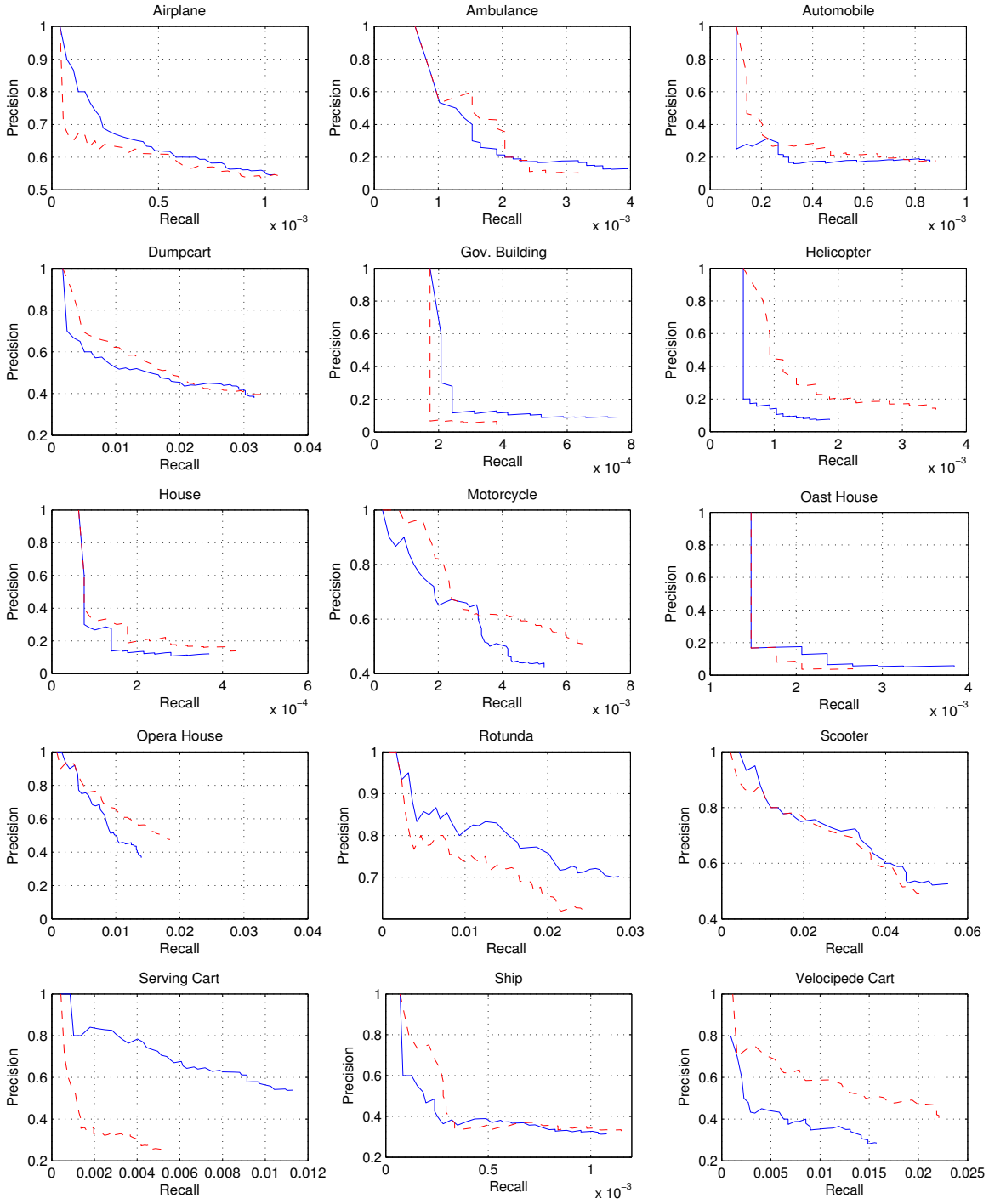

Figure 6.1. Average Precision×Recall curves by object of Sketch-Finder 1.0 and Mind-Finder approach on the ImageNet subset (535K). Each curve represents the average Precision×Recall curve of five queries. The continuous blue line represents the curve of Mind-Finder approach, while the red dashed line represents Sketch-Finder 1.0. Both curves are representation of the first fifty ranked images.

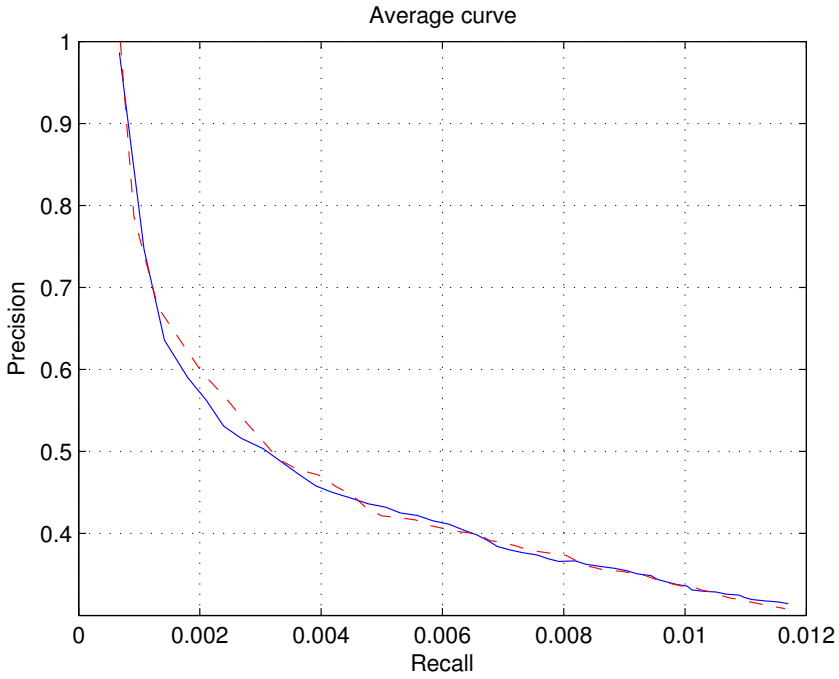

Figure 6.2. Average Precision×Recall curves from 75 queries of the Sketch-Finder 1.0 and Mind-Finder approach on the ImageNet subset (535K). The continuous blue line represents the curve of Mind-Finder approach while the red dashed line represents Sketch-Finder 1.0. Both curves are representation of the first fifty ranked images.

Efficiency evaluation using the ImageNet dataset

In order compare Sketch-Finder 1.0, 2.0 and Mind-Finder, we evaluated the CPU query time, in seconds, and I/O reading, in bytes. To evaluate the CPU time and I/O bytes of the queries, we used the ImageNet dataset and the same 75 queries used to evaluate the best rank precision. The experiments with Sketch-Finder 1.0, 2.0 and Mind-Finder approaches were evaluated in a machine with 24 CPU cores and 72Gb of RAM memory, without other user processes running at the same time. Also, both approaches used a single thread for their queries, *i.e.*, only one processor was used in these experiments.

In order to estimate the CPU cost of both approaches, we do not consider the time for I/O reading of inverted files, *i.e.*, the time was not computed before an I/O request and immediately restarted after the file loading. This interruption of time during the I/O emulates the system like if it had the data in main memory, however, the I/O cost was not discarded and this measurement, in bytes, is presented in Table 6.8, with its average value (AVG) and standard deviation (SD) for the 75 queries. The CPU cost is presented in seconds in Table 6.7, also with its average and standard deviation for

Table 6.7. Average CPU query time estimated in seconds for 75 queries using the ImageNet dataset. This table presents the average time (CPU AVG), in seconds, and the standard deviation (CPU SD).

Approach	CPU AVG	CPU SD
Sketch-Finder 1.0	**6.56**	**1.13**
Sketch-Finder 2.0	31.66	11.73
Mind-Finder	394.43	401.57

Table 6.8. Average I/O estimated in bytes for 75 queries using the ImageNet dataset. This table presents the average (I/O AVG) data reading, in bytes, and the standard deviation (I/O SD).

Approach	I/O AVG	I/O SD
Sketch-Finder 1.0	$\mathbf{2.89\times10^8}$	$\mathbf{1.58\times10^7}$
Sketch-Finder 2.0	1.52×10^9	4.08×10^8
Mind-Finder	2.96×10^9	2.79×10^8

75 queries.

All approaches were implemented using inverted files lists of images ID's and the indexes are stored on disk. Although the original implementation of Mind-Finder is designed for main memory, we used disk implementation in order to have the same criteria of evaluation. Further, on disk, it is possible to have a scalable approach in the size of the dataset, that can grows. As shown in Table 6.7, the average CPU time of the queries in Sketch-Finder 1.0 is more than 47 times faster. This occurs because Sketch-Finder 1.0 has less data to process. Additionally, the standard deviation is smaller in Sketch-Finder 1.0 due to the fixed number of inverted lists to process, what does not occur in Sketch-Finder 20 and Mind-Finder. This is an advantage for the Sketch-Finder 1.0 approach, once that we can have an expectation of the query time without very much variation. As expected, the addition of OCM in the algorithm of Sketch-Finder 2.0 increases the query time, however, it is still under Mind-Finder.

It is important to detach that query time of our Mind-Finder implementation is higher than its original because we perform the second similarity comparison of OCM to all images of the dataset, while the original implementation compares only the 0.0025% best classified images in the first OCM similarity comparison. We decided to compare all the images in order to have a precise evaluation of effectiveness.

Table 6.9. Index size in bytes.

Approach	Index size
Sketch-Finder 1.0	**2.14×10^9**
Sketch-Finder 2.0	2.05×10^{11}
Mind-Finder	2.61×10^{10}

The index size of Sketch-Finder is the biggest among the evaluated approaches due to the storage of the preprocessed neighborhood edgel maps used in the OCM similarity comparison, however, as advantage, this index makes possible a faster query than the approach of Cao et al. [2011], see Table 6.7. The total index size of each approach, measured in bytes, is given in Table 6.9.

6.2.1 Parameter Relevance Evaluation on Sketch-Finder 2.0

This section presents an evaluation of the parameters relevance of Sketch-Finder 2.0. Discovering the most relevant parameters allows us tuning the best configuration of Sketch-Finder 2.0 with focus to important variables. For this analysis, we also used the Paris dataset with 110 sketches as in the experiments of Section 6.1.

Aiming to evaluate the relevance of the parameters of our approach, we applied the 2^k Factorial Design model presented in Jain [2008]. We have six parameters or factors and, as in the 2^k Factorial Design each factor of k has two alternative levels, the higher and lower values of the factor, therefore, our analysis has 2^6, or 64 possibilities configuration of parameter combinations.

The parameters and the effect of each one are described in the following:

a) neighborhood edgel radius for the first wavelet transform: this parameter corresponds to the first neighborhood edgel map $\mathcal{G}_\theta^{r_1}$ radius size used before the wavelet transform;

b) neighborhood edgel radius for the second wavelet transform: this parameter corresponds to the second $\mathcal{G}_\theta^{r_2}$ radius size used before the wavelet transform;

c) neighborhood edgel radius for the third wavelet transform: this parameter corresponds to the third $\mathcal{G}_\theta^{r_3}$ radius size used before the wavelet transform;

d) neighborhood edgel radius for pixel matching (OCM): this parameter corresponds to the radius $r_{_{OCM}}$ used in the OCM. In our approach, 0 means to not use the pixel matching, *i.e.*, only measure the wavelet similarity ($\mathcal{W}_{\mathcal{Q},\mathcal{T}}$);

e) wavelet coefficients threshold: this parameter corresponds to the wavelet coefficient threshold ω that determines what coefficients are used to represent each

Table 6.10. 2^k Factorial Design Model. In this table, Factor is the parameter of our approach that we measure the impact, L represents the lowest value of the parameter, H the highest value and R its respective relevance in percentage.

Factor	**L**	**H**	**R**
a) Edgel map 1	1	10	1.8%
b) Edgel map 2	10	25	18%
c) Edgel map 3	25	45	1.5%
d) Edgel map (OCM)	0	45	47.5%
e) Coefficient Threshold	3.0	8.0	3.8%
f) Contour Threshold	0.15	0.30	27.4%

neighborhood edgel map in the compressed-domain of wavelet and which coefficients are discarded. The coefficient is selected if its absolute value is greater than ω.

f) threshold of the image contours: this parameter is used for thresholding the UCM image for indexing.

Table 6.10 presents the relevance (R) of each factor in percentage, as well as, the lowest (L) and the highest (H) values used to test each factor.

The two most important parameters in our set are the threshold of the image contours and the radius size of the OCM similarity comparison, both summing almost 75% of impact in the precision. The variation of second neighborhood edge map radius presents almost 18% of impact, while the variation of the second and the third radius maps, and the variation of the wavelet threshold are almost insignificant.

The main difference between Sketch-Finder 2.0 over Sketch-Finder 1.0 is the addition of the Oriented Chamfer Matching for comparing sketch and image contours. The effect of this parameter, presented by the 2^k Factorial Design, indicates an important improvement on the precision.

6.2.2 Parameter Tuning With Genetic Algorithm

The parameters described in Section 6.2.1 need to be well set for obtaining the best performance of our approach. Genetic algorithms are robust search and optimization techniques for finding the global optimum in a multimodal landscape. Therefore, we present in this section, how the parameters of our approach were chosen using a genetic evolution approach [Srinivas and Patnaik, 1994]. We chose genetic evolution mainly for two reasons: first, genetic evolution is an efficient way to converge the parametrization

to the global or to some local high mean average precision; and second, the algorithm is relatively easy and quick to write comparing to other optimization approaches.

To set the parameters, due to the large number of experiments, we used a small dataset, the Paris dataset with a ground-truth for the sketches that we collected. The parameters and their intervals were chosen as described in Section 6.2.1. We started the genetic algorithm with a population of one hundred random different configurations, where each parameter had a random value between the lowest and highest limits (L and H) presented in Table 6.10. On each "evolution" iteration, the best fitness solution was preserved for the next iteration or "generation". The other fitness solutions above the average were preserved for crossover and mutation, while solution fitness sub averaged were disrupted. Random best solutions in pairs were used to generate two new solutions with crossover and mutation of parameters. For each hight fitness pair, with a probability P_c of 90%, we applied a crossover into three of the six random parameters, and with a probability P_m of 10% we applied a mutation into two random parameters inside the limits. Table 6.10 presents the lowest and highest limits (L and H) used in the mutation of the parameter.

To represent and evaluate each individual parameter configuration in a single fitness value, we used the Mean Average Precision (MAP) obtained from the precision×recall curve [Davis and Goadrich, 2006].

We conducted extensive experiments for achieving the best rank of the proposed approach. More than 50 genetic generation iterations, each one with 100 individual set of parameters were performed, what gave more than 5,000 experiments. Each experiment built one index solution and performed on it, 110 queries by sketch, *i.e.*, a total of 550,000 queries.

6.2.3 Evaluation of Sketch Retrieval Effectiveness

The experiments on sketch retrieval used the best parameters obtained by the genetic algorithm in average of 110 queries. For Sketch-Finder 2.0, this configuration is respectively **9, 15, 28, 6.0, 44** and *0.16* for the parameters *a*, *b*, *c*, *d*, *e* and *f*, described in Section 6.2.1. For Sketch-Finder 1.0, we used the same parameters in common with Sketch-Finder 2.0. This configuration is respectively **9, 15, 28, 44** and *0.16* for the parameters: *a*, *b*, *c*, *e* and *f* presented in Section 6.2.1.

Regarding to the evaluation of the Mind-Finder [Cao et al., 2011], we applied the same parameters to the steps in common with Sketch-Finder 1.0/2.0, in other words, image resolution of 256×256, same contour detection (UCM) and threshold = 0.16. For the radius r we experimented several configurations, between 25 and 65 pixels,

finding that r = 45 brings the best fitness on 110 queries of the Paris dataset, using the same criteria for the fitness as in our approach. The selected parameters were used as default for the Mind-Finder in the comparisons with Sketch-Finder 1.0/2.0. The experiments were realized in a machine with CPU Intel Xeon X5670 with 2.93GHz and 72Gb of RAM memory. For the effectiveness evaluation, we considered the 50 first ranked images. According to the results, Sketch-Finder 2.0 overcame Sketch-Finder 1.0 and Mind-Finder (MF) in terms of effectiveness as shown by the precision curve of the 50 first ranked images presented in Figure 6.3.

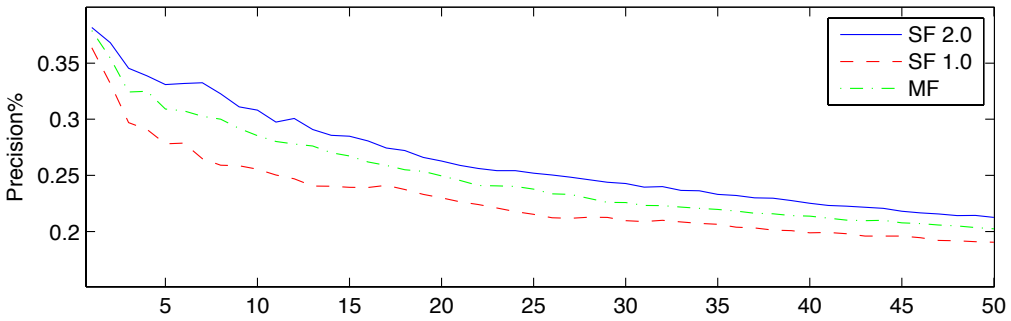

Figure 6.3. First 50 ranked images.

Figure 6.4 shows some queries for the Paris dataset using Sketch-Finder 2.0. The first image on each line is the query image and the following images are the five first results.

6.2.4 Evaluation of the Similarity Measure for Sketch-Finder 2.0

During the development of Sketch-Finder 2.0, we experimented several similarity measures, in this section we compare the similarity that we proposed in Equation 4.7 with two classical models used for general information retrieval, the "*term frequency inverse document frequency*" (*tf-idf*) weighting [Salton and McGill, 1986; Sivic and Zisserman, 2003; Robertson, 2004] and *Okapi BM25* [Robertson et al., 2004].

Term Frequency-Inverse Document Frequency

Term frequency - inverse term frequency is the product of two statistical measures. Inside *tf-idf*, the "term frequency" (tf) part, captures the relevance of the word inside a document d. In our case, the word is the wedgel while the document is obviously an image.

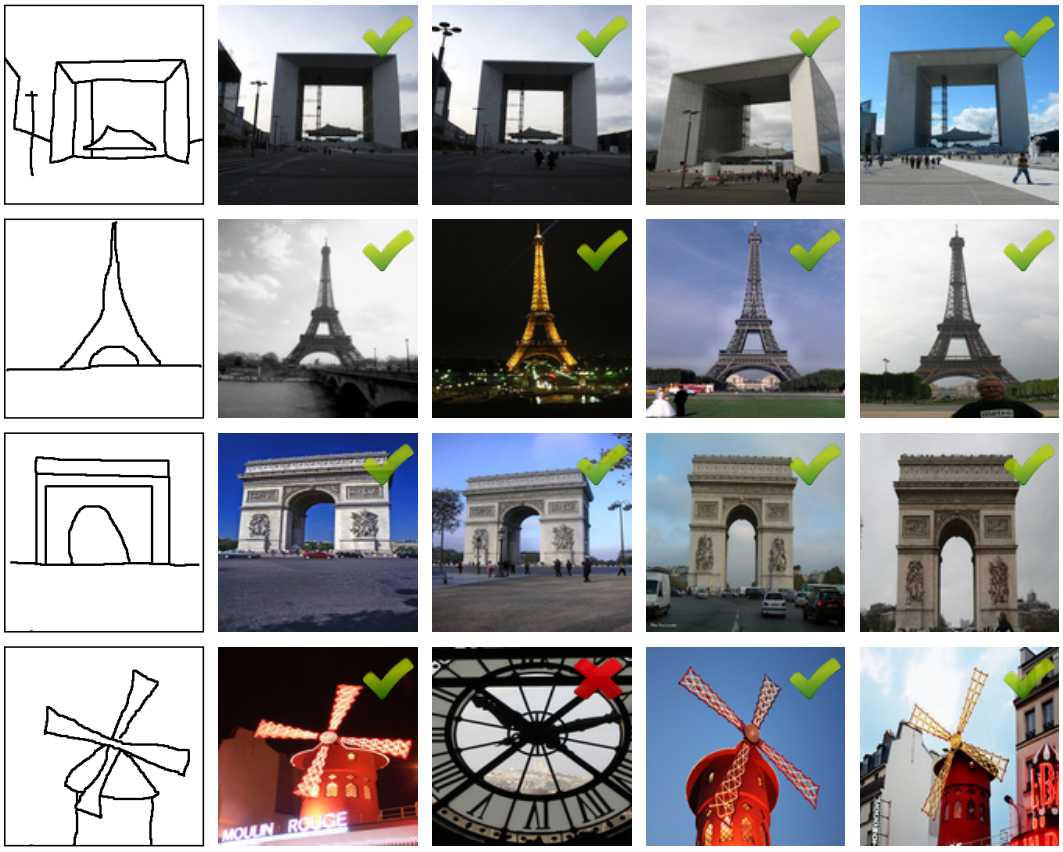

Figure 6.4. Sketch-Finder 2.0 query result examples on the Paris dataset - *La Defense; Tour Eiffel; Arc de Triomphe;* and *Moulin Rouge*. It is shown in this figure the first five results of each query sketch.

The *idf* part captures the informativeness of visual words. According to the mathematical theory of communication [Shannon, 1948], the quantity of information of a random variable is inversely proportional to its probability of occurrence. Thus, visual words that appear in many different images are less informative than those that appear rarely.

Given a set of visual words, $\mathcal{A} = \{w_1, w_2, ..., w_n\}$, the tf and idf are respectively computed by Equations 6.1 and 6.2.

$$tf(w_i) = \frac{n_{id}}{n_d} \tag{6.1}$$

Where n_{id} is the number of occurrences of a word i in the document d, and n_d is the total number of words in the document d. For boolean frequencies, this is our case, $tf(w_i) = 1$ if the word exists in a document d, and 0 otherwise. The idf is presented on the following.

$$idf(w_i) = \log \frac{N}{n_i} \tag{6.2}$$

Where n_i is the number of occurrences of the word i in the whole dataset and N is the number of documents in the whole dataset, or the number of images.

Finally, *tf-idf* is the product of the two terms, *tf* and *idf*, shown in Equation 6.3.

$$tfidf(w_i) = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \tag{6.3}$$

Okapi BM25

In information retrieval, *Okapi BM25* is a ranking function used by search engines to rank text documents according to their relevance. Part of the *BM25* weight is based in the previous presented *idf*. *Okapi BM25* is not a single function, but actually a family of scoring function, with slightly different components and parameters. We present in Equation 6.4, a *BM25* function which is considered to be one of the most prominent.

$$BM25(w_i) = idf(w_i) \cdot \frac{n_{id} \cdot (k_1 + 1)}{n_{id} + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \tag{6.4}$$

where n_{id} is the number of word occurrences i in a document d, $|\, d \,|$ is the number of words in the document d, $avgdl$ is the average number of words per documents, and k_1 and b are free parameters, by default, $k_1 \in [1.2, 2.0]$ and $b = 0.75$.

Comparison of Ranking Functions

Once that in our approach, the number of word occurrences i in a document d (n_{id}) is always binary, we discard this information because every *hit* of wavelet coefficient is also an occurrence of a word, what is not discriminating. Thus, we rewrote the *BM25* as *BM25A*, as presented in Equation 6.5:

$$BM25A(w_i) = idf(w_i) \cdot \frac{(k_1 + 1)}{k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \tag{6.5}$$

We compared our weighting function presented in Equation 4.7 with "*term frequency inverse document frequency*" (*tf-idf*) and *Okapi BM25A* weighting functions. For this comparison, we followed the same parameter set of Sketch-Finder 2.0, changing only the weighting function. We used the Paris dataset with 110 sketch queries, as in the experiments presented in Section 6.1.1. Sketch-Finder 2.0 was configured with values **9, 15, 28, 6.0, 44** and *0.16*, which are respectively the parameters *a*, *b*, *c*, *d*, *e* and *f*, described in Section 6.2.1.

Table 6.11. Weighting function comparison for Sketch-Finder 2.0.

Ranking	MAP
Ours	0.103
tf-idf	0.080
BM25A	0.097

Table 6.12. Changing the variable b in BM25B ranging function.

b	0.0	0.2	0.4	0.6	0.8	1.0
MAP	0.085	0.093	0.099	0.102	**0.104**	**0.104**

In those experiments, (*tf-idf*) and *BM25A* was used as weighting function for the *Hit* function presented in Equation 4.6.

For the experiments with *BM25A* we used $k_1 = 1.0$ and $b = 0.75$. Table 6.11 presents the comparison result of the three weighting functions. Ours, *tf-idf* and BM25A. The results are presented in Mean Average Precision (MAP).

As shown in Table 6.11, the weighting function based on *tf-idf* presented a low MAP while *BM25A* was almost as good as our proposed metric. This result suggested a new experiments on *BM25A*, and our idea was to remove the *idf* from *BM25A*. Thus, this new equation, named *BM25B* can be written as:

$$BM25B(w_i) = \frac{(k_1 + 1)}{k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \tag{6.6}$$

The removal of *tf-idf* improved the MAP from 0.097 to 0.103, a competitive value to our weighting function.

In order to evaluate the variable b the Equation 6.6, we ranged it in the interval $[0, ..., 1]$, while keeping $k_1 = 1$. The results are presented in Table 6.12, also in MAP.

According to the results presented in Table 6.12, the best value for b is 1. For $b = 1$ and $k_1 = 1$, *BM25B* can be rewritten as:

$$BM25C(w_i) = \frac{2}{\frac{|d|}{avgdl}} \tag{6.7}$$

Once the dividend of Equation 6.7 is a constant 2, changing the constant to 1 does not change the weighting function. Thus the function can be rewritten as:

Table 6.13. Comparison of several SBIR approaches using *Flickr15K* dataset.

Approach	MAP
Sketch-Finder 2.0	0.0871
Mind-Finder	0.0851
GF-HOG	0.1222
HOG	0.1093
SIFT	0.0911
SSIM	0.0957
ShapeContext	0.0814
StructureTensor	0.0798

$$BM25C(w_i) = \frac{avgdl}{|\,d\,|} \tag{6.8}$$

According to the results presented in Table 6.12, the weighting function of Equation 6.8 can be used as an alternative to the one that we propose in Equation 4.7. Equation 6.8 is more simple and computationally faster than Equation 4.7.

6.2.5 Comparison With Other SBIR Approaches

Aiming to compare our approach with others, based on the BoVW, we used the *Flickr15K* dataset presented in [Hu and Collomosse, 2013]. The local descriptors for the BoVW that we compared are: GF-HOG [Hu and Collomosse, 2013], HOG [Dalal and Triggs, 2005], SIFT [Lowe, 2004], SIMM [Shechtman and Irani, 2007], ShapeContext [Belongie et al., 2002] and StructureTensor [Eitz et al., 2009]. The experiments used the same image dataset, sketches and methodology of [Hu and Collomosse, 2013].

Although these approaches are designed for SBIR variant to scale and position, and moreover, they make use of the BoVW techniques, this comparison gives us an idea of our approaches and other SBIR methods. The results of Mind-Finder, Sketch-Finder 2.0 and the local descriptors using the *Flickr15K* dataset are presented in Table 6.13.

It is important to detach that our approach is sensitive to affine transforms, which is a desired characteristic when the user has chosen to query an image by sketch, exactly because he/she wants not only the desired object that has in mind, but also, at similar position and scale. Notwithstanding our approach presented a lower MAP than the SSIM, HOG and GF-HOG, the ground-truths used are designed to approaches where the image can vary its position, based on BoVW, what brought a negative impact to

our method and Mind-Finder.

6.3 Conclusion

In this chapter, we evaluated the effectiveness of our approach with Mind-Finder and several others. We also evaluated the efficiency of Sketch-Finder 2.0 and Mind-Finder, showing that our approach is more efficient than Mind-Finder.

The experiments on the Paris dataset guided the development of the presented approach improving its effectiveness. It was concluded that, even in the compressed domain, the density of edgels on each orientation map plays an important role in the weights of the similarity measure. Also, joining several indexes, built with different neighborhood edgel radius, improves the effectiveness of our approach.

Our most simple implementation of Sketch-Finder is equivalent to Mind-Finder in effectiveness with natural contours as input. This feature is important because this kind of query can be performed after a query-by-sketch. In this proposed scenario, the user selects one example image on the rank of the sketch query to use their contours as input for a second query. Further, it is important to detach that our approach is faster and feasible to run in large datasets, like the ImageNet that we experimented, or even much bigger datasets.

The number of wedgels used to encode the contours of the dataset presents a small impact on the effectiveness of our approach. This was observed by the experiments on Sketch-Finder 1.0 (varying the number of wedgels) and by the 2^k factorial analysis for Sketch-Finder 2.0, which shows that variation on the number of wedgels impacts only on 3.8% of the precision over all parameters, what corroborates this affirmation.

Further, the 2^k factorial analysis on Sketch-Finder 2.0 shows that the verification of pixel consistency on the pixel domain plays an important role on the effectiveness improvement of our approach, followed in terms of impact, by the right selection of the threshold for the natural contours of the image dataset.

In order to optimize Sketch-Finder 2.0, a large volume of experiments was performed using a genetic algorithm to find the best parameter configuration. These experiments ran during six months performing more than a half million queries in this optimization process.

The experiments of Sketch-Finder 2.0 shows that our approach outperformed Mind-Finder in effectiveness, and although Sketch-Finder 2.0 is less efficient than Sketch-Finder 1.0, the new version is still more efficient than Mind-Finder. Also, for sketch-based image retrieval, where affine transforms are considered, we believe

that our approach is the state of the art, as far as we know. For sketch-based-image retrieval without affine transforms, our approach presented quite good result among several others.

Chapter 7

Application: A Mobile App for SBIR

Mobile devices with touchscreen display are an ideal gadget to easily create sketches that can be used in SBIR. The wide spread of mobile devices and the absence of SBIR mobile application for *Android* motivated us to create a SBIR practical application using our SBIR approach for *Android* platform. As far as we know, there is no equivalent application for *Android*.

Thus, in this dissertation, we also present an application that fills this gap. In general lines, the application has an input interface where the user draws the desired sketch and another where the user visualize the results.

7.1 Sketch-Finder Mobile Application

Sketch-finder mobile application is an end user tool for image query-by-sketch. This application was implemented in *Java* for *Android* platform. The mobile application, or just *app*, is simple to use and intuitive. The query screen presents a square white canvas, where the user can draw the query sketch in black lines through the touch screen display (Figure 7.1 (a)). Besides the drawing area, we also have three edit buttons: *undo*, *erase* and *draw strokes*, respectively represented by icons, as well as, one button to clear the sketch depiction area *"Clear"*, and to submit the query *"Search"*. Once the user is satisfied with the drawn sketch, he/she clicks on search button to perform the search, this is all the user needs to do. The query image is sent in *png* [Miano, 1999; Adler et al., 2003] because this format compresses the query without loss of information. Further, this format reaches very good compression rates in this kind of binary image, that usually does not have very much details.

Figure 7.1 (b) presents the interface for result display. In this screen, the query sketch, followed by the most relevant images are shown in a slide bar, ordered by relevance in a queue. There is one area to display, in larger size, the selected thumbnail image in the slide bar.

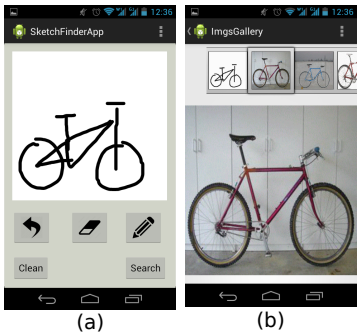

Figure 7.1. Mobile Sketch-Finder *Android* application prototype. In (a) we show the drawing interface with an area for sketch depiction and edit buttons (*undo*, *erase* and *draw strokes*), represented by icons. The drawing interface has also one button to clear the sketch depiction area *"Clear"*, and to submit the query *"Search"*. In (b), we show the query result interface, containing the query sketch, followed by the ten fist ranked images. There is also an area to display the selected result image in larger size.

7.1.1 Sketch-Finder Query Server

Sketch-Finder server accomplishes three main tasks, it stores the image dataset, their index, and hosts Sketch-Finder search engine. This engine stays waiting a connection of some Sketch-Finder *app* client. Once some connection is closed, the server receives the query sketch, preprocesses it, and performs the query mechanism. As result, the server sends back to the *app* client the ten most relevant images, and a list with the classification order of them. Preprocessing consists on two main steps: first, the binary query image is inverted, once the Sketch-Finder works with images in black back ground and white strokes; second, the strokes are thinned [Chatbri and Kameyama, 2012] to be used in the orientation algorithm that requires thinned strokes. The preprocessing steps are shown in Figure 7.2. The routines running in the server are detached inside the blue dashed line box. The steps inside "Sketch-Finder Quering" box of Figure 7.2 encapsulate the steps of Figure 4.4 where the Sketch-Finder performs the query-by-sketch.

In order to present a better efficiency and quickly send back the results for the user, the OCM is not performed, *i.e.*, the query is performed using Sketch-Finder 1.0, where we just compare wavelet coefficients as presented in Equation 4.2.

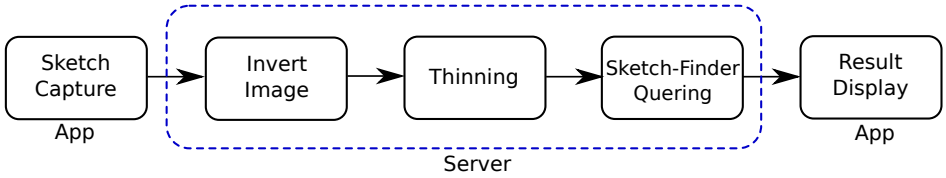

Figure 7.2. Sketch-Finder client/server querying workflow. From left to right, the first box represents the sketch capture in the mobile device, which sends the query sketch to the server using TCP/IP protocol. Inside the blue dashed line is represented the remote server routines. They are: image inversion, image thinning and Sketch-Finder querying steps. In the final, Sketch-Finder server delivers to the Android client *app* the ten most relevant images.

The setup of Sketch-Finder is given by the variables presented on Section 4.1.2 for the similarity measure of Sketch-Finder 1.0. Where, $N_{\mathcal{G}}$ represents the number of neighborhood edgel map $\{1, 2 \text{ or } 3\}$ respectively with radius r_1, r_2, r_3, given in pixels; N_{Θ} the number orientations (always six); α the weighting function; UCM_t the threshold for the dataset image contours in the interval $[0, ..., 1]$; and N_w the number of coefficients to encode each natural image contour. Thus, we have: $N_{\mathcal{G}} = 3$, $r_1 = 9$, $r_2 = 15$, $r_3 = 28$, $N_{\Theta} = 6$, $UCM_t = 0.16$, $\alpha = 0$ and $N_w = 120$.

7.1.2 The Project Infrastructure

The infrastructure of the project is shown in Figure 7.3. Sketch-Finder mobile application, either in a mobile phone or a tablet, connects to the server using an Internet TCP/IP socket connection. The mobile device must be connected to the Internet, either using 3G/4G connection, either WiFi network [Tanenbaum, 2002].

Figure 7.3 presents the infrastructure of Sketch-Finder mobile project. On the left, mobile gadgets with Sketch-Finder *app* installed, on the middle, an abstraction of the internet infrastructure used to connect the mobile devices and Sketch-Finder server. Finally, on the right, the server, hosting the dataset, the index and the search engine.

Figure 7.4 presents some query examples performed by the Sketch-Finder *app* on the *Flickr15k* dataset.

7.2 Conclusion

The mobile application for SBIR is intuitive and easy to use. This application is useful and can help users to find images of their interest. The touchscreen interface enables a good creation of sketches, and the resources available to edit the sketch, although sim-

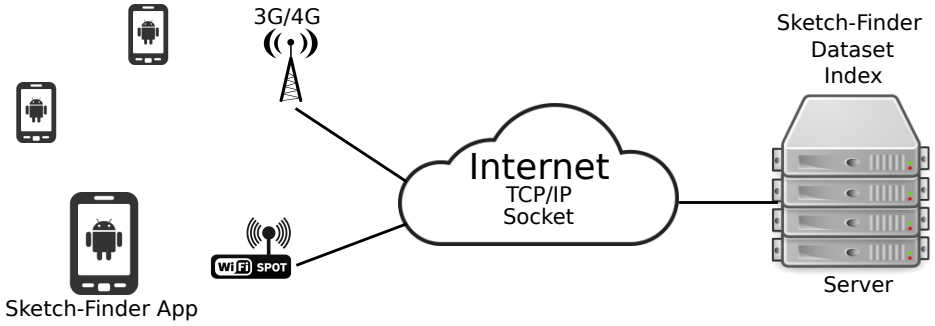

Figure 7.3. Mobile Sketch-Finder Infrastructure. On the left, mobile gadgets with Sketch-Finder application. On the middle, Internet connects Skecth-Finder mobile application to Sketch-Finder search engine using a TPC/IP socket connection. On the right, the server stores Sketch-Finder search engine, the image dataset and its index.

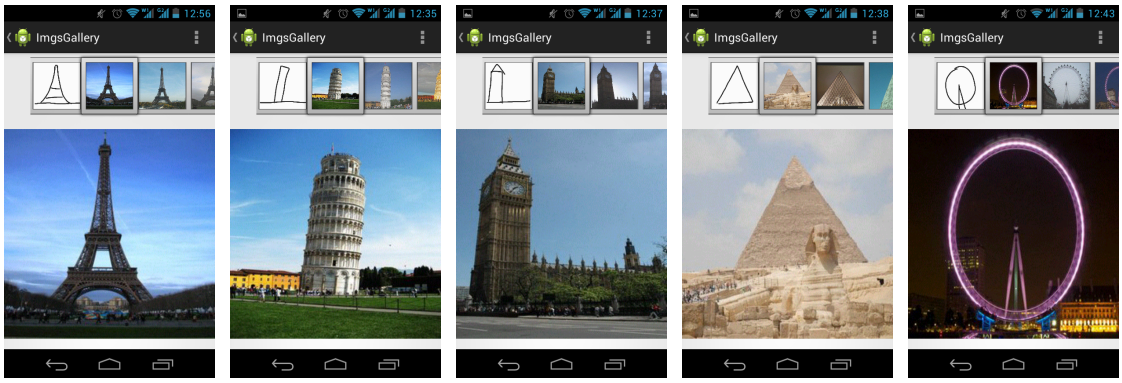

Figure 7.4. Mobile Sketch-Finder query examples.

ple, are sufficient to draw sketches for SBIR. The main difficult in this implementation was the communication protocol over TCP/IP using different languages. On the client side, the protocol was written in Java while in the server side, written in C++.

The algorithm used on the server was Sketch-Finder 1.0 because this is the most efficient approach. The query usually takes one second to complete, but, most part of the waiting time is expended on the transmission of the resulting images, especially if the user is using a mobile 2G, or even 3G, connection.

Chapter 8

Conclusion and Future Work

Building a practical sketch-based image search application is a challenging problem, either in academic or industrial communities. In this dissertation, we developed and systematically evaluated an approach for SBIR for large image dataset. The approach is designed for line-based sketches, where the dataset image contours are coded in the compressed-domain of wavelets and also in the pixel domain. The compressed-domain index allows the comparison between the sketch and the dataset image contours using just a few set of data, while the pixel domain is used to improve the precision by applying a spatial pixel consistency verification with the Oriented Chamfer Matching (OCM) algorithm. The OCM algorithm impact, reveled by the 2^k Factorial Design analysis (Section 6.2.1), showed that the improvement on effectiveness of our proposal overcomes Mind-Finder [Cao et al., 2011].

The query of our approach is faster than the one presented in Cao et al. [2011] because we use the compressed domain. Also, for the OCM, we store the neighborhood edgel maps with precomputed radius. As a drawback, our approach has a bigger index comparing to Mind-Finder, however, in a real time application, we can avoid this strategy and compute the radius of the edgel maps only for the best ranked images revealed by the wavelet similarity. Thus, improving efficiency and reducing memory, while preserving effectiveness.

Another advantage of our approach is that we can control the size of the compressed-domain index (CDI), just like we choose the size of an image controlling its compression rate. The low impact on the number of wedgels used to build the CDI, reveled by 2^k Factorial Design analysis, enforces this affirmation. Further, the use of OCM to improve effectiveness can be used or not, depending on the urgency of the query time that the user desire. It can be an advantage to avoid the OCM comparison when the server is performing several requests at the same time or even running other

important processes. Thus, the OCM can be applied when the server load is low or when the query effectiveness is priority. Although the computation of the OCM increases the similarity measure cost of Sketch-Finder, even with this second step, our approach is still faster and more efficient than Mind-Finder.

8.1 Future Work

As future work, we intend to use two different thresholds, one to encode the image contours on the compressed-domain index and another for the OCM. If this is the case, we believe that just the most significant contours obtained with a larger threshold can be used to encode the contours in the CDI, while more contour details obtained with a smaller threshold can be used on the OCM algorithm for effectiveness refinement.

Sketch-Finder app can be improved to make a second query using the natural contours of some image selected by the user in the results of a previous query-by-sketch. Sketch-Finder app can also be modified to receive a hashtag attached to the query sketch, in order to bring semantic information and improve the results. An other idea is to keep all the sketches drawn by the Sketch-Finder app users, aiming to build a big sketch dataset associated to its hashtag, where this sketch dataset can fill the gap of annotated sketches bases. Moreover, a sketch collection can be used to evaluate SBIR systems, be applied to machine learning techniques aiming sketch classification and recognition, among other applications.

Finally, we intend to crawl web images and build a big dataset. This big dataset can be used on Sketch-Finder app, as well as, it can be used on the development of a web SBIR engine, that is an other future work idea.

Bibliography

Adler, M., Boutell, T., Bowler, J., Brunschen, C., Costello, A., Crocker, L. D., Dilger, A., Fromme, O., Gailly, J. L., Herbroth, C., Jakulin, A., Kettler, N., Lane, T., Lehmann, A., Lilley, C., Martindale, D., Mortensen, O., Pickens, K. S., Poole, R. P., Pehrson, G. R., Roelofs, G., Schaik, W. v., Schalnat, G., Schmidt, P., Stokes, M., Wegner, T., and Wohl, J. (2003). Information technology computer graphics and image processing Portable Network Graphics (PNG): Functional specification. Technical report ISO/IEC 15948:2003, International Organization for Standardization.

Ahmed, N., Natarajan, T., and Rao, K. (1974). Discrete cosine transfom. *IEEE Transactions on Computers*, 23(1):90–93.

Angelaccio, M., Catarci, T., and Santucci, G. (1990). Qbd*: A graphical query language with recursion. *IEEE Trans. Softw. Eng.*, 16(10):1150--1163.

Ansari, R. and Memon, N. (2000). *The JPEG lossy image compression standard.* Academic Press.

Antonini, M., Barlaud, M., Mathieu, P., and Daubechies, I. (1992). Image coding using wavelet transform. *IEEE Transactions on Image Processing*, 1(2):205--220.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898--916.

Aslandogan, Y. A., Thier, C., and Yu, C. (1996). A system for effective content based image retrieval. In *Proceedings of the fourth ACM international conference on Multimedia*, MULTIMEDIA '96, pages 429--430, New York, NY, USA. ACM.

Assfalg, J., Bimbo, A. D., and Pala, P. (2002). Three-dimensional interfaces for querying by example in content-based . . . *IEEE TRANS. VISUALIZATION AND COMPUTER GRAPHICS*, 8:2002.

Badue, C., Baeza-yates, R., Ribeiro-neto, B., and Ziviani, N. (2001). Distributed query processing using partitioned inverted files. In *In Proc. of the 9th String Processing and Information Retrieval Symposium (SPIRE*, pages 10--20. IEEE CS Press.

Belongie, S., Carson, C., Greenspan, H., and Malik, J. (1998). Color- and texture-based image segmentation using em and its application to content-based image retrieval. pages 675--682.

Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(4):509--522.

Benois-Pineau, J., Precioso, F., and Cord, M. (2012). *Visual Indexing and Retrieval*. Springer Publishing Company, Incorporated.

Beylkin, G., Coifman, R., and Rokhlin, V. (1991). Fast wavelet transforms and numerical algorithms I. *Comm. Pure Appl. Math.*, 44(2):141--183.

Bimbo, A. D., Mugnaini, M., Pala, P., and Turco, F. (1998). Visual querying by color perceptive regions. *Pattern Recognition*, 31(9):1241–1253.

Bird, C. L., Elliott, P. J., and Hayward, P. M. (1999). Content-based retrieval for european image libraries. In *Proceedings of the 1999 international conference on Challenge of Image Retrieval*, IM'99, pages 2--2, Swinton, UK, UK. British Computer Society.

Borgefors, G. (1983). *An Improved Version of the Chamfer Matching Algorithm*. Reports // LINKOEPING NATIONAL DEFENCE RESEARCH INSTITUTE.

Borgefors, G. (1988). Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(6):849--865.

Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media.

Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107--117.

Brogefors, G. (1988). Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):849–865.

Cao, Y., Wang, C., Zhang, L., and Zhang, L. (2011). Edgel index for large-scale sketch-based image search. In *CVPR*, pages 761–768.

Cao, Y., Wang, H., Wang, C., Li, Z., Zhang, L., and Zhang, L. (2010). Mindfinder: interactive sketch-based image search on millions of images. In *ACM Multimedia*, pages 1605–1608.

Carson, C., Belongie, S., Greenspan, H., and Malik, J. (1999). Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1026--1038.

Castelli, V. and Bergman, L. (2004). *Image Databases: Search and Retrieval of Digital Imagery*. Wiley InterScience electronic collection. Wiley.

Chalechale, A., Member, S., Naghdy, G., Mertins, A., and Member, S. (2005). Sketch-based image matching using angular partitioning. *IEEE Trans. Systems, Man, and Cybernetics*, 35:28--41.

Chandrasekaran, B., Josephson, J. R., and Benjamins, V. R. (1999). What are ontologies, and why do we need them? *IEEE Intelligent Systems*, 14(1):20--26.

Chang, S. K., Shi, Q. Y., and Yan, C. W. (1987). Iconic indexing by 2-d strings. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9(3):413--428.

Chatbri, H. and Kameyama, K. (2012). Towards making thinning algorithms robust against noise in sketch images. In *ICPR*, pages 3030–3033.

Chavda, M. and Wood, P. T. (1997). Towards an odmg-compliant visual object query language. In *Proceedings of the 23rd International Conference on Very Large Data Bases*, VLDB '97, pages 456--465, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Christopoulos, C., Skodras, A., Member, S., Member, S., and Member, T. E. (2000). The JPEG2000 still image coding system: An overview. *IEEE Transactions on Consumer Electronics*, 46:1103--1127.

Chui, C. K. (1992). *An introduction to wavelets*. Academic Press Professional, Inc., San Diego, CA, USA.

Cormen, T. H., Stein, C., Rivest, R. L., and Leiserson, C. E. (2001). *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 886--893, Washington, DC, USA. IEEE Computer Society.

Datta, R., Ge, W., Li, J., and Wang, J. (2007). Toward bridging the annotation-retrieval gap in image search. *MultiMedia, IEEE*, 14(3):24 –35.

Datta, R., Joshi, D., Li, J., James, and Wang, Z. (2006). Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 39.

Daubechies, I. (1992). *Ten Lectures on Wavelets*. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 233--240, New York, NY, USA. ACM.

Dawson-Howe, K. (2014). *A Practical Introduction to Computer Vision with OpenCV*. Wiley Publishing, 1st edition.

De Grandis, L. (1986). *Theory and use of color*. Abrams.

Del Bimbo, A. (1999). *Visual Information Retrieval*. The Morgan Kaufmann Series in Multimedia Information and Systems Series. Morgan Kaufmann.

Del Bimbo, A., Campanai, M., and Nesi, P. (1993). A three-dimensional iconic environment for image database querying. *IEEE Trans. Softw. Eng.*, 19(10):997--1011.

Del Bimbo, A. and Pala, P. (1997). Visual image retrieval by elastic matching of user sketches. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(2):121--132.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

DeVore, R. A., Jawerth, B. D., and Lucier, B. J. (1992). Image compression through wavelet transform coding. *IEEE Transactions on Information Theory*, 38(2):719–746.

Eitz, M., Hays, J., and Alexa, M. (2012). How do humans sketch objects? *ACM Trans. Graph.*, 31(4):44:1--44:10.

Eitz, M., Hildebrand, K., Boubekeur, T., and Alexa, M. (2009). A descriptor for large scale image retrieval based on sketched feature lines. In *Proceedings of the 6th Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '09, pages 29--36, New York, NY, USA. ACM.

Eitz, M., Hildebrand, K., Boubekeur, T., and Alexa, M. (2011). Sketch-based image retrieval: Benchmark and bag-of-features descriptors. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1624--1636.

Engel, D., Herdtweck, C., Browatzki, B., and Curio, C. (2011). Image retrieval with semantic sketches. In *Proceedings of the 13th IFIP TC 13 international conference on Human-computer interaction - Volume Part I*, INTERACT'11, pages 412--425, Berlin, Heidelberg. Springer-Verlag.

Faloutsos, C., Equitz, W., Flickner, M., Niblack, W., Petkovic, D., and Barber, R. (1994). Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3:231--262.

Fisher, Y., editor (1995). *Fractal Image Compression: Theory and Application.* Springer-Verlag, London, UK, UK.

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. (1995). Query by image and video content: The qbic system. *Computer*, 28(9):23--32.

Gonzalez, R. C. and Woods, R. E. (2006). *Digital Image Processing (3rd Edition).* Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Gupta, A. and Jain, R. (1997). Visual information retrieval. *Commun ACM*, 40(5):70–79.

Haykin, S. (2007). *Neural Networks: A Comprehensive Foundation (3rd Edition).* Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Hu, R. and Collomosse, J. (2013). A performance evaluation of gradient field hog descriptor for sketch based image retrieval. *Comput. Vis. Image Underst.*, 117(7):790--806.

Idris, F. M. and Panchanathan, S. (1995). Image indexing using vector quantization. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 373–380.

Jacobs, C. E., Finkelstein, A., and Salesin, D. H. (1995). Fast multiresolution image querying. In *Proceedings of SIGGRAPH 95*, pages 277--286.

Jaimes, A., Omura, K., Nagamine, T., and Hirata, K. (2004). Memory cues for meeting video retrieval. In *Proceedings of the the 1st ACM workshop on Continuous archival and retrieval of personal experiences*, CARPE'04, pages 74--85, New York, NY, USA. ACM.

Jain, A. K. (1989). *Fundamentals of digital image processing.* Prentice-Hall information and system sciences series. Prentice Hall.

Jain, R. (2008). *The Art Of Computer Systems Performance Analysis:.* Wiley India Pvt. Limited.

Knuth, D. E. (1998). *The art of computer programming, volume 3: (2nd ed.) sorting and searching.* Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA.

Kurita, T., Otsu, N., and Hirata, K. (1992). A sketch retrieval method for full color image database-query by visual example. In *1992 IEEE Computer Society Conference on Computer Vision and Applications (IAPR 1992)*, pages 530–533. IEEE.

Laaksonen, J., Koskela, M., Laakso, S., and Oja, E. (2000). Picsom content-based image retrieval with self-organizing maps. *Pattern Recogn. Lett.*, 21(13-14):1199--1207.

Lee, A. J. T. and Chiu, H.-P. (2003). 2d z-string: A new spatial knowledge representation for image databases. *Pattern Recognition Letters*, 24(16):3015–3026.

Lee, S. K. and Whang, K.-Y. (2001). Voql*: A visual object query language with inductively defined formal semantics. *J. Vis. Lang. Comput.*, 12(4):413–433.

Lee, Y. J. and Grauman, K. (2009a). Shape discovery from unlabeled image collections. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 2254–2261. IEEE.

Lee, Y. J. and Grauman, K. (2009b). Shape discovery from unlabeled image collections. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 0:2254–2261.

Lee, Y. J., Zitnick, C. L., and Cohen, M. F. (2011). Shadowdraw: Real-time user guidance for freehand drawing. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, pages 27:1--27:10, New York, NY, USA. ACM.

Liu, M.-Y., Tuzel, O., Veeraraghavan, A., and Chellappa, R. (2010). Fast directional chamfer matching. In *CVPR*. IEEE Computer Society.

Liu, Y., Zhang, D., Lu, G., and Ma, W.-Y. (2007). A survey of content-based image retrieval with high-level semantics. *Pattern Recogn.*, 40(1):262--282.

Liu, Y., Zhou, X., and Ma, W.-Y. (2004). Extraction of texture features from arbitrary-shaped regions for image retrieval. In *Inter. Conf. on Multimedia and Expo (ICME04)*, pages 1891--1894.

Loupias, E. and Sebe, N. (2000). Wavelet-based salient points: Applications to image retrieval using color and texture features. In *Proceedings of the 4th International Conference on Advances in Visual Information Systems*, VISUAL '00, pages 223--232, London, UK, UK. Springer-Verlag.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91--110.

Ma, W. Y. and Manjunath, B. S. (1997). Netra: a toolbox for navigating large image databases. In *Proceedings of the 1997 International Conference on Image Processing (ICIP '97) 3-Volume Set-Volume 1 - Volume 1*, ICIP '97, pages 568--, Washington, DC, USA. IEEE Computer Society.

Mallat, S. (2008). *A Wavelet Tour of Signal Processing, Third Edition: The Sparse Way*. Academic Press, 3rd edition.

Martin, D. R., Fowlkes, C. C., and Malik, J. (2004). Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):530--549.

Matusiak, S., Daoudi, M., Blu, T., and Avaro, O. (1998). Sketch-based images database retrieval. In *Proceedings of the 4th International Workshop on Advances in Multimedia Information Systems*, MIS '98, pages 185--191, London, UK, UK. Springer-Verlag.

Mehrotra, R. and Gary, J. E. (1995). Similar-shape retrieval in shape data management. *Computer*, 28(9):57--62.

Meyers, D. (1994). Multiresolution tiling.

Mezaris, V., Kompatsiaris, I., and Strintzis, M. G. (2003). An ontology approach to object-based image retrieval. In *In Proc. IEEE Int. Conf. on Image Processing (ICIP03*, pages 511--514.

Miano, J. (1999). *Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP*. ACM Press Series. Addison Wesley.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Commun. ACM*, 38(11):39--41.

Moores, C. (1951). Datacoding applied to mechanical organization of knowledge.

Müller, S. and Rigoll, G. (1999). Improved stochastic modeling of shapes for content-based image retrieval. In *Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries*, CBAIVL '99, pages 23--, Washington, DC, USA. IEEE Computer Society.

Müller, H., Müller, W., Squire, D. M., Marchand-Maillet, S., and Pun, T. (2000). Performance evaluation in content-based image retrieval: Overview and proposals.

Nievergelt, J., Hinterberger, H., and Sevcik, K. C. (1984). The grid file: An adaptable, symmetric multikey file structure. *ACM Trans. Database Syst.*, 9(1):38--71.

Paiva, J. G., Florian, L., Pedrini, H., Telles, G., and Minghim, R. (2011). Improved similarity trees and their application to visual data classification. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2459--2468.

Papantonakis, A. and King, P. J. H. (1995). Syntax and semantics of gql, a graphical query language. *J. Vis. Lang. Comput.*, 6(1):3–25.

Philbin, J., Chum, O., Isard, M., Sivic, J., and Zisserman, A. (2008). Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Plataniotis, K. and Venetsanopoulos, A. (2000). *Color Image Processing and Applications*. Digital Signal Processing. Springer.

Rao, K. R. and Yip, P. (1990). *Discrete Cosine Transform: Algorithms, Advantages, Applications*. Academic Press Professional, Inc., San Diego, CA, USA.

Robertson, S. (2004). Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60:2004.

Robertson, S., Zaragoza, H., and Taylor, M. (2004). Simple bm25 extension to multiple weighted fields. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 42--49, New York, NY, USA. ACM.

Saavedra, J. and Bustos, B. (2010). An improved histogram of edge local orientations for sketch-based image retrieval. In Goesele, M., Roth, S., Kuijper, A., Schiele, B., and Schindler, K., editors, *Pattern Recognition*, volume 6376 of *Lecture Notes in Computer Science*, pages 432–441. Springer Berlin Heidelberg.

Salton, G. and McGill, M. J. (1986). *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

Samet, H. (2006). *Foundations of Multidimensional and Metric Data Structures*. The Morgan Kaufmann series in computer graphics and geometric modeling. Elsevier Science.

Sciascio, E. D., Sciascio, E. D., and Mongiello, M. (1999). Query by sketch and relevance feedback for content-based image retrieval over the web. *Journal of Visual Languages and Computing*, 10:565--584.

Sclaroff, S., Taycher, L., and Cascia, M. L. (1997). Imagerover: A content-based image browser for the world wide web. In *In Proc. IEEE Workshop on Content-based Access of Image and Video Libraries*, pages 2--9.

Shannon, C. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27:379--423, 623--656.

Shechtman, E. and Irani, M. (2007). Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition 2007 (CVPR'07)*.

Sivic, J. and Zisserman, A. (2003). Video google: A text retrieval approach to object matching in videos. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 1470--, Washington, DC, USA. IEEE Computer Society.

Smeulders, A. W. M., Worring, M., Santini, S., Gupta, A., and Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349--1380.

Smith, J. R. and Chang, S.-F. (1996). Visualseek: a fully automated content-based image query system. In *Proceedings of the fourth ACM international conference on Multimedia*, MULTIMEDIA '96, pages 87--98, New York, NY, USA. ACM.

Smith, J. R. and fu Chang, S. (1996). Integrated spatial and feature image query. *Multimedia Systems*, 7:129--140.

Srinivas, M. and Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 24(4):656–667.

Stenger, B. D. R. (2004). Model-based hand tracking using a hierarchical bayesian filter.

Stollnitz, E. J., DeRose, T. D., and Salesin, D. H. (1995a). Wavelets for computer graphics: A primer, part 1. *IEEE Comput. Graph. Appl.*, 15(3):76--84.

Stollnitz, E. J., DeRose, T. D., and Salesin, D. H. (1995b). Wavelets for computer graphics: A primer, part 2. *IEEE Comput. Graph. Appl.*, 15(4):75--85.

Sun, X., Wang, C., Xu, C., and Zhang, L. (2013). Indexing billions of images for sketch-based retrieval. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 233--242, New York, NY, USA. ACM.

Sun, Z., Wang, C., Zhang, L., and Zhang, L. (2012). Query-adaptive shape topic mining for hand-drawn sketch recognition. In *Proceedings of the 20th ACM International Conference on Multimedia*, MM '12, pages 519--528, New York, NY, USA. ACM.

Tanenbaum, A. (2002). *Computer Networks*. Prentice Hall Professional Technical Reference, 4th edition.

Town, C. and Sinclair, D. (2001). Content based image retrieval using semantic visual categories. Technical report.

Tsai, Y. H. (2012). Hierarchical salient point selection for image retrieval. *Pattern Recogn. Lett.*, 33(12):1587--1593.

Tseng, K.-Y., Lin, Y.-L., Chen, Y.-H., and Hsu, W. H. (2012). Sketch-based image retrieval on mobile devices using compact hash bits. In *Proceedings of the 20th ACM international conference on Multimedia*, MM '12, pages 913--916, New York, NY, USA. ACM.

Venters, C. C., Hartley, R. J., and Hewitt, W. T. (2005). Content-based image retrieval query paradigms. In *Encyclopedia of Information Science and Technology (I)*, pages 556–563.

Villasenor, J. D. (1993). Full-frame compression of tomographic images using the discrete fourier transform. In Storer, J. A. and Cohn, M., editors, *Data Compression Conference*, pages 195–203. IEEE Computer Society.

Wang, C., Zhang, J., Yang, B., and Zhang, L. (2011). Sketch2cartoon: composing cartoon images by sketching. In *Proceedings of the 19th ACM international conference on Multimedia*, MM '11, pages 789--790, New York, NY, USA. ACM.

Zhang, D., Islam, M. M., and Lu, G. (2012). A review on automatic image annotation techniques. *Pattern Recogn.*, 45(1):346--362.

Zheng, W., Wang, C., and Chen, X. (2011). Shape-based web image clustering for unsupervised object detection? In *Proceedings of the 2011 IEEE International Conference on Multimedia and Expo*, ICME '11, pages 1--6, Washington, DC, USA. IEEE Computer Society.

Zhou, W., Lu, Y., Li, H., Song, Y., and Tian, Q. (2010). Spatial coding for large scale partial-duplicate web image search. In *Proceedings of the international conference on Multimedia*, MM '10, pages 511--520, New York, NY, USA. ACM.

Zhou, X. S. (2000). Cbir: from low-level features to high-level semantics. *Proceedings of SPIE*, 3974:426--431.

Attachment A

Paris Sketch

The objective of this work is to retrieve similar images according to an input sketch. In order to this, the user must draw this sketch image.

Figure A.1 presents an example of input image and its similar results. The image in black background and white lines is the input image, while the following color images are the similar ones. From left to right and up to bottom, Figure A.1 shows, in color, the most similar to the less similar images, according to the sketch input image.

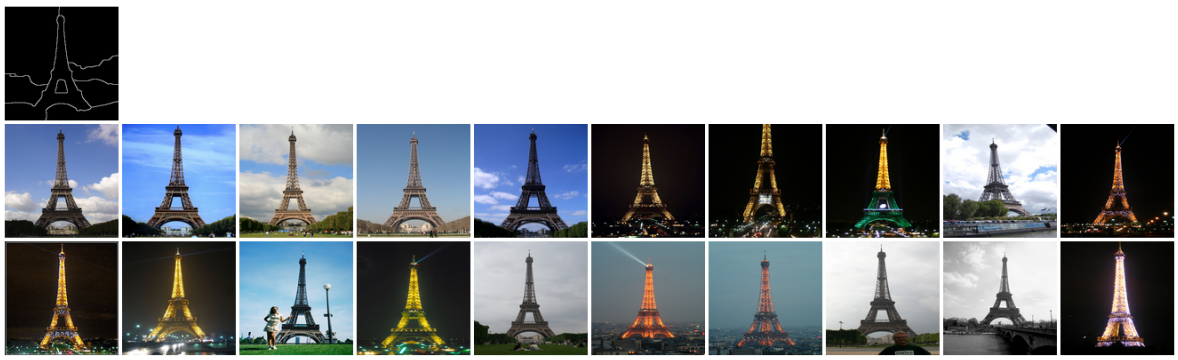

Figure A.1. Query example of the Eiffel Tower.

Instructions for drawing the sketch:

1. Use your favorite tool for drawing, it can be the windows paint for example.

2. Open the black image of the landmark you want to draw (zip file attached in your e-mail). For example, for Eiffel Tower, use *eiffel_XMW.bmp* and so on. The list of landmarks and 6 examples images for each one are shown in the next pages of this document

3. Draw the landmark in the black background with thin white lines, then save the file in the same format and resolution.

4. Draw the landmark in the form, position and desired perspectives, there is no restrictions. You can try to base your sketch in one of the six example images of the landmarks or you can just create one image that represent the monument without imitate any given example.

5. You can try to base your sketch in one of the six example images of the landmarks or you can just create one image that represent the monument without imitate any given example.

6. Try to not draw "blurry" lines.

7. The sketches does not need to be sophisticated, simple lines are welcome.

8. Clue: create the drawn in the zoom mode of the painting tool, so it is easier to not draw blurry lines when using the free hand pencil.

9. In the final, please send the sketches to the e-mail: fragapimentel@gmail.com You don't have to draw all the 11 sketches, the sketches sent are welcome.

10. You don't have to draw all the 11 sketches, the sketches sent are welcome.

Some examples of sketches are shown in Figure A.2:

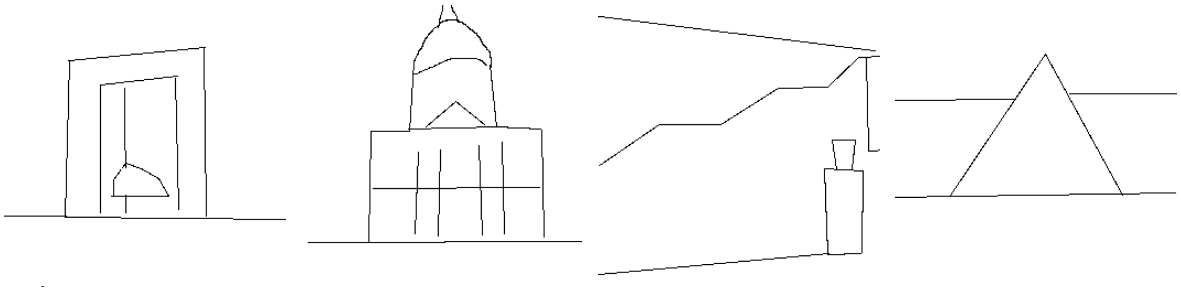

Figure A.2. Examples of sketches

Thank you very much for your contribution!

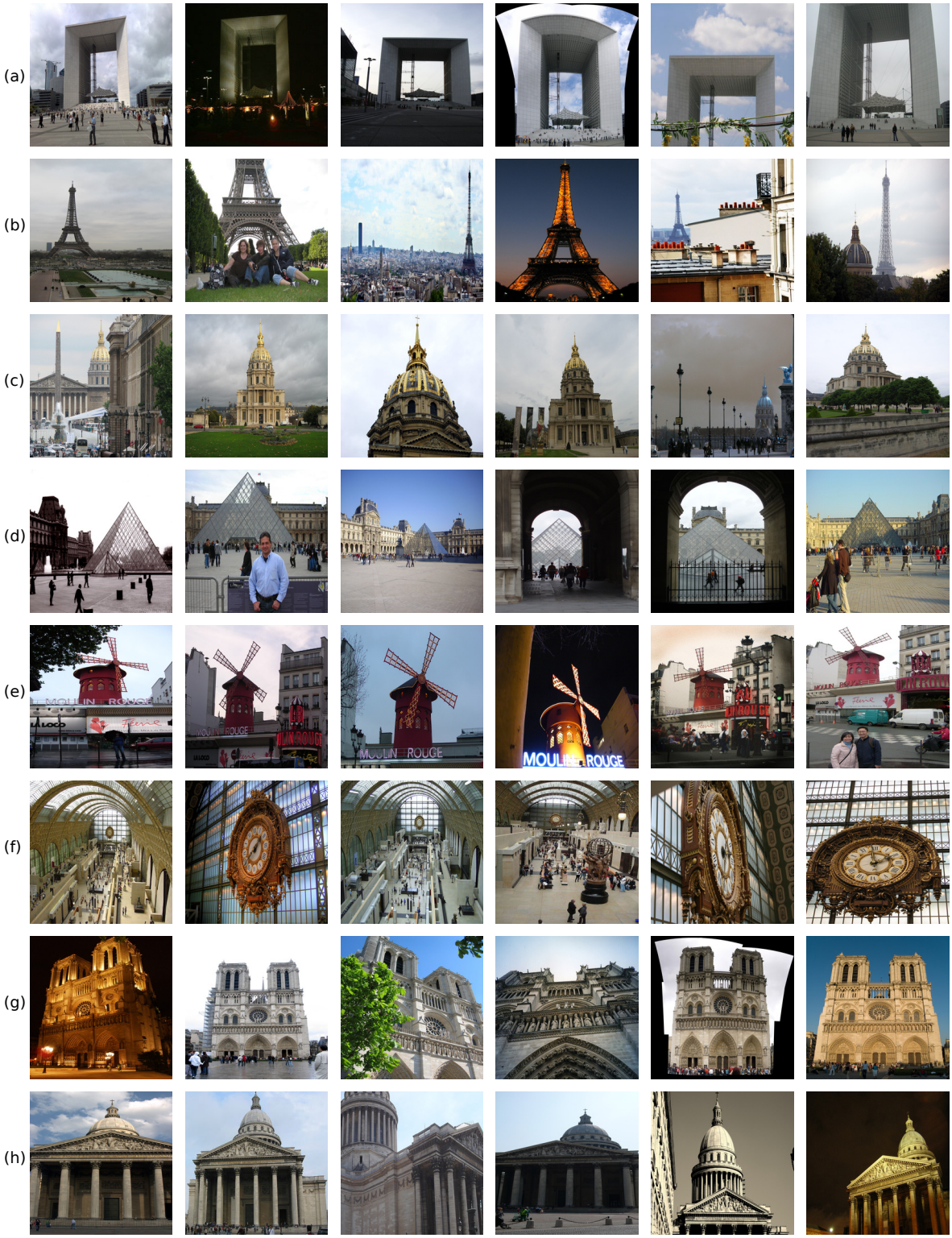

Figure A.3. Paris images examples - (a) *La Defense*, (b) *Eiffel Tower*, (c) *Hotel des Invalides*, (d) *Musée du Louvre*, (e) *Moulin Rouge*, (f) *Musée d'Orsay*, (g) *Notre Dame*, (h) *Panthéon*

Figure A.4. Paris images examples - (i) *Pompidou*, (j) *Sacré Cœur* and (k) *Arc de Triomphe*