FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Robust and efficient retrieval system for fashion magazines recognition

Fábio Dantas Carneiro Araújo e Silva

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Internal supervisor: Daniel Moura

External supervisor: André Vidal

February 27, 2015

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES 2014/2015

A Dissertação intitulada

"Robust and Efficient Retrieval System for Fashion Magazines Recognition"

foi aprovada em provas realizadas em 12-02-2015

o júri

Presidente Professor Doutor Jaime dos Santos Cardoso
Professor Auxiliar do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professora Doutora Paula Viana
Professor Adjunto Departamento de Engenharia Eletrotécnica do Instituto Superior de Engenharia do Porto

Professor Doutor Daniel Cardoso de Moura
Investigador Auxiliar do Departamento de Engenharia Informática da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.

Autor - Fábio Dantas Carneiro Araújo e Silva

Faculdade de Engenharia da Universidade do Porto

Resumo

Nos últimos anos, tem-se assistido a um aumento dos acessos a câmaras, especialmente pela sua incorporação sistemática em smartphones. Em adição, com o aumento da largura de banda de rede e um grande compromisso mundial com a mídia social *online*, hoje enfrentamos uma enorme quantidade de dados multimédia.

Trabalhos relevantes sobre a área muitas vezes esquecem as limitações inerentes dos dispositivos móveis, resultando numa clara necessidade de técnicas mais adequadas para estes dispositivos. Nesta ordem de ideias o principal objetivo deste trabalho é investigar o uso de descritores de *keypoints* de representação binária no contexto do reconhecimento e recuperação de revistas de moda.

No contexto da recuperação de imagens baseada em conteúdo, uma abordagem popular para a descrição das imagens é modelo *bag of keypoints* que utiliza um vocabulário aprendido a partir de um conjunto de descritores de *keypoints* produzindo assinaturas globais de imagens, quantizando cada descritor de *keypoint* extraído de uma imagem. Para a geração do vocabulário o modelo *bag of keypoints* baseia-se normalmente em alguma variante do algoritmo de agrupamento de dados *k-means*, que é bem definido apenas para dados de valor real. Na base de uma discussão aprofundada sobre as diferentes questões relacionadas ao uso de dados binários no modelo *bag of keypoints*, neste trabalho é proposto e validado um sistema de recuperação de imagens baseada em conteúdo utilizando descritores binários. Os resultados mostraram que os descritores binários, embora simples e compactos, são muito discriminativos e têm um papel a desempenhar nos sistemas de recuperação de imagens utilizando a abordagem *bag of keypoints*, uma vez que atingem precisões semelhantes do que descritores tradicionais.

Esse trabalho levou ao desenvolvimento de uma execução robusta e eficiente que após optimizações tem o potencial de melhorar os sistemas visuais em cenários de pesquisa do mundo real.

Abstract

Recent years has seen the rise of cheap and ubiquitous access to cameras, especially by its systematic embedding into smartphones. In addiction, with the increase of network bandwidth and a large worldwide commitment to online social media, we face today a huge amount of multimedia data.

Relevant works on the area often miss the inherent limitations of mobile devices resulting in a clear need of more adequate techniques. In this order of ideas the main purpose of this work is to investigate the use of binary keypoint descriptors in the context of fashion magazines recognition and retrieval.

In content-based image retrieval a popular approach for images description is the bag of keypoints model which using a vocabulary learned from a set of local keypoint descriptors produces global image signatures by quantizing each local keypoint descriptor extracted from an image. For the vocabulary generation the bag of keypoints model typically relies on some variant of the k-means clustering algorithm which is well defined only for real-valued data. In the basis of a thorough discussion about the different issues regarding the use of binary data in the bag of keypoints model, in this work is proposed and validated a content-based image retrieval system using binary descriptors. The results showed that binary descriptors, while simple and compact, are very discriminative and have their role to play in image retrieval systems using the bag of keypoints approach since they achieve similar accuracy than traditional descriptors.

This work led to the development of a robust and efficient implementation that after optimizations has the potential to improve visual search systems in real world scenarios.

Acknowledgements

I want to thank my family, I could never have done this work without their love and support. They were always my greatest supporters.

I would like to thank my advisors Daniel Moura and André Vidal for supervising my work, for all the discussions, patience and availability.

I would also like to thank all the ASAP54 team, that received me so well in the company and for the daily good work atmosphere.

Fábio Silva

"Ambition is the path to success.
Persistence is the vehicle you arrive in. "

Bill Bradley

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Context and Goals . . . 2
- 1.3 Contributions . . . 4
- 1.4 Document Structure . . . 4

2 Related methods and techniques **5**
- 2.1 Image retrieval . . . 5
- 2.2 Keypoint-based image description . . . 6
  - 2.2.1 Keypoint detection . . . 6
  - 2.2.2 Keypoint description . . . 7
  - 2.2.3 Keypoint matching . . . 8
- 2.3 Bag of keypoints model . . . 8
  - 2.3.1 k-means clustering . . . 9

3 Keypoint detectors and descriptors **11**
- 3.1 Keypoint detectors . . . 11
  - 3.1.1 Corner detectors . . . 11
  - 3.1.2 Blob detectors . . . 14
- 3.2 Keypoint descriptors . . . 17
  - 3.2.1 Real-valued descriptors . . . 17
  - 3.2.2 Binary descriptors . . . 19
- 3.3 Performance evaluation . . . 21

4 Proposed system for content-based image retrieval **23**
- 4.1 Logical architecture . . . 23
- 4.2 Solution approach . . . 25
  - 4.2.1 Vocabulary generation in the binary space . . . 26
  - 4.2.2 Quantization . . . 27
  - 4.2.3 Inverted index . . . 27
  - 4.2.4 Re-ranking and validation . . . 28
- 4.3 Final remarks . . . 29

5 Experimental Results **31**
- 5.1 Equipment and software used . . . 31
- 5.2 Dataset . . . 32
- 5.3 Keypoint detectors and descriptors evaluation . . . 33
- 5.4 Brute-Force system approach . . . 42

5.5 Vocabulary generation 42
5.6 Vocabulary size 43
5.7 Vocabulary generalization 44
5.8 Confidence System 45
5.9 Final prototype 47

6 Conclusions and Future Work **51**
6.1 Future Work 52

References **53**

List of Figures

1.1 images.google.com retrieves visually similar images from an image queried by an user. . . . 3

2.1 K-means algorithm representation. Data points are shown as dots, and cluster centers are shown as crosses. (a) Original dataset. (b) Random initial cluster centers. (c-f) Illustration of running two iterations of *k-means*. In each iteration, each data point is assigned to the closest cluster center (shown by "painting" the data points with the same color as the cluster center to which is assigned); then each cluster center is moved to the mean of the points assigned to it. Source: [1]. 10

3.1 Harris corners on rotated images. Source: [2] . . . 12
3.2 FAST detection method: pixel p and the corresponding circle of 16 pixels used to test if p is a point of interest. Source: [3] . . . 13
3.3 Overview of the DoG blob detection. Source: [2] . . . 15
3.4 Left: discretized and cropped gaussian second order partial derivatives Lyy (y-direction) and Lxy (xy-direction). Right: the respective approximations using box filters. Source: [4] . . . 16
3.5 SIFT keypoint descriptor is built by first computing the gradient magnitude and orientation in a region around the keypoint location. The results are then weighted by a Gaussian window (left). These samples are subsequently summed into orientation histograms (right). The histograms are then concatenated to form a final representation. The standard SIFT descriptor uses a 16×16 sample array on the left and 4×4 histograms of size 36 on the right, resulting in a 128-dimensional vector which is ultimatly normalized. Source: [5] . . . 18
3.6 Sampling schemes for the different binary descriptors. Sources: [6, 7, 8, 9] . . . 20
3.7 Image matching done using Oriented FAST and rotated BRIEF. Green lines denote matches, while red circles denote unmatched points in both images. In this case, there is a visible viewpoint variation. Source: [7] . . . 21

4.1 Representation of the vocabulary generation pipeline. . . . 24
4.2 Representation of the pipeline used for building the database. . . . 25
4.3 Representation of the pipeline used in query retrieval. . . . 25
4.4 Example of the voting scheme step used in k-majority to compute centroids from clusters of binary data. Note: n_d is the number of keypoint descriptors assigned to a cluster C. . . . 27
4.5 Illustrative example of the efficient indexing structure developed. Left: All database images are loaded into the index mapping visual words to image numbers. Right: A new query image is mapped to indices of database images that share a visual word. . . . 28

5.1 Sample pages from the Vogue UK (June, 2014) fashion magazine used to create the dataset of photos taken from this magazine. 32
5.2 Set of images taken from the Fashion Magazine dataset. In the left column is shown the original magazine image and in the right column the respective query images. It can be seen a great variety of conditions in the query images, for example, different perspectives and scales, rotations, different lighting conditions, etc. 34
5.3 Feature detection computational complexity results for the two experiments made. 36
5.4 Keypoint detectors and descriptors evaluation sequences. 37
5.5 Repeatability score for image sequences with affine and perspective transformations (viewpoint and scale & rotation changes). 38
5.6 Repeatability score for image sequences with non-geometric transformations (in this case illumination and blur). 39
5.7 Descriptors run time using $N = 1000$ keypoints detected by SURF. 40
5.8 Average number of best matches in the Mikolajczyk sequence. 41
5.9 Vocabulary generation tests. 43
5.10 Precision values obtained for several vocabularies created with ORB keypoints clustered using k-majority. 44
5.11 Top N precision and the validation results using the brute force approach. 45
5.12 Some examples of query images of the type 5. The top line shows images that the system recognizes successfully, returning the correct page despite the difficulty associated with such images. Below examples of query images for which the system can not return a reliable result are presented. 48
5.13 Interface developed to be used as system's proof of concept. 49

List of Tables

5.1 Average processing time for the feature detection in an image of size 1024×768 pixels (N = 1000 keypoints). 35
5.2 Average repeatability (in percentage) obtained for different image sequences. . . 39
5.3 Correct correspondences rate for the Fashion Magazine dataset. 42
5.4 Generalization test comparing the precision of the queries from the Fashion Magazine dataset (Vogue UK magazine) when using a vocabulary generated from images of the same fashion magazine with a vocabulary generated from another fashion magazine (Cosmopolitan UK). 45
5.5 Precision obtained for different values of the confidence ratio and the top N pages used to perform the brute force. 46
5.6 Error percentage obtained for different values of the confidence ratio and the top N pages used to perform the brute force. 46
5.7 Percentage of times the system will not have the confidence to answer to the query inserted by the user. 46
5.8 Percentage of matches between the query image and the corresponding magazine page in the database. 47

Abbreviations

AGAST	Adaptive and Generic Acellerated Segment Test
BRIEF	Binary Robust Independent Elementary Features
BRISK	Binary Robust Invariant Scalable Keypoints
CBIR	Content-Based Image Retrieval
CPU	Central Processing Unit
DoG	Difference of Gaussians
DoH	Determinant of Hessian
FAST	Features from Acellerated Segment Test
FREAK	Fast Retina Keypoint
LoG	Laplacian of Gaussian
ORB	Oriented FAST and Rotated
RANSAC	Random Sample Consensus
SIFT	Scale Invariant Feature Transform
SURF	Speeded-Up Robust Features

Chapter 1

Introduction

A large number of computer vision applications, such as visual search, object recognition or 3D reconstruction are based on finding correspondences between images. This is a challenging problem that can be solved by representing image patches, in a way that is invariant to viewpoint changes and different imaging conditions, finding the correspondences in the resulting representation space. The representation of an image patch, referred to as a *keypoint descriptor*, can be typically modelled as a multi-dimensional vector of floating-point or binary values. In addition to robustness to various image transformations, keypoint descriptors should also possess a high discriminative power, allowing to find its correct match with good probability even in a large database of keypoints. The keypoint matching problem has received considerable attention as evidenced by the large number of publications on this topic, such as [5, 10, 4, 2, 6, 7, 8, 9] to name a few. Despite the significant progress that been made, it remains a very active research area, because of the continuously changing and ever more demanding nature of the applications built that rely on it and since most of the proposed solutions are mainly designed for standalone machines with high processing power and few memory limitations.

1.1 Motivation

With the proliferation of camera-enabled mobile devices (e.g. phones and tablets), there is an ever growing need for Computer Vision technologies deployed on portable devices that have limited computational power and storage space. The increasing content of visual data along with the need to access this data from any point in the world and as quickly as possible requires extremely fast processing. This new scale of processing has driven several recent works that propose binary descriptors [6, 7, 8, 9], which reduce memory usage and allows for significantly faster processing due to the efficient computation of the Hamming distance (a proper metric to compute distances in the binary space). Besides this, they are typically built as a concatenation of simple intensity comparisons which results in an extremely short computation time. Binary descriptors have quickly become an attractive alternative to floating-point descriptors, especially for real-time applications run on low-end handheld devices.

However, maintaining a large number of local feature descriptors to represent the image content, even if binary, is not scalable due to the prohibitively high dimensionality of the resulting representation which grows linearly with the number of local descriptors and their dimensionality, becoming overly expensive to compute similarities between image representations in large scale databases. A lot of research has been focused on providing working solutions to address this problem in the multi-dimensional space of floating-point vectors [11, 12, 13, 14]. Surprisingly few works, however, focus on this problem in the case of binary descriptors. It is therefore necessary to investigate if the solutions available for the floating-point vectors can be used to produce feature encoding schemes in the binary descriptors case.

1.2 Context and Goals

Being able to describe an image based on its regions using an effective and efficient representation, robust to various imaging conditions and viewpoint changes, is crucial for a variety of Computer Vision applications. Among the most important ones are:

Wide baseline image matching. Matching a pair of images taken from substantially different viewpoints, known as wide baseline matching, is an important component of 3-D reconstruction systems. It is usually carried out by first detecting salient regions in each of the images, followed by a matching step which based on the distance in the region descriptor space (e.g. by nearest-neighbour matching). This brings up the importance of having a discriminative local feature descriptor, i.e. the distance between the descriptors of regions corresponding to the same part of a scene should be smaller than the distance between descriptors of regions coming from different parts of the scene.

Large-scale visual search. Due to the existence of search engines like Google[1], the idea to search the web from a set of words is, nowadays, a natural way to discover the world and to answer several questions. However, with image capturing devices being in abundance, the users become more and more interested in querying the internet not only with textual hints but also with visual data. In fact, a visual image can convey an idea or an emotion more effectively than words. Thus, there is a significant demand to provide solutions for visual search which can be defined as retrieving information, both in textual and visual domains, in response to a visual query, e.g. in the form of an image. A conventional approach to visual search is based on the text retrieval scheme. It relies on the representation of images using visual words, which are obtained by quantizing image local descriptors. Figure 1.1 presents a screenshot of a representative example of a large-scale visual search application, namely **images.google.com**. From an image queried by the user a set of high relevant images are retrieved. Also, a new class of applications which use the camera phone

[1] www.google.com

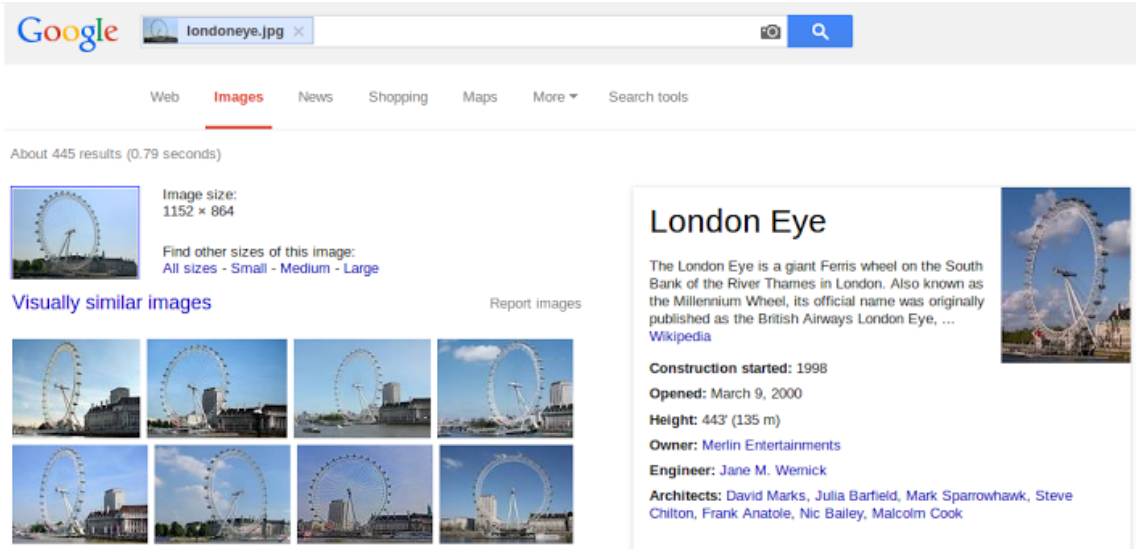


Figure 1.1: images.google.com retrieves visually similar images from an image queried by an user.

to initiate search queries about objects in visual proximity to the user has emerged. Such applications can be used, e.g., for identifying products, comparison shopping, finding information about movies, CDs, buildings, shops, real estate, print media, artworks and fashion. Some good examples of such systems include Google Googles[2], Amazon Shopping[3], CamFind[4] and ASAP54[5].

This work tackles the problem of visual recognition of specific pages from fashion print magazines in a large collection of images. The main goal is to retrieve the correct page and some relevant information about the contents of it using only a query image taken by user. Visual recognition of specific fashion magazines pages is a complex problem due to the fact that there are sometimes very similar pages, contributing to the increase in the system's likelihood to confuse the correct page with no relevant pages. Besides this, the image must be retrieved quickly and accurately, despite some imaging conditions such as scale, lighting and partial occlusions. In this thesis a robust and efficient method to retrieval of specific pages of fashion magazines, is proposed. During the development of the method several combinations of keypoints detectors and binary descriptors were tested in order to ascertain which of the combinations is best suited to this specific problem. Furthermore, given the extensive research on binary descriptors in the scientific community, such a comparison was made with standard techniques.

[2] https://support.google.com/websearch/answer/166331?hl=en
[3] https://play.google.com/store/apps/details?id=com.amazon.mShop.android.shopping&hl=en
[4] http://camfindapp.com
[5] http://www.asap54.com

1.3 Contributions

This thesis addresses the problem of learning discriminative image representations. This means the representation of images or their regions as floating-point or binary vectors in the vector space. Such representations are the basis of several computer vision frameworks, since the latter rely on a suitable representation of the image data they are dealing with. Since the floating-point feature descriptors are older than the binary ones, having been used in numerous applications, it is necessary to understand if the methods used for applications using real valued vector representation are susceptible to be used also in applications using binary representations. Thus, this thesis provides an analysis of the multi-dimensional space generated by binary descriptors and a robust and efficient method for visual search based on a compact representation of images, referred as *image signature*. More precisely the contributions of this thesis are the following:

- An extensive study of several state-of-the-art feature detection and description techniques.
- A performance evaluation of Open Source Computer Vision Library (OpenCV) implementations of several keypoints detectors and descriptors effectively yielding the best solution to that specific problem.
- A new approach to integrate the binary descriptors in the bag of keypoints representation.
- A prototype visual search system for fashion print magazines. The system takes a picture from a user as input, returning the corresponding magazine and page together with relevant information about the contents of the pages.

1.4 Document Structure

This dissertation is divided into six main chapters, the first of which is the Introduction. Chapter 2 offers an overview of the state of the art methods and technologies used to solve problems in this context. Chapter 3 an extensive overview of the state of the art in the field of keypoint detectors and descriptors is provided. In Chapter 4 the solution approach that has been performed within the context of this dissertation is explained, based on the study demonstrated on Chapters 2 and 3. Chapter 5 exposes the test environment and main results that were obtained following the implementation of the application. Finally, on Chapter 6, some conclusions are made and some future work is defined.

Chapter 2

Related methods and techniques

Print magazine recognition is essentially an content-based image retrieval (CBIR) task with many challenges caused by the large variation of imaging conditions. It is closely related to many computer vision problems and tasks ranging from image description to object recognition. In this chapter the basic concepts about image retrieval are presented, focusing on the standard image retrieval scheme.

2.1 Image retrieval

Image retrieval is the task of searching, finding and retrieving the most similar images from a database to a given query image. As the images grow in complexity and diversity, retrieving the right images in large scale catalogues, or databases becomes a difficult challenge. The task is challenging for two main reasons. Firstly, the system should retrieve occurences of the query object despite all the difficult imaging conditions seen in real-world images. These include changes as scale, viewpoint, lighting and partial occlusions. Secondly, the system should scale to very large image corpora while still producing good retrieval results, and being able to rapidly return results to the user (i.e. $\sim$ 1 second per query).

As already mentioned, searching images according to their semantic content is a very challenging problem since the system is ideally robusto to several factors such as resolution, illumination variations and occluded objects. A number of approaches had tacked the problem of retrieval under these difficulties with some success, but they became truly scalable and robust when the link was made to text retrieval techniques. While global features are known to be limited in face of these difficulties, which describe a picture in a holistic way (eg. one histogram represents a signature for the whole image), keypoint descriptors provide a way to describe several salient patches within the images, demonstrating great discriminative power. The cardinality of the set of keypoint descriptor vectors depends on the detected points in the picture, resulting in a huge number to cope with for the large-scale retrieval system.

Nowadays, the most popular approach is the bag of keypoints model, first proposed by [11]. The idea is to first quantize local descriptors into "visual words" and represent each image as

a vector of words, mimicking the text-document search paradigm. Effectively, such approach departs from one popular indexing scheme used in systems using text documents search, where the word occurences in a phrase are moduled as a histogram over a pre-existing vocabulary of words, named Bag of Words model. Then text-retrieval systems can mimic, applying scalable indexing and fast search on this vector space. Bag of keypoints has shown good performances not only for the retrieval [15] task but also for other vision tasks like object recognition [16], image classification [12, 17].

2.2 Keypoint-based image description

Due to the increase in resolution of digital images and since they are high dimensional complex data structures, computationally efficient ways of representing images with smaller dimension have been studied so that certain types of image processing techniques can be used. In the field of image retrieval various approaches have been proposed to make content-based image representations. First, most of the approaches were based on overall image representations also known as *global features* such as color histograms and their variations. This type of approach has been used successfully at least in applications where the images have distinctive colours as the user's main interest is the total composition of the image as a whole, rather than one foreground object, for instance. Global representations of the image do not distinguish foreground with the background of the images, mixing both parts of the information, limiting the usefulness of systems based on these approaches to cases with clean background or cases which require prior segmentation routine to extract the object of interest from the input image.

One of the main problems of global features is that most images come from the real world and thus contain non-distinctive parts (e.g. uniform colour patches) contributing to the representation of the image in global approaches. It becomes then very useful to extract only meaningful parts of the image; in other words, their aim is to retrieve only a small amount of very discriminative image areas. A common way in computer vision to fulfil this purpose is to find locations containing a lot of usefull and desciminative information, and then to describe them using their local pixel neighbourhood. Such locations are generally called keypoints, and their description is generally referred to as descriptors or keypoint descriptor. The keypoint-based image description is thus a set of keypoints and their descriptors. The problem of the high dimensionality of standard representation is solved because the descriptors are low-dimensional structures (arbitrarily). In this work we will settle our analysis at this type of image representation.

2.2.1 Keypoint detection

As explained above, the first requirement of keypoint-based image representation is to find discriminative image patches. This process is commonly known as keypoint detection and has been solved using several approaches [18, 19, 20, 5, 4, 3, 21] to name a few. The common goal of the several developed techniques is to find semantically meaningful object parts. The keypoint detectors select points in the image based directly on the underlying intensity patterns. These detected

points are ideally discriminative points in the image and have a certain amount of invariance to image transformations. This means that given two images of certain objects taken from different conditions (e.g. viewing angle, illumination, etc), the detector should be able to locate the same keypoints in both images. Optimal keypoint detectors should detect keypoints that are [2]:

- **repeatable:** Given two images of the same object or scene, taken under different viewing conditions, a high percentage of the keypoints detected on the scene part visible in both images should be found in both images.
- **local:** The keypoints should be local, so as to reduce the probability of occlusion and to allow simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions.
- **distinctive:** The intensity patterns underlying the detected keypoints should show a lot of variation, such that they can be distinguished and matched.
- **sufficient:** The number of detected keypoints should be sufficiently large, such that a reasonable number of salient patches are detected even on small objects. However, the optimal number of keypoints depends on the application. Ideally, that number should be adaptable over a large range by a simple and intuitive threshold.
- **accurate:** The detected keypoints should be accurately localized, both in image location, as with respect to scale and possibly shape.
- **efficient:** Preferably, the keypoint detection in a new image should allow for time-critical applications.

Clearly, the importance of the properties described depends on the actual application and settings, and compromises need to be made. Repeatability is the most important quality for a keypoint detection technique and is required in all application scenarios [2]. It directly depends on the other properties like invariance, robustness, etc. Depending on the application, increasing or decreasing the number of keypoints may result in higher repeatability.

In the next chapter the most important techniques in the literature to detect keypoints are summarized.

2.2.2 Keypoint description

Having detected keypoints in the image, the next step is to represent them in a way that is invariant to unwanted image transformations. This task is commonly referred as feature description and it consists in encoding certain properties of the image in the local neighborhoods centered at the detected keypoints into a vector, referred as descriptor. Several characteristics can be desired for a good keypoint descriptor. First, the size (i.e. dimensionality) of the descriptor need to be bounded to a sufficiently small value, in order to keep the benefits of the keypoint-based image representation, and to be able to match features efficiently. For matching purposes, the description should

be descriptive enough to allow a good discrimination but, at the same time, robust to variations in its neighbourhood. Robustness typically refers to invariance to in-plane rotation, scale, affine or perspective transformation, compression noise, etc. As seen in the previous section, scale and rotation information can be provided by the detector.

2.2.3 Keypoint matching

A simple approach for pairwise image matching consists in linearly comparing their descriptors and counting how many of them match each other. Descriptors can be compared by thresholding the distance between them which can be computed using some distance measure such as Euclidean, Manhattan or Cosine distance. An alternative criteria to evaluate the quality of a matching candidate was proposed in the original SIFT matching scheme were it was introduced the notion of ratio threshold, a measure where a pair of features is considered a match if the distance ratio between the closest and second closest matches is below some threshold t:

$$t > \frac{d(D, D'_{nn})}{d(D, D'_{2nn})} \tag{2.1}$$

where D is the descriptor to be matched and D'_{nn} and D'_{2nn} are the nearest and the second nearest descriptors from the model database, with $d(D_1, D_2)$ denoting the Euclidean distance between two descriptors, D_1 and D_2. An empirically determined threshold value of $t = 0.8$ is typically used. According to [5] the ratio threshold is effective for general recognition because correct discriminative keypoints often have the closest neighbor significantly closer than the closest incorrect match.

2.3 Bag of keypoints model

The Bag of keypoints model was originally introduced in Computer Vision by [11] where the aim was detecting objects in a video sequence. To that end it reformulates the CBIR problem as a text retrieval one where the documents are the images in the database and the analog to a word in text documents is a visual word, a set of neighboring pixels sharing a texturing pattern. The process starts by extracting keypoint descriptors from a collection of database images, which are then quantized into visual words, and finally text search methods are used to retrieve similar images to a given query image. An optional re-ranking step based on geometric constraints is performed on the top "closest" results.

The general method to create a bag of keypoints model is derived from the initial bag of words (BoW) model. That is, first a vocabulary needs to be constructed. This is generally done using keypoint-based representation of images. The general method to create a vocabulary (also called codebook) of size n (*i.e.* the number of visual words) is the following:

1. Detect keypoints and extract descriptors from a set of images.

2. Apply a clustering method to find k representatives in the keypoint descriptor space.

3. The set containing the k representatives is defined as the vocabulary.

Recently, hierarchical clustering has been used instead of flat clustering and have been shown to yield to good performance [15]. In this context, the codebook is sometimes called a Vocabulary Tree.

The bag of keypoints representation of an image I given a vocabulary V is then created using the following steps:

1. Detect keypoints and extract descriptors in I using the same keypoint detectors and descriptors than the ones used to create the visual vocabulary.

2. Find the visual words corresponding to each keypoint descriptor by assigning its closest visual word in V using a nearest neighbour search.

3. Create an histogram of the visual words in I with respect to the codebook V.

4. The normalized histogram is the bag of keypoints representation of I (also referred as *image signature*).

2.3.1 k-means clustering

Clustering is an unsupervised learning technique of machine learning typically used in the bag of keypoints model.

The method of k-means clustering (or simply k-means) takes unlabeled input data and gives as output a k-partition of the data. The k partitions, here called clusters, are disjoint and non-hierarchical. Thus, each data sample is labeled as belonging to one of the clusters. Given a k number of clusters and a set of n data points $\chi \in \mathbb{R}^d$.The number k of centers C should be chosen so as to minimize the potential function:

$$\phi = \sum_{x \in \chi} \min_{c \in C} ||x - c||^2 \tag{2.2}$$

The k-means algorithm is a simple and fast algorithm for the clustering problem, although it offers no approximation guarantees at all and is highly dependent of the initialization of the clusters. The standard algorithm was first proposed in [22] and has also been introduced in [23], which is why the algorithm is generally referred to as Lloyd's algorithm (or Lloyd-Forgy algorithm).

k-means algorithm (Lloyd-Forgy)

- **Initialization:** Initialize k vectors, which refer to the cluster centers $C = \{c_1, c_2, \ldots, c_k\}$.

- **Assignment step:** For each $\{i \in 1, \ldots, k\}$, set the cluster C_i to be the set of points in χ that are closer to c_i than they are to c_j for all $j \neq i$.

- **Update step:** For each $\{i \in 1,\ldots,k\}$, set c_i to be the center of mass of all points in C_i:

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \tag{2.3}$$

- Alternate between assignment and update step until C no longer changes.

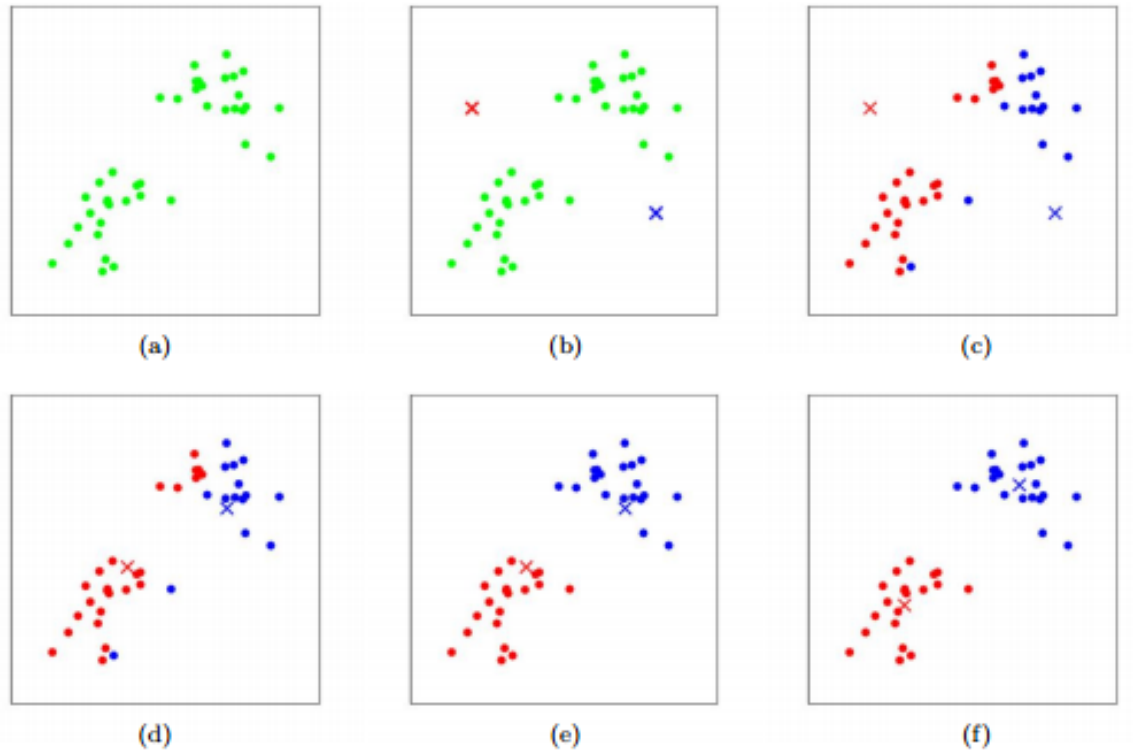


Figure 2.1: K-means algorithm representation. Data points are shown as dots, and cluster centers are shown as crosses. (a) Original dataset. (b) Random initial cluster centers. (c-f) Illustration of running two iterations of *k-means*. In each iteration, each data point is assigned to the closest cluster center (shown by "painting" the data points with the same color as the cluster center to which is assigned); then each cluster center is moved to the mean of the points assigned to it. Source: [1].

Figure 2.1 shows a typical representation of the algorithm. For the initialization problem a new algorithm *k-means++* was proposed in [24]. The method provides a distribution of initial cluster centers which allows the algorithm to converge quickly to a clustering with small error (with respect to the optimal clustering).

Chapter 3

Keypoint detectors and descriptors

This chapter presents the main methods to detect keypoints in an image. After that, several keypoints descriptors used in the literature are also presented.

3.1 Keypoint detectors

Depending on the keypoint type detected, detectors can be organized in two mainclasses, namely corner-like detectors and blob-like detectors. Below are presented several examples for each class in chronological order of appearance in the literature.

3.1.1 Corner detectors

The expression *corner detection* is commonly used in computer vision but has a specific meaning. The detected points by corner detectors correspond to points in the 2D image with high curvature. These do not necessarily correspond to projections of 3D corners. Corners are found at various types of junctions, on highly textured surfaces, at occlusion boundaries, etc. For many applications, this is sufficient, since the goal is to have a set of stable and repeatable keypoints [2]. In this section some of these algorithms are summarized.

Harris detector. The Harris detector [18], also known as the Harris operator is one of the first successful attempts to localize discriminative points in an image. It proposes to use a measure of cornerness, defined as the response strength of the following operation:

$$R = det(A) - k\, trace^2(A) \tag{3.1}$$

$$\text{where } A = \sum_{u,v} w(u,v) \begin{bmatrix} I_x^2 & IxIy \\ IxIy & I_y^2 \end{bmatrix}$$

and $w(u,v)$ is an uniform or Gaussian weighting function and the x and y subscripts indicate differentiation. A is usually referred to as the second-moment matrix of the image computed in a rectangular window around the pixel under consideration and $k \approx 0.04$ helps to distinguish between lines and corners. Finally local maximum supression in a 3×3 neighborhood is applied. Despite Harris' efficiency and ability to compute and localizing corners well under rotation (see Figure 3.1), it has been critisized for its poor repeatability under viewpoint and scale changes [19]. To overcome this, several extensions of Harris detector have been proposed, one of the most successful ones being Harris-Laplace [25] and Harris-Affine detectors [20]. The Harris-Laplace operator introduces scale invariance by using the Harris function (see equation 3.1) to localize points in each level of the scale-space representation and then selecting points for which the Laplacian-of-Gaussian attains a maximum over scale, allowing the detection of stable feature points across different scales and image resolutions. The Harris-Affine detector instead of detecting circular neighbourhoods, iteratively estimates elliptical affine regions in order to obtain affine invariant corners. This approach provides a better estimate of the object shape, regardless of transformations caused by viewpoint changes.

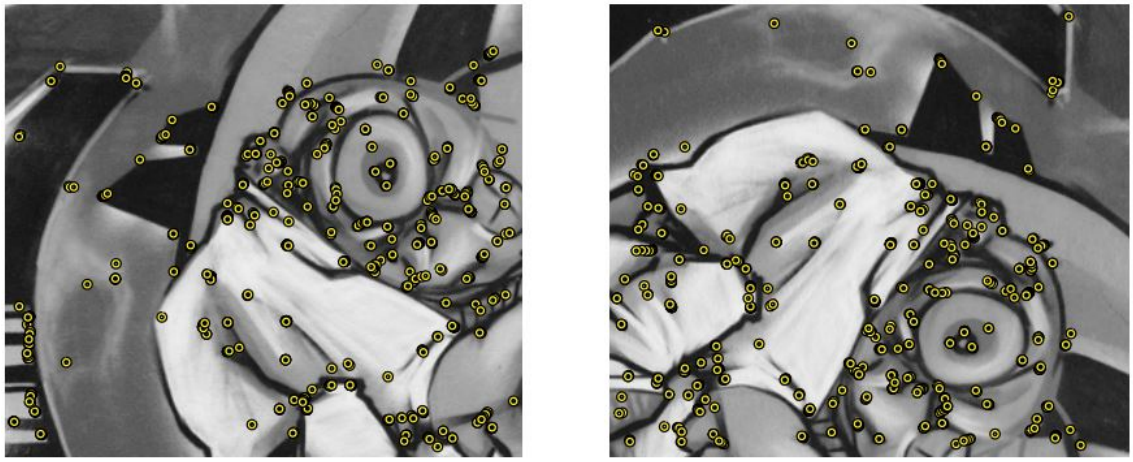


Figure 3.1: Harris corners on rotated images. Source: [2]

FAST. Features from Acellerated Segment Test (known as FAST) is an algorithm proposed originally by [3] to identify corner based keypoints. FAST was designed to be very efficient and suitable for real-time applications of any complexity. The segment test criterion operates by considering a circle of sixteen pixels around the corner candidate p. The original detector classifies p as a corner if there are a set of n contiguous pixels in a circle around it which are all brighter than the intensity I_p of the candidate pixel plus a threshold t, or all darker than I_p - t (Figure 3.2). Additionally, the authors presented a machine learning approach to create decision trees that allow FAST to classify a candidate point with only a few pixel tests, speeding-up the detection process. Althouth FAST is only a detector, it is proven to be quite reliable and used in the upstream for lots of descriptor generating processes. FAST-based keypoint detection is widely used because of its computational

properties, however, this type of features are not very robust in the presence of noise as stated by their authors [3] because of the lack of an orientation component. To address the weaknesses of FAST, several approaches have been proposed [21, 7, 8] as will be discussed below.

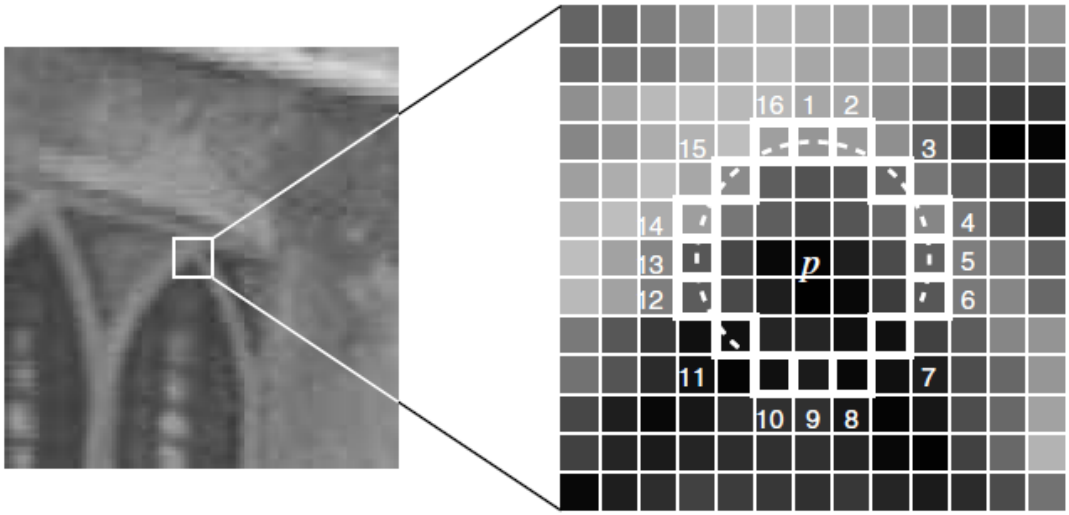


Figure 3.2: FAST detection method: pixel p and the corresponding circle of 16 pixels used to test if p is a point of interest. Source: [3]

AGAST. In the Adaptive and Generic Acellerated Segment Test [21], the performance of FAST is increased by changing the way in which the decision tree is created. Instead of using a fixed decision tree to classify a point as is used in the original idea, AGAST uses a more complex method, which is built by choosing one of the pixels to test and posing one question (whether the pixel is darker, similar or darker). The response is used to decide the next pixel and question to pose. This turns the process of searching for a corner into a process of traversing a binary tree. Consequently, the configuration space increases by the addition of two more states: "not brighter" ($\bar{b}$) and "not darker" ($\bar{d}$). Based on [3], Mair *et al.* defined the state of a pixel relative to the nucleus n, denoted by $n \to x$, is assigned as follows:

$$S_{n\to x} = \begin{cases} d, & I_{n\to x} < I_n - t & & & \text{(darker)} \\ \bar{d}, & I_{n\to x} \not< I_n - t & \wedge & S'_{n\to x} = u & \text{(not darker)} \\ s, & I_{n\to x} \not< I_n - t & \wedge & S'_{n\to x} = \bar{b} & \text{(similar)} \\ s, & I_{n\to x} \not> I_n + t & \wedge & S'_{n\to x} = \bar{d} & \text{(similar)} \\ \bar{b}, & I_{n\to x} \not> I_n + t & \wedge & S'_{n\to x} = u & \text{(not brighter)} \\ b, & I_{n\to x} > I_n + t & & & \text{(brighter)} \end{cases} \tag{3.2}$$

where $S'_{n\to x}$ is the preceding state, I is the brightness of a pixel and u means that the state is still unknown. This results in a binary tree representation, as opposed to a ternary tree, allowing a single evaluation at each node. With their experiences, Mair *et al.* showed that, in a controlled scenario, a speed-up of almost 50% can be achieved with respect to FAST.

ORB detector. A drawback of FAST and AGAST is that they do not produce scale-invariant keypoints. To address this weakness the authors of ORB [7] proposed a scale-invariant FAST extension which they called Oriented FAST (also referred as ORB detector). Since FAST does not produce multi-scale keypoints, they employed a scale pyramid of the image, detecting FAST keypoints (filtered by the Harris response, see equation 3.1) on each level of the pyramid. For example, for a target of N keypoints, they first set the threshold low enough to get more keypoints than the target N ordering them according to the Harris measure, and then, the top N keypoints are picked. The detector was also modified to estimate the orientation of a keypoint by computing a simple but effective measure of corner orientation, the *intensity centroid* [26] providing that way rotation-invariance.

BRISK detector. Similar to ORB detector, in Binary Robust Invariant Scalable Keypoints detector [8] (an AGAST extension) the scale-invariance is obtained by searching corners not only in the original image plane, but also in the scale-space. Mimicking the SIFT scale-space analysis, in BRISK, the AGAST detector is executed for each pyramid layer separately and non-maxima suppression is applied in the scale-space: the point p is classified as a corner only if its score s is greater than all its neighbouring pixels in the same layer and in the layer above and below.

3.1.2 Blob detectors

Historically, blob detectors were at first motivated by the lack of scale invariance of early corner detectors. Then the scale-space idea that is at the heart of most blob detectors developed, got back to corner detectors, spawning the scale-invariant corner detectors. Informally, a blob is a region of a digital image in which some properties are constant or vary within a prescribed range of values.

DoG One of the most-known blob detectors is the Difference of Gaussian (DoG) function. It is an approximation of a more complex and computationally expensive, but also effective blob detector, named Laplacian of Gaussian (LoG). LoG operator is defined as a trace of the Hessian matrix. Given a point $x = (x, y)$ in an image I, the Hessian matrix $H(x, \sigma)$ in x at scale σ is defined as follows:

$$H(x, \sigma = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix} \tag{3.3}$$

where Lxx is the convolution of the Gaussian second order derivative $\frac{\partial^2}{\partial x^2} g(\sigma)$ with the image I in point x, and similarly, $L_{xy}(x, \sigma)$ and $L_{yy}(x, \sigma)$. In real world applications however, an approximation of this operator - DoG - is typically preferred. Introduced by Lowe and coupled with the SIFT descriptor [27], DoG operator avoids computing second-order derivatives inherent in the LoG operator by calculating differences between Gaussian-blurred images at different scales. The authors proposed to use a scale-space pyramid where each consecutive image is a smoothed version of the preceding and the transition from one octave to the next corresponds to downsampling the image by a factor of 2. Figure 3.3 shows the process of applying the DoG operator. Generation of the image scale space is very efficient, since the pyramid is build from the bottom to the top and each consecutive image is a Gaussian smoothed version of the preceding image. Additionally, images

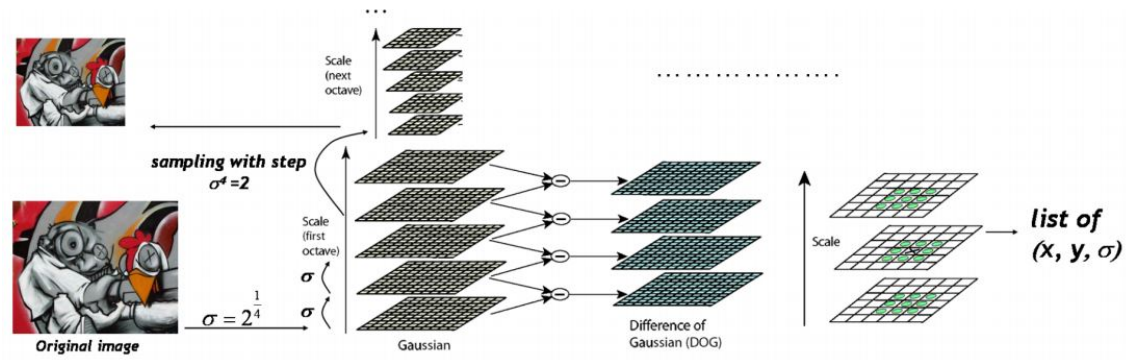


Figure 3.3: Overview of the DoG blob detection. Source: [2]

at higher octaves are subsampled versions of the original and therefore Gaussian blurring can be applied even faster. Once the scale-space representation is computed, candidate points are defined as local maxima/minima in that which are typically identified by comparing a sample point to its eight neighbors in the current image and nine neighbors in the scale above and below. The final set of keypoints is usually obtained by applying non-maximal suppression on the candidate points.

DoH The determinant of Hessian (DoH) is another blob detection method that can be derived from the Hessian matrix (see equation refeq:hessian). Instead of approximating the trace of Hessian as it is done in DoG, the DoH operator looks at its determinant. Although the performance of DoH in terms of scale selection was proven superior to this of DoG [28], its popularity can be mainly traced to the success of the SURF [4] keypoint detector which ingeniously approximates DoH using integral images[1] to reduce the computation time. To provide scale-space invariance, the DoH operator, similarly to the Harris-Laplace corner detector, should be coupled with the appropriate Laplace operator across the scale direction. Nonetheless, the SURF detector suggests using the determinant of Hessian not only in the spatial domain, but also across the scales. More precisely, Bay et al. propose to use box filters to roughly approximate Gaussian kernels as shown

[1] integral images are used as a quick and effective way of calculating the sum of values (pixel values) in a given image, allowing for the fast implementation of box type convolution filters.

in Figure 3.4 and use the resulting approximations to generate the scale-space pyramid. This step, when combined with integral images [29], leads to over five-fold speedup in comparison to the DoG operator with virtually no performance loss, since there are many more important sources of noise present in the detection pipeline.

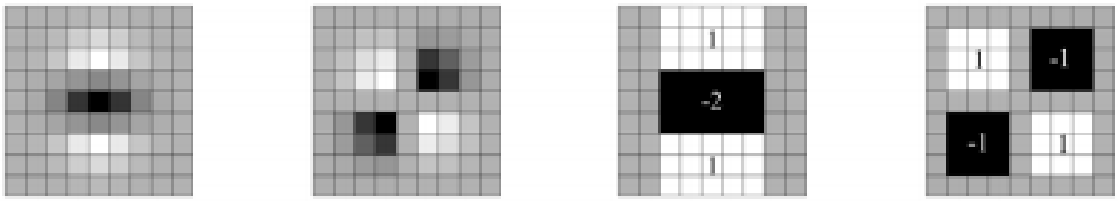


Figure 3.4: Left: discretized and cropped gaussian second order partial derivatives Lyy (y-direction) and Lxy (xy-direction). Right: the respective approximations using box filters. Source: [4]

3.2 Keypoint descriptors

The keypoint descriptors were traditionally floating-point, as they generally encode normalized difference of gradients (i.e. orientations). In the section below a first look at some of the classical ones is done and then more recently developed binary descriptors are explained.

3.2.1 Real-valued descriptors

SIFT Scale Invariant Feature Transform (SIFT) local feature descriptor was introduced by David Lowe in 1999 [27]. Although many competing algorithms have been proposed since then, SIFT still stands out as it provides a fairly stable performance regardless of the application and image registration conditions. Moreover, it has become a de facto standard descriptor for matching objects and scenes underlying various applications, such as visual search or panorama stitching. In this section the key concepts behind the SIFT descriptor are presented. Although theoretically, computation of local feature descriptors can be independent from feature detection step, Lowe proposes to build SIFT descriptors for features detected with the DoG detector, described in detail in the previous section. The rationale behind this decision is the goal of obtaining scale and rotational invariance of the resulting representation. Since DoG provides a fairly stable set of candidate feature points across different scales, the scale invariance is guaranteed for the candidate features. Orientation invariance, however, has to be provided separately. This requires an effective method of assigning a repeatable orientation to the feature points. To that end, SIFT takes the keypoints, along with the detected scale and selects from the scale-space pyramid the image $L(x,y)$ that corresponds to the closest scale to the keypoint's actual scale. For a given feature point at location (x,y) and scale s, the gradient magnitude $m(x,y)$ and orientation θ(x, y) are precomputed using pixel differences, as follows:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}, \tag{3.4}$$

$$\theta(x,y) = \arctan\left[\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right]. \tag{3.5}$$

Orientation values θ(x, y) around the feature location are used to form a 36-bin histogram which covers 360 degrees. Each sample added to the histogram is weighted by the corresponding gradient magnitude $m(x,y)$ and a Gaussian around the keypoint to smooth the noisy magnitudes estimates. Finally, peaks of the orientation histogram correspond to dominant directions of the local gradients and the highest peak is used as the keypoint's orientation. If there are other peaks above 80% of the highest, additional keypoints are created and assigned the corresponding orientations. Although this step seems redundant at first, it contributes significantly to the increased stability of SIFT.

In the last step, local gradient information is summarized into a 128-dimensional descriptor by concatenating sub-regional histograms of gradients around a keypoint as illustrated in Figure 3.5. After concatenating the histogram values in a single vector, the vector is normalized, to make it

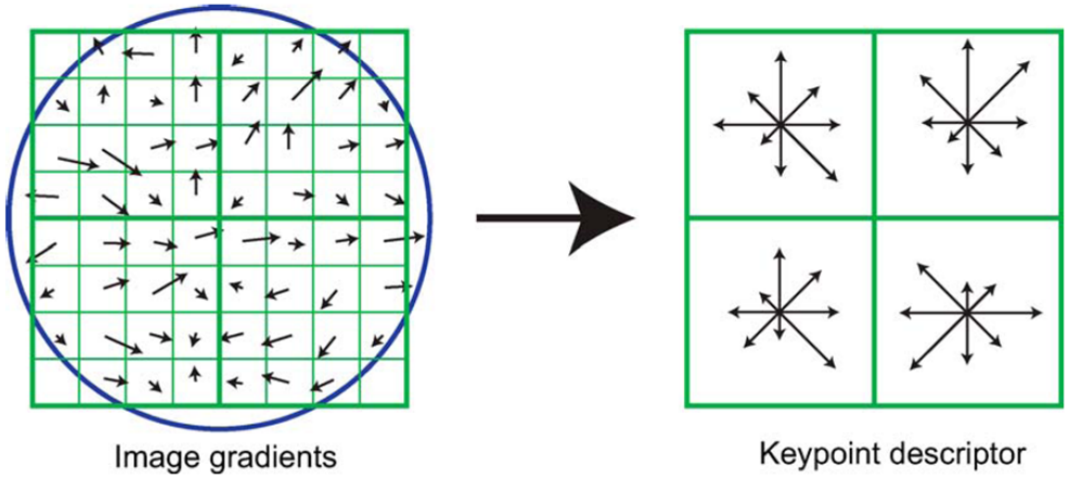


Figure 3.5: SIFT keypoint descriptor is built by first computing the gradient magnitude and orientation in a region around the keypoint location. The results are then weighted by a Gaussian window (left). These samples are subsequently summed into orientation histograms (right). The histograms are then concatenated to form a final representation. The standard SIFT descriptor uses a 16×16 sample array on the left and 4×4 histograms of size 36 on the right, resulting in a 128-dimensional vector which is ultimatly normalized. Source: [5]

more robust to illumination changes. The above procedure makes the descriptor scale and rotation invariant, because the histogram is computed at the keypoint's scale while the gradients are all rotated into a canonical orientation depending on the keypoint's orientation.

SURF Another well known algorithm for keypoint description is the Speeded-Up Robust Features (SURF) [4] descriptor. It relies on integral images to approximate convolutions, which provides an appreciable improvement in efficiency (compared to SIFT for example). The descriptor extraction algorithm of SURF works as follows:

- A weighting Gaussian function centered at the keypoint is applied.
- The general orientation is computed using sums of Haar wavelet filter responses using integral images to speed up the process.
- The patch is divided to form a square grid (oriented in the direction of the general orientation computed in step (2)). The grid consists of 16 cells (called subregions).
- Orientations inside each cell is computed using Haar wavelet filter responses.
- A vector of 4 different sums of the orientations is computed in each cell.
- The 64 (16 x 4) values are normalized and form the SURF descriptor.

Variants such as SURF-36 (3x3 grid) or SURF-128 (extended set of features) are based on the same method. Applications using the SURF descriptor shown that it provides invariance to illumination,

viewpoint and contrast variations. The standard SURF descriptor is a 64-dimensional real valued vector.

3.2.2 Binary descriptors

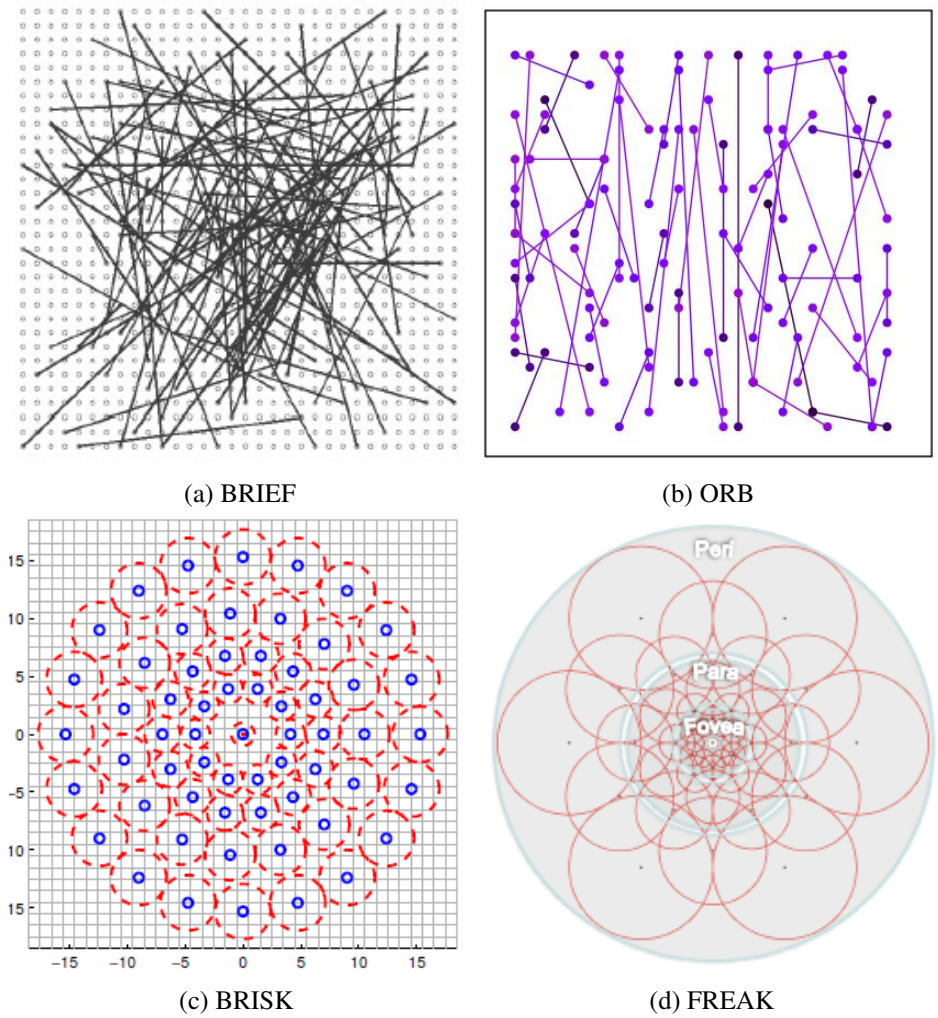
Binary descriptors were inspired by previous works [30, 31] that showed that image patches could be effectively classified on the basis of a relatively small number of pairwise intensity comparisons [6], arising from the need to have features computed quickly and are compact in their representation [32]. In addiction, due to their binary nature, the Hamming distance can be used as similarity measure which is also very efficient to compute. These features make them an attractive solution for many modern applications, especially for mobile platform where both the computing and memory resources are limited. The existing binary descriptors are fitted with common basic properties:

- the descriptors are built from a set of pairwise intensity comparisons being each bit in the descriptors exactly the result of a comparison.
- a fixed sampling pattern is used (except for a possible scaling and rotation).
- the Hamming distance is used as similarity measure.

Although the binary descriptors present common points, each of them has unique properties to achieve their design goals. Recently the number of efforts (or approaches) to provide methods to directly calculate binary descriptors from local image patches has increased given the aforementioned advantages of this type of descriptors. Below some of the main differences between binary descriptors are detailed.

BRIEF The Binary Robust Independent Elementary Features (BRIEF) descriptor proposed by [6] is the simplest binary descriptor but is a good example that meets the common principles set out above. In [6] several ways of selecting the set of sampling points were presented (Figure 3.6a) but the authors suggest the use an isotropic Gaussian distribution centered at the feature location as sampling pattern as it is more advantageous. It is important to mention that generally the BRIEF descriptor can take three different sizes depending on the number of selected points (eg. 128, 256, 512). By itself BRIEF is neither scale nor rotation invariant which opened the door to the authors of [7] who proposed an extension of BRIEF providing rotation invariance and more robustness to noise, which will be explained below.

ORB The Oriented FAST and Rotated BRIEF (ORB) descriptor, proposed in [7], came as a faster and more efficient alternative to common real valued descriptors such as SIFT and SURF and overcoming the lack of rotation invariance of BRIEF (as shown in Figure 3.7). In contrast to what happens in BRIEF descriptor, the sampling pattern employed in ORB always uses 256 pairwise intensity comparisons applying machine learning techniques to maximize the descriptor's variance



(a) BRIEF (b) ORB

(c) BRISK (d) FREAK

Figure 3.6: Sampling schemes for the different binary descriptors. Sources: [6, 7, 8, 9]

and to minimize the correlation under orientation changes getting the optimal set of sampling pairs (as can be seen in Figure 3.6b).

BRISK The Binary Robust Invariant Scalable Keypoints (BRISK) descriptor proposed by Leutenegger *et al.* [8] as a dramatically faster alternative to SIFT and SURF with comparable matching performance. To describe the features, the authors turn away from the random or learned patterns of BRIEF and ORB, and instead use a symmetric sampling pattern. Sample points are positioned in concentric circles surrounding the keypoint, with each sample point representing a Gaussian blurring of its surrounding pixels. The standard deviation of this blurring is increased with the distance from the center of the feature (Figure 3.6c). In order to compute the global orientation of the keypoint, BRISK uses a set of long range pairs. The sampling pattern is then rotated in the direction of the estimated orientation and the descriptor built using a deterministic set of 512 short range pairs.

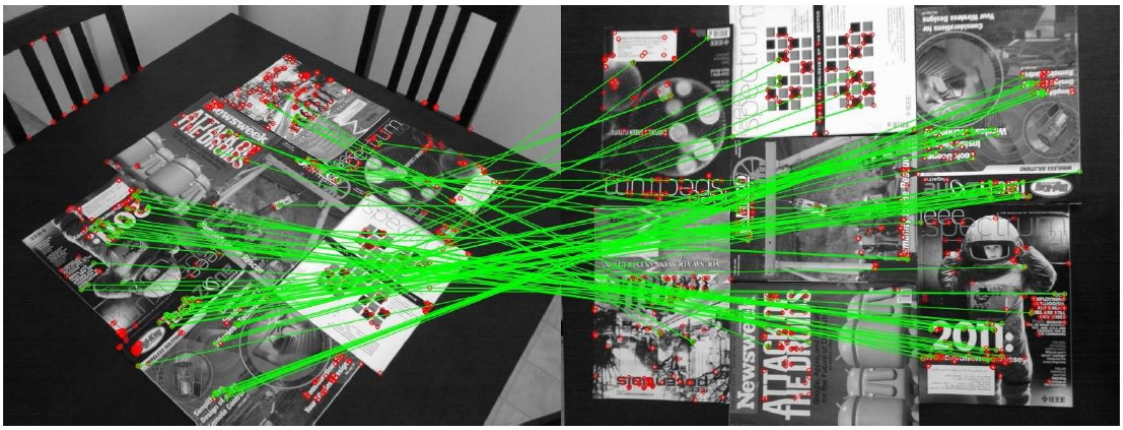


Figure 3.7: Image matching done using Oriented FAST and rotated BRIEF. Green lines denote matches, while red circles denote unmatched points in both images. In this case, there is a visible viewpoint variation. Source: [7]

FREAK The Fast Retina Keypoint (FREAK) descriptor proposed by [9] also provides scale and rotation invariance, however its pattern is based on Gaussians. As its name suggests, it is biologically inspired on the retinal pattern of the human eye. In practice, FREAK improves upon the sampling pattern and method of pair selection that BRISK uses. Thus, for computing this descriptor, 43 weighted Gaussians at locations around the keypoint are evaluated. As can be seen in Figure 3.6d, overlappings are considered in order to compute average values related to some points. Moreover, the patterns are much more concentrated near the keypoint that leads to a more accurate description of the keypoint. To speed up the matching process, the actual FREAK algorithm also uses a cascade for comparing these pairs, and selects the 64 most important bits in the beginning of the descriptor. Just like BRISK, FREAK leads to a 512 bit binary descriptor.

3.3 Performance evaluation

The recently growing popularity of keypoint detectors and descriptors has been followed also by a growing interest in exploring the performance of the various options available in this field since keypoint matching performance is extremely important for many computer vision areas. Initially [20] proposed a new approach to detect affine and scale invariant keypoints and presented a comparative evaluation of different keypoint detectors using the repeatability criterion introduced by [33] to evaluate the stability and accuracy of detectors. They also provided a keypoint detector evaluation dataset to test the effect of blur, illumination, scale/rotation, and perspective changes in interest point detectors which was used in this project to perform some groundtruth tests. The following year, [10] presented an evaluation , this time comparing the performance of different local descriptors in the presence of real geometric and photometric transformations and introducing two important metrics: recall and 1-precision. Recently [32] performed a comparative evaluation

of binary descriptors (eg. BRIEF, ORB and BRISK) analyzing the performance of several detector and descriptor pairings. They concluded that a binary descriptor will suffer in performance when it takes into account a transform not present in the data and that both detector and descriptor should be invariant to the same set of transforms. Later [34] extended Heinly et al. evaluation by including the latest binary descriptor, FREAK.

Chapter 4

Proposed system for content-based image retrieval

This chapter presents implementation details for the robust and efficient visual search system for fashion magazine recognition and retrieval. The logical architecture, and the main algorithms are described using pseudo-code and diagrams. At the end of the chapter, some conclusions are drawn.

4.1 Logical architecture

As a content-based image retrieval system implementation the typical architecture is followed with some changes to achieve the specific objectives. This system can then be divided into three different phases:

- **Vocabulary generation**
- **Database building**
- **Query image retrieval**

These steps can also be defined as offline and online steps. First, in the offline steps a vocabulary is constructed from a set of images of fashion print magazines. After that, an *image signature* is created for all the magazine pages to be introduced in the database. In the online step it is thus necessary to create - accordingly - an *image signature* to the query entered by the user (*i.e.* using the same methods as for the database pages) and search for the nearest images. All steps are sequential and dependant on previous steps. They constitute the logical architecture of the developed system and will be detailed below.

Vocabulary generation

1. Sample images
2. Feature extraction

3. Clustering

A great number of keypoint descriptors extracted from a large corpus of representative images from print fashion magazines are used to populate the keypoint descriptor space with keypoint descriptor instances. Next the sampled keypoint descriptors are clustered in order to quantize the space into a discrete number of k visual words. This pipeline is illustrated in Figure 4.1.

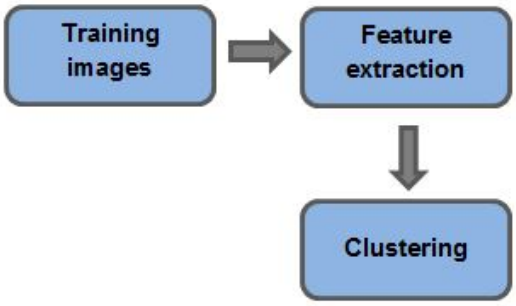


Figure 4.1: Representation of the vocabulary generation pipeline.

Database building

1. Database images
2. Feature Extraction
3. Quantization
4. Inverted Index

After generating the vocabulary to be used, a database containing the fashion print magazines to be later available for the actual visual search, is created. To do this, it is necessary to generate an image signature for each image page of the fashion print magazines. After that the image signatures are indexed efficiently using an inverted index strategy. Figure 4.2 shows the main steps of this pipeline.

Query image retrieval

1. Query image
2. Feature Extraction
3. Quantization
4. Index search
5. Re-ranking

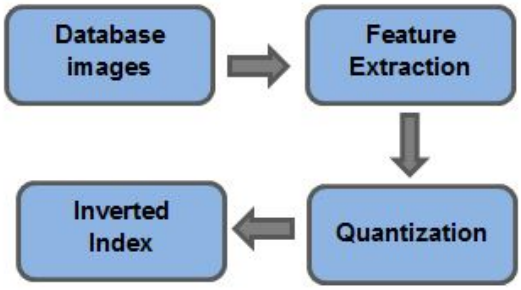


Figure 4.2: Representation of the pipeline used for building the database.

6. Validation

The last and most important step is where the interaction between the system and the user is made. When a user submits a query image in the system, an image signature is created (using the same methods described in the previous step) and then a similarity search is performed using the efficient inverted index created in the previous step. Through experimental tests, it was verified that the correct match is almost always within the group of best rated pages (*top N* ranked pages) despite not always be the best rated page (*top 1*) . Thus, we analyze a set of top ranked pages and a brute force matching is applied on the descriptors of these pages (that would be computationally prohibitive to use for all pages, but using a relatively low number N can be computationally affordable). Figure 4.3 shows the pipeline used in this step.

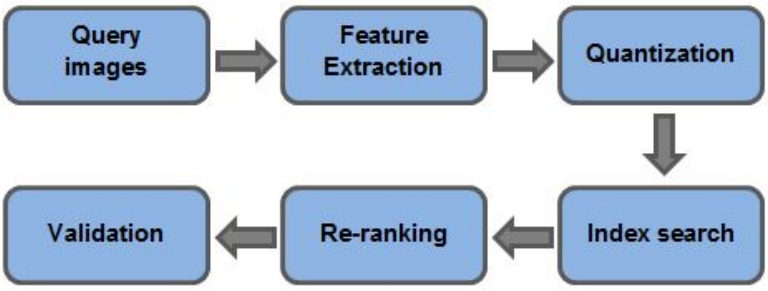


Figure 4.3: Representation of the pipeline used in query retrieval.

4.2 Solution approach

In this section, the main parts of the application are discussed. A bag of keypoints algorithm was implemented in two different versions: one baseline version and a naive approach using binary keypoint descriptors. For the baseline approach, SIFT was used for the keypoint detection and description, then clustering the floating-point vectors using *k-means* to generate the vocabulary with the *k-means++* initialization. For the naive binary approach ORB detector was used, coupled

with different descriptors and the clustering method used for this case is a k-means extension designed to cluster binary data which will be explained in the next section together with other fundamental steps of this work.

4.2.1 Vocabulary generation in the binary space

After the feature extraction procedure already explained, keypoint descriptors require an additional quantization step to be transformed into global and fixed-lenght representations of the image, an *image signature*. The classic approach is to employ k-means clustering using Euclidean distance between feature vectors, and this has proved to be effective, even if computationally demanding during the vocabulary generation with the sampling keypoint descriptors. Unfortunately when dealing with a vector of binary descriptors, Euclidean distance is not the metric of choice, because averaging binary vectors is undefined. Therefore, the notion of geometry mean must be redifined for data of this nature. The closest equivalent to the geometric mean in the binary space is to take the component-wise median, which is equivalent to maximum voting for each component. In the assignment step, data samples are assigned to their closest cluster using Euclidean distance. For this step, it is sufficient to replace the metric by the Hamming distance. In the update step, the centroid is computed using the geometric mean of all points currently associated to the cluster.

[35] proposed an approach for using binary keypoint descriptors as input to the bag of keypoints model called *k-majority* which was used as basis to generate the binary visual vocabulary, presented in the following pseudo-code:

Algorithm 1 K-majority algorithm

1: Given a collection D of binary vectors
2: Randomly generate k binary centroids C
3: **while** centroids not changed **do**
4: **for** $d \in D$ **do** ▷ Assign data to centroids
5: $c_d \leftarrow \arg\min_{c \in C} HammingDistance(c,d)$
6: **end for**
7: **for** $c \in C$ **do** ▷ Majority voting
8: **for** $d \in D|_{c_d=c}$ **do**
9: v accumulates d votes
10: **end for**
11: $c' \leftarrow majority(v)$
12: **end for**
13: **end while**

The algorithm presented takes as input a set of D binary vectors and seeks for a number k of centroids, that will become the visual vocabulary for the bag of keypoints model. Initially, these k centroids are initialized randomly (line 2). On each iteration, each binary vector is assigned to the closest centroid (lines 4-6). The next step (lines 7-12) is the majority voting step: for each cluster c, every binary vector d belonging to it is taken into consideration. Every bit of an accumulator vector v is increased by 1 if the corresponding bits in d is 1. At the end, the majority rule is used to

form the new centroid c': for each element v_i in v, if the majority of vectors voted for 1 as bit value in v_i, then c'_i takes 1, otherwise c'_i takes 0. The algorithm iterates until no centroids are changed during the previous iteration.

The voting scheme, illustrated in Figure 4.4, operates as follows: 1) given a set of binary vectors assigned to some cluster C, they are bitwise accumulated into an accumulator v such that each of its elements represents the count of bits on that position set to 1, 2) to compute the cluster centroid the majority rule is applied for each element *vi* in *v*.

$$c_i = \begin{cases} 1 & if \quad v_i > \frac{n_d}{2} \\ 0 & otherwise \end{cases}$$

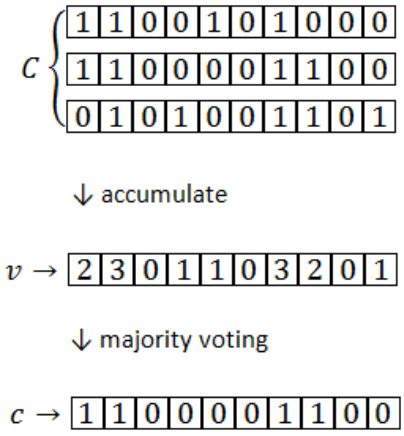


Figure 4.4: Example of the voting scheme step used in k-majority to compute centroids from clusters of binary data. Note: n_d is the number of keypoint descriptors assigned to a cluster C.

4.2.2 Quantization

For each image the nearest visual word in the vocabulary is identified for each keypoint descriptor extracted. This maps the image from a set of descriptors to an histogram (where each component is the respective word number). This bag of keypoints histogram of visual word occurences is then used to summarize the image content - the *image signature*.

4.2.3 Inverted index

The motivation for the inverted index is to take advantage of the sparseness of the term vectors obtained in the quantization step. Arbitrarily if one considers that in each image there are approximately 500 keypoints detected and the vocabulary size is $5 * 10^3$ terms then the resulting term

vector will have at most 10% of its entries as non-zero entries. This propriety can be exploited computationally, by neglecting the zero entries.

A simple approach to ranking query results is to compute the distance between the query vector and the term vectors of each gallery image, requiring an $O(N)$ computation in term of the number of gallery images. The majority of those images are likely to have very few terms in common due to the sparsity of the vectors, however. Instead, one can look at each non-zero term in the query vector and get the list of images in which that term appears via the inverted index. While the worst case complexity remains $O(N)$, there is a huge speed improvement for the average case, compared to the naive-approach of computing the distance between the query vector and the term vectors of each gallery image. Figure 4.5 shows an illustrative example of the inverted index implemented.

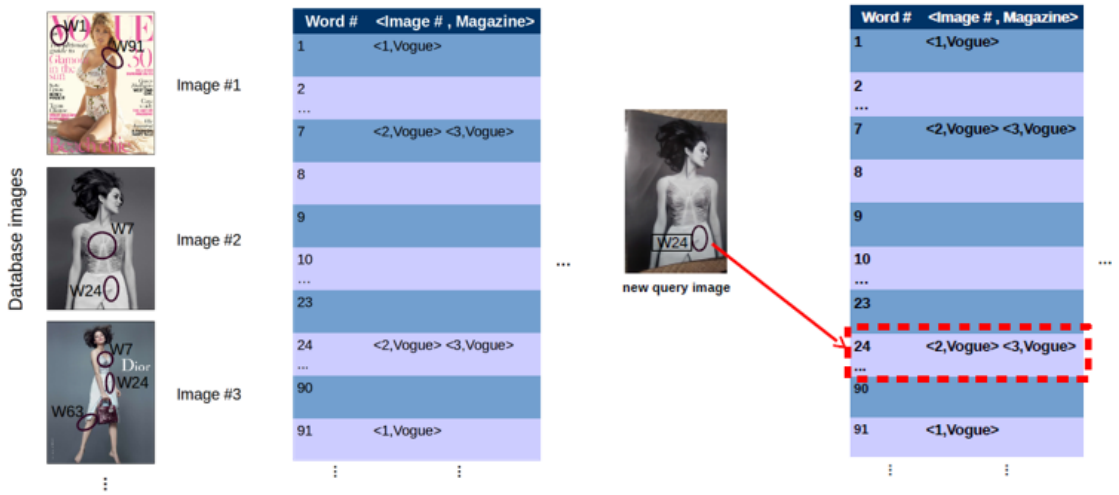


Figure 4.5: Illustrative example of the efficient indexing structure developed. Left: All database images are loaded into the index mapping visual words to image numbers. Right: A new query image is mapped to indices of database images that share a visual word.

4.2.4 Re-ranking and validation

Since the inverted index maintains no information about the relative spatial layout of the words per image, typically a spatial verification step is performed on the images retrieved for a given query. However, as this step is computationally heavy it was decided to compare the keypoint descriptors of the top N retrieved images with the keypoint descriptors of the query image using the ratio threshold test described in 2.2.3 performing that way a re-ranking of the top N retrieved images. To avoid the system to return false matches, a validation step had to be included. This was again carried out by using a ratio test, defined as:

$$\tau > \frac{m_{top \leftrightarrow query}}{m_{2nd \leftrightarrow query}} \tag{4.1}$$

where $m_{top \leftrightarrow query}$ represents the number of correct matched keypoints (matched keypoints which pass the ratio threshold test) between the query image and the top ranked page in the re-ranking step and similarly $m_{2nd \leftrightarrow query}$ refers to the number of correct matched keypoints between the query image and the second best rated image. τ can be tuned according to the degree of confidence that the system aims to achieve.

4.3 Final remarks

A custom bag of keypoints implementation using binary keypoints was described. In this method, after quantizing each binary keypoint descriptor for both the input image and the database images into visual words, the number of occurrences of all visual words in each image is counted to build all frequency histograms. After that, an inverted index was built to ensure efficient search during the query time retrieval.

Chapter 5

Experimental Results

In this chapter are presented the performance evaluating of the image retrieval pipeline proposed in this thesis.

By undertaking the evaluation of this algorithm, the main goal was to answer questions such as, *how does varying pipeline modules affect system behaviour?*, *how does vocabulary size influence retrieval performance?*, *Is it possible to generate a perfect vocabulary for a certain task?* To address such questions, in this section will be explained the environment setup, the datasets, the tests performed and the respective results.

5.1 Equipment and software used

Hardware
In proposed system the following development platforms were used:

Laptop computer i7-4500U

- **CPU:** Intel Core i7 4500, 4 cores @ 1.8 GHz
- **Memory:** 8GB RAM @ 1600 MHz
- **Base system:** Ubuntu 14.04 LTS 64bit

Software
The following software and libraries were used:

- GCC 4.8.2 with glibc 2.19
- Qt Creator 3.2.0
- OpenCV library 2.4.9

5.2 Dataset

The Fashion Print Magazine dataset was built for this thesis from a 264 pages fashion magazine - Vogue UK (June 2014). Figure 5.1 shows some images from the fashion magazine used to create the Fashion Magazine dataset.

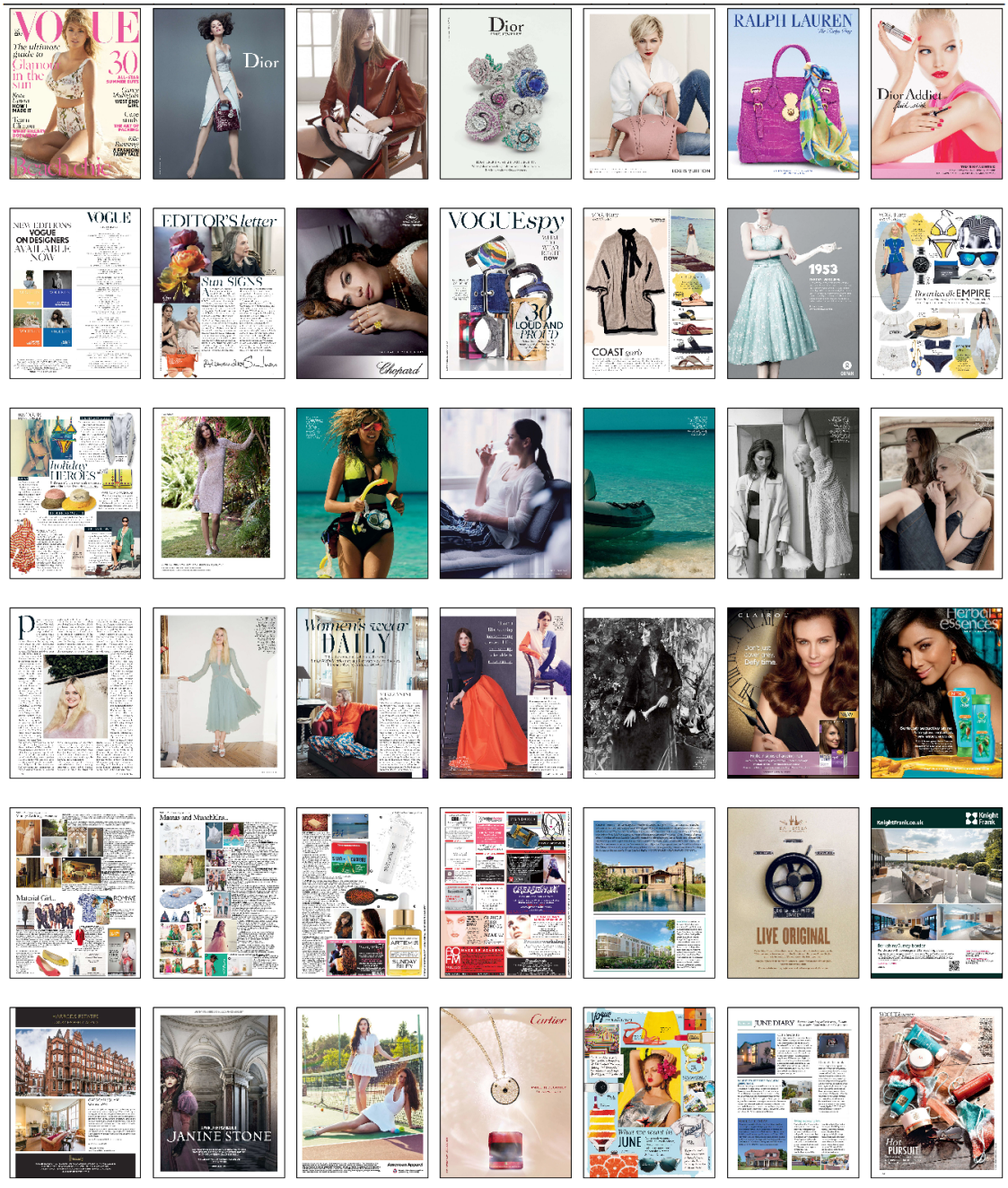


Figure 5.1: Sample pages from the Vogue UK (June, 2014) fashion magazine used to create the dataset of photos taken from this magazine.

For each image fashion magazine were taken 5 input photographs each with different point of view, different perspective or imaging conditions. Thus is defined five different types of query:

- **type 1** The purpose of this type of query was having a query image nearest of the "ideal query", ie, a picture taken with care, to capture the full page of the magazine and with minimal conditions that could affect the matching accuracy.

- **type 2** In this type of queries pages from the magazine were taken with changes in rotation and point of view.

- **type 3** These types of query image should characterize the photographs taken by most users containing moderate changes of perspective and point of view

- **type 4** This type of query images are displayed with a high variance of perspective and viewpoint variation.

- **type 5** The most problematic queries widely varying lighting conditions, perspective distortion, introducing background clutter and partial occlusions.

In Figure 5.2 is showed some images of the created dataset. Note that all the photos were taken from a smartphone so that the images captured could be comparable to a normal situation in which a user takes a picture and introduce it in the system. The different types of query generated are used to test the system with different input images, which in turn have different difficulties associated with the recognition.

5.3 Keypoint detectors and descriptors evaluation

This section presents a performance evaluation of several keypoint detectors and descriptor. All tests were executed on a single core. Those introduced in Chapter 3 were used, with exception of AGAST that has no OpenCV implementation, and BRISK detector that features some design flaws[1] in OpenCV 2.4.9 and, due to this, was discarded in the experiments. The processing time as well as keypoint repeatabily were evaluated in different image transformations approaches. Then, Section 5.3 focus on local descriptors performance evaluation, comparing both the real-valued standard descriptor SIFT with the binary descriptors (all those mentioned in Section 3.2 from Chapter 3) in terms of computational complexity and matching accuracy. The main objective of this evaluation was to investigate whether it is possible to build an efficient and robust solution to different forms of noise using binary descriptors.

[1] http://code.opencv.org/issues/3976

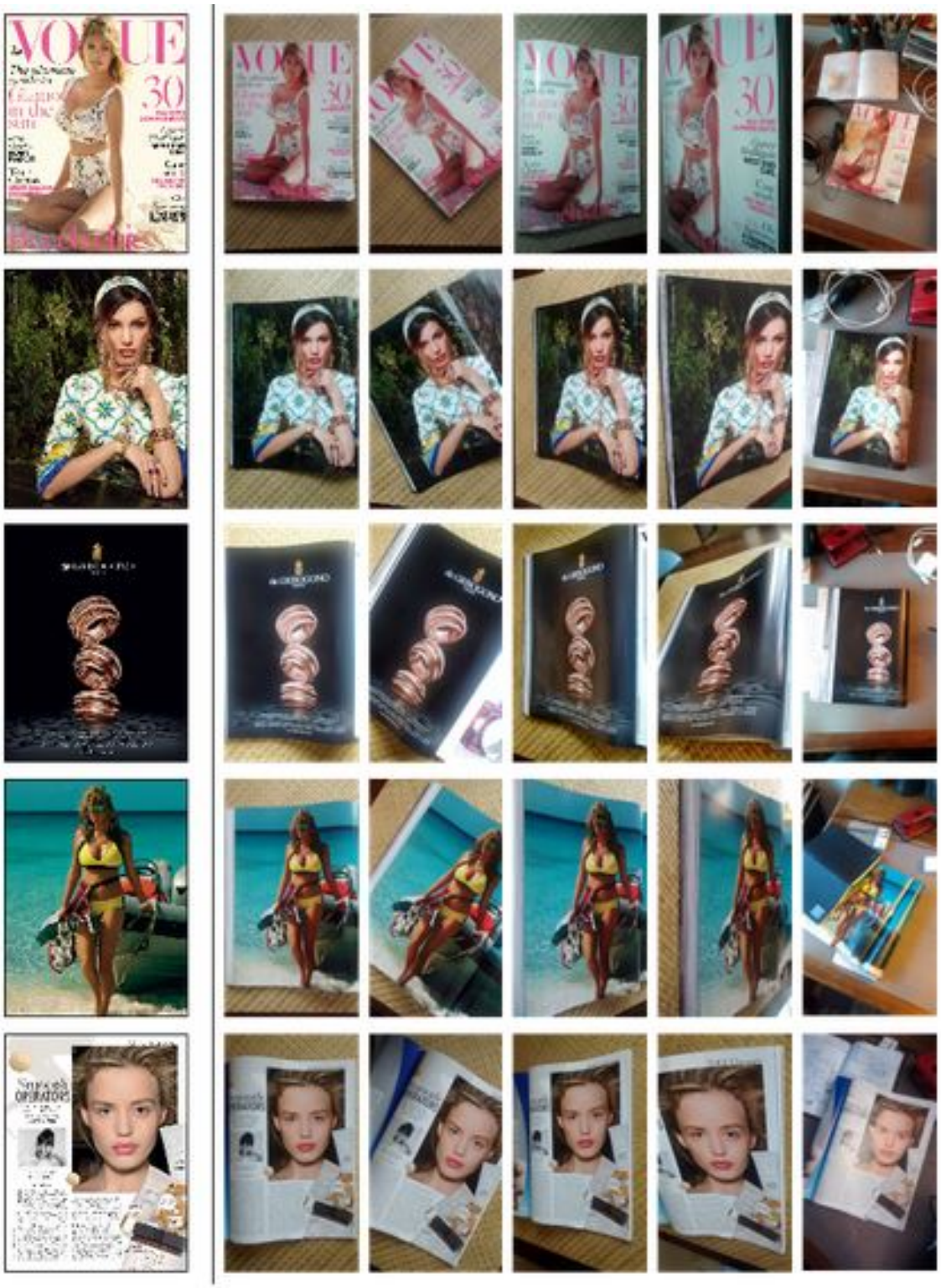


Figure 5.2: Set of images taken from the Fashion Magazine dataset. In the left column is shown the original magazine image and in the right column the respective query images. It can be seen a great variety of conditions in the query images, for example, different perspectives and scales, rotations, different lighting conditions, etc.

Detector evaluation

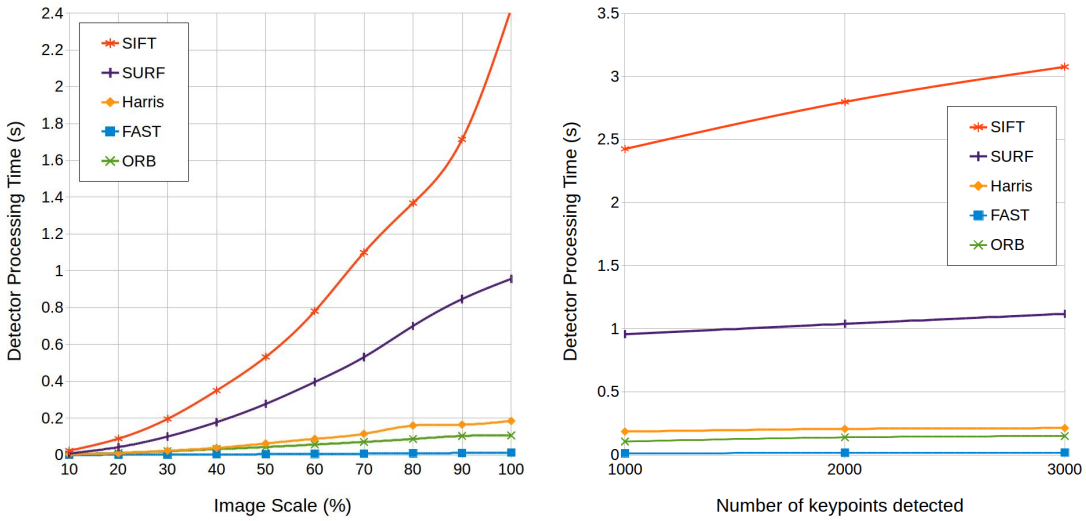
Processing time As a first experiment, the keypoint detection processing time was evaluated. In general, the processing time depends on two factors: i) the number N of detected keypoints, which can be varied by suitably tuning the detection threshold of each detector; and ii) the size of the input image. For the evaluation, 5 images (all with size of 1024×768 pixels) of each of the 17 categories from The Oxford Buildings Dataset[2] were selected randomly forming a subset of 85 images. Each image was progressively resized from its original size to a minimum size (10% of the original size). Table 5.1 shows the times obtained for the feature detection step in an image of size 1024×768 pixels. It is also shown a speed gain comparison for all the detectors using

Table 5.1: Average processing time for the feature detection in an image of size 1024×768 pixels (N = 1000 keypoints).

Detector	Proc. Time	Speed gain
SIFT	2.42 s	1.0
SURF	0.96 s	2.5
Harris	0.18 s	13.1
FAST	0.01 s	199.2
ORB	0.10 s	22.9

SIFT detector as a standard. The speed gain $SG_D = T_{SIFT}/T_D$ is used to make a comparison of the processing time of the standard SIFT detector with faster detectors, where T_{SIFT} is the processing time for SIFT and T_D is the processing time for the other detectors. It is noted that FAST reaches a speed increase in the order of *200 times* faster than SIFT and approximately *80 times* faster than SURF. ORB, the second best rated can detect the same number of features in less *23 times* of the time it takes SIFT and is also nearly *10 times* faster than the SURF detector. In Figure 5.3a the average processing time is shown as a function of the image size. In this case, the number of detected keypoints is kept fixed to $N = 1000$. For all the detectors, the processing time is a quadratic function of the image size, which means that complexity of detection increases linearly with the spatial resolution. The fastest algorithms are FAST and ORB, while Harris, exhibits slightly worse performance. On the other hand, the processing time of SURF and SIFT detectors are noticeably higher than those of all the other detectors. Then, the average processing time as a function of the number of keypoints detected is shown in Figure 5.3. For all the detectors the processing time T increases with the number of keypoints N. However, it is noted that FAST and ORB present a wide proportional decline (approximately 25%, comparing the $N = 1000$ to $N = 2000$ leap with $N = 2000$ to $N = 3000$) which suggests that the time may be constant for a greater number of keypoints. Furthermore, these two detectors were again the most efficient.

[2] http://www.robots.ox.ac.uk/ vgg/data/oxbuildings/



(a) Image scale test

(b) Number of keypoints test.

Figure 5.3: Feature detection computational complexity results for the two experiments made.

Repeatability The detectors are also evaluated with the repeatibility metric, defined in as the ratio between the number of point-to-point correspondences and the minimum number of keypoints detected in a given pair of images. For the evaluation the dataset provided by Mikolajczyk[3] was used. Each sequence includes a reference image and a set of images that are progressively modified by one or more geometric or photometric transformations. In Figure 5.4 are shown the sequences used in this test. The selected sequences are the ones that best represent the changes expected in the real world:

- Graffiti (6 images with increasing view point angle);
- Boat (6 images with different zoom and rotation angles);
- Bikes (6 images with increasing blur);
- Cars (6 images with decreasing illumination).

To match correspondent regions in the image pairs, groundtruth homography matrices are provided. Following the same approach as in [33], two keypoints were considered as matching if: i) their relative position error with respect to the groundtruth is less than 1.5 pixels; ii) their corresponding neighbourhoods are overlapped at least by 40%. The repeatability scores for the four considered image sequences are shown in Figures 5.5a, 5.5b, 5.6a and 5.6b.

[3] http://lear.inrialpes.fr/people/mikolajczyk/Database/

(a) Graffiti sequence.

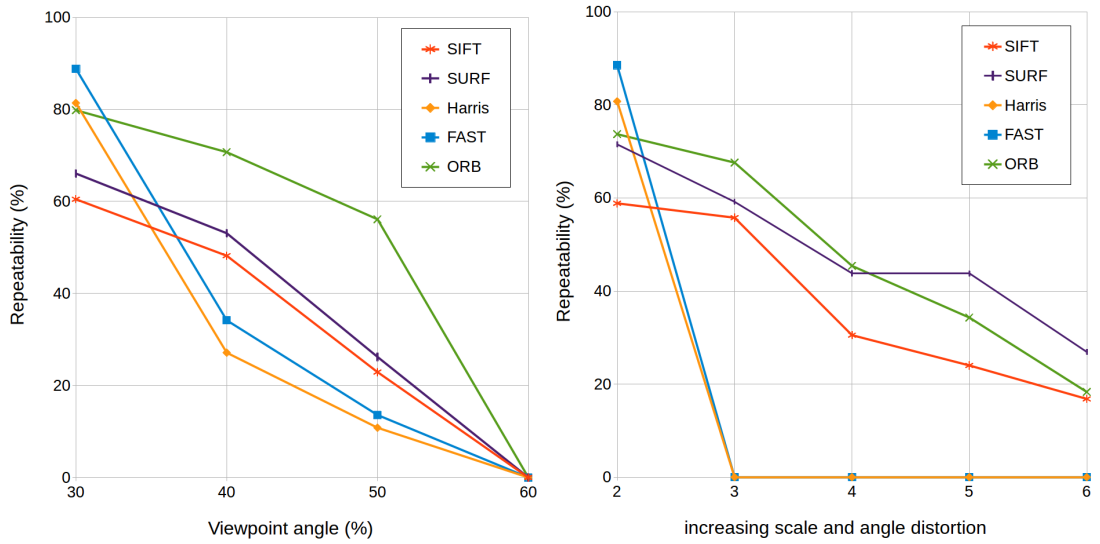
(b) Boat sequence.

(c) Bikes sequence.

(d) Cars sequence.

Figure 5.4: Keypoint detectors and descriptors evaluation sequences.

In the graffiti sequence (Figure 5.5a), the highest repeatability is achieved by ORB which surprisingly has a 70% average in the first three transformations (viewpoint angles of 30%, 40%, 50%). SURF and SIFT comes after, getting worse but acceptable performance, while the repeatability of FAST and Harris drastically decreases as the viewpoint angle increases.



(a) Graffiti (b) Boat

Figure 5.5: Repeatability score for image sequences with affine and perspective transformations (viewpoint and scale & rotation changes).

As expected, the repeatability of non-scale invariant detectors (FAST and Harris) is very poor in case of zoom transformations (see Figure 5.5b, boat sequence). Among the other detectors, ORB and SURF exhibit the best performance followed by SIFT. Note that ORB provides the best results for less accentuated transformations then being outperformed by SURF in the two most abrupt ones.

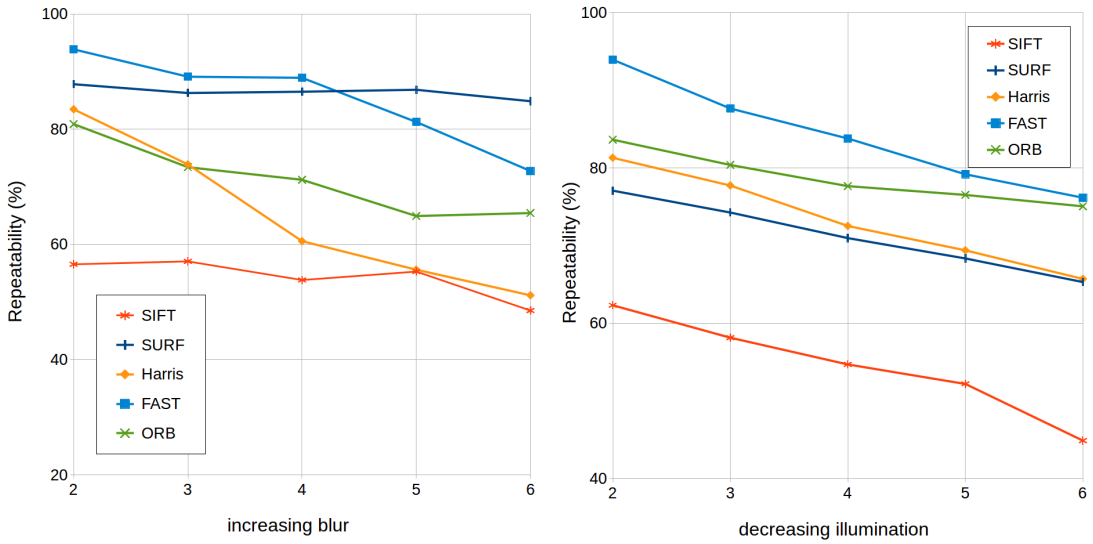
The sequence of images with increasing blur (Figure 5.6a, bikes sequence) is the one that shows better global results. Should be noted that SURF does not seem to be affected by the increase of blur since it presents constant values of repeatability for this sequence.

For the cars sequence (Figure 5.6b the best score is obtained by the FAST detector that in all decreasing illumination changes shows superior repeatability to 75%. Then appears ORB also with a very high repeatability. Although Harris and SURF detectors also offer high repeatability, SIFT performs poorly in this sequence.

Analyzing the results obtained in the comparative evaluation of the detectors there are some important points to consider:

1. FAST is the most efficient detector followed by ORB. On the other hand, SIFT is the one with higher computational complexity.

2. In all cases presented SURF performs better than SIFT which suggests that for the listed situations there is no reason to use SIFT. The same applies for the FAST detector compared



(a) Bikes (b) Cars

Figure 5.6: Repeatability score for image sequences with non-geometric transformations (in this case illumination and blur).

to Harris. One possible explanation for this is that both SURF as FAST were constructed based on SIFT and Harris (respectively), in a way to improve them.

3. ORB is the best performing detector for geometric transformations sequences followed by SURF.

4. For non-geometric transformations sequences FAST provides the best results.

5. ORB is the detector that has the best average repeatability (about 60%) closely followed by SURF.

In the Table 5.2 is also showed the average repeatability of the results for all the tested sequences of the Mikolajczyk evaluation dataset.

Table 5.2: Average repeatability (in percentage) obtained for different image sequences.

	SIFT	SURF	Harris	FAST	ORB
Graffiti	26.32	29.07	23.86	27.31	41.33
Boat	37.19	49.02	16.15	17.7	47.87
Bikes	54.44	71.19	73.34	84.15	78.66
Cars	54.25	86.46	64.94	85.18	71.18
Total	43.05	**58.94**	44.57	53.58	**59.76**

Descriptor evaluation

Processing Time Similar experiment was done for the calculation of descriptors. To make the comparison fair, the same number of keypoints extracted by SURF was used. The results are summarized in Figure 5.7.

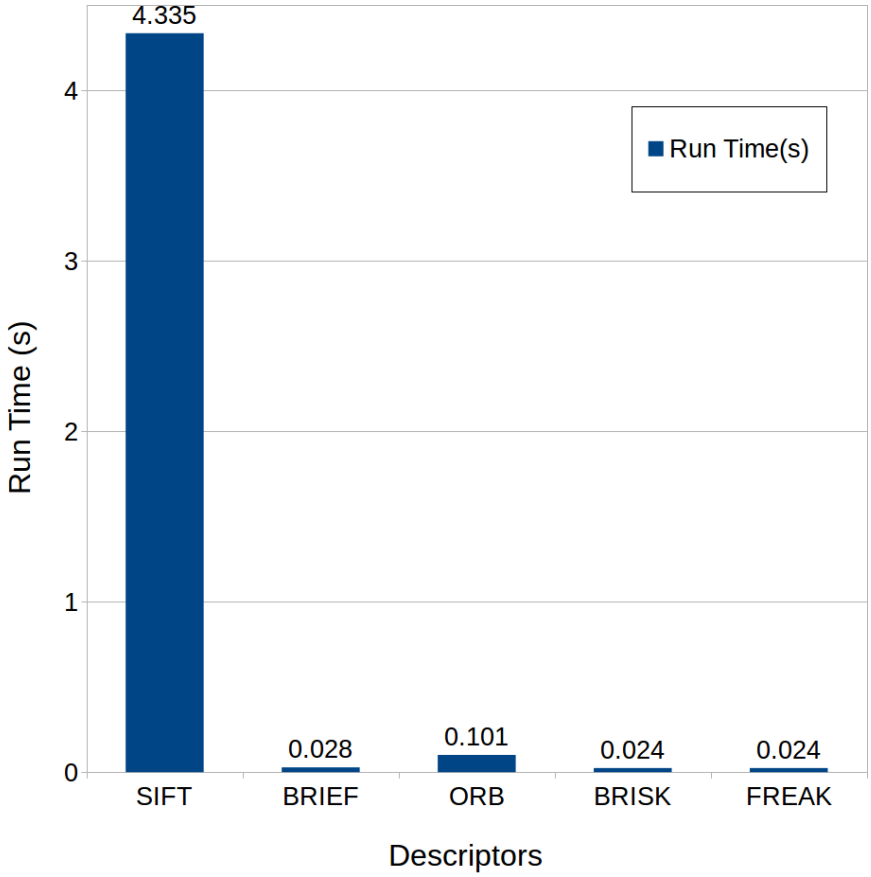


Figure 5.7: Descriptors run time using $N = 1000$ keypoints detected by SURF.

Looking at the figure presented we can conclude that the binary descriptors can be calculated much faster than a SIFT-like descriptor. BRISK and FREAK reach a 180 times speed up compared to the standard SIFT descriptor.

Matching accuracy For this test the dataset proposed by Mikolajczyk is used again. Since ORB detector presented the best repeatability rate in the detectors evaluation it is used to detect keypoint for the binary descriptors (BRIEF, ORB, BRISK, FREAK). The keypoints detected are then described using different keypoint descriptors. Then each query image is matched with the reference using the descriptors extracted. The ratio of the distance of the best match to the second best match for each point correspondence is taken and if this ratio is greater than or equal to 0.8 both of the matches will be ignored, if not the best match will be kept. The number of matches

obtained by the ratio threshold test can be considered a set of best matches between the two images tested. Figure 5.8 show the results obtained.

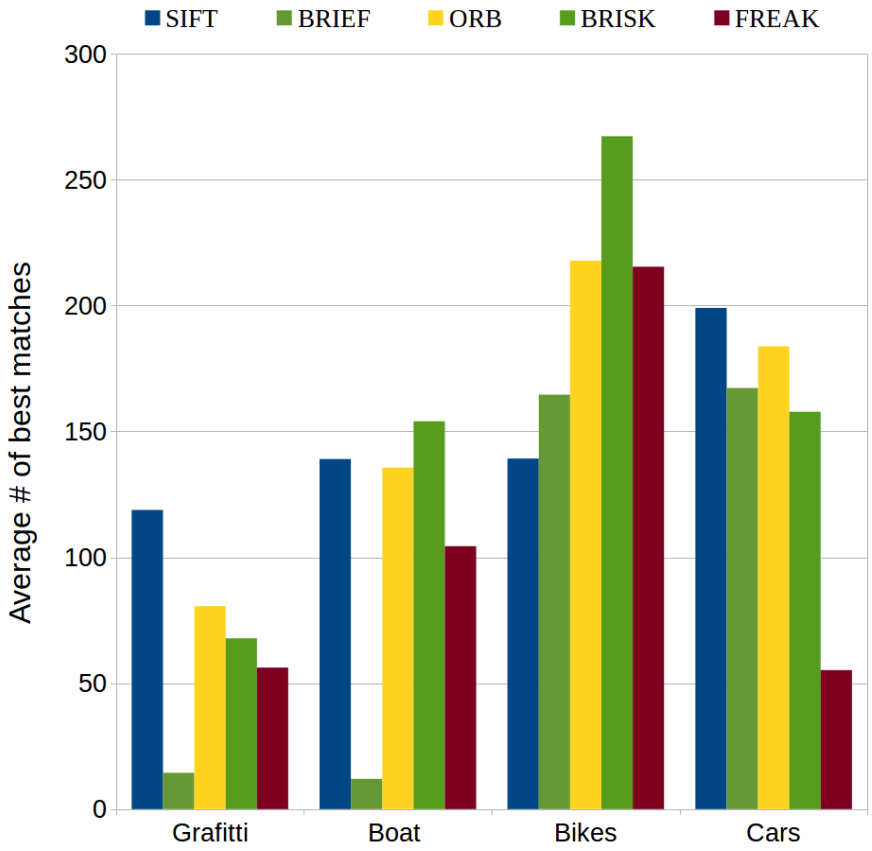


Figure 5.8: Average number of best matches in the Mikolajczyk sequence.

The results show that the most reliable descriptors in this situation for the sequences used are SIFT, ORB and BRISK. Note that BRIEF, BRISK and FREAK are not designed to describe keypoints detected by the ORB detector, however BRISK still presents acceptable results. The introduction of the BRISK detector in the test can result in significant improvements in both BRISK as FREAK.

5.4 Brute-Force system approach

The first approach was to exhaustively compare all descriptors of an input image with the descriptors of all magazine images using the Fashion Print Magazine dataset presented in the previous section. One correspondence between query and magazine page is considered correct if present the largest number of best matches. For this test ORB detector coupled with ORB descriptor and BRISK are tested and SIFT is used as a baseline. For the three scenarios 500 keypoints were detected in each image. The results are presented in Table 5.3.

Table 5.3: Correct correspondences rate for the Fashion Magazine dataset.

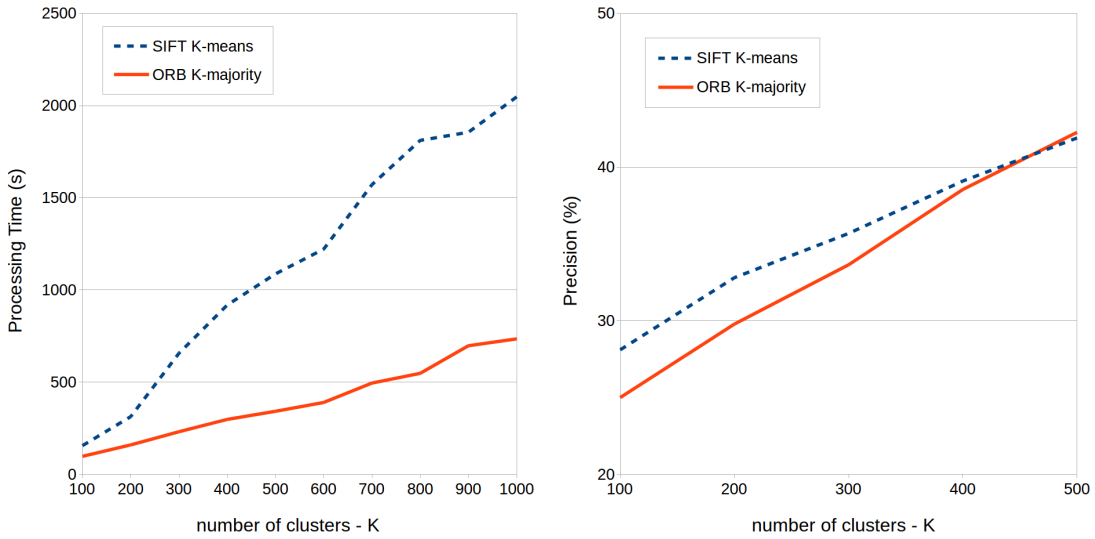
	SIFT	ORB	BRISK
Correct Correspondence Rate	79.77%	84.2424%	81.5152%

Although the results presented by the combination of ORB detector and BRISK be satisfactory, these results show that ORB provides the most reliable binary combination for this specific case. These results are very positive in terms of accuracy. However, as already explained exhaustively comparing descriptors of a query image with all images in the database is an inefficient task and must be bypassed. The main objective of this test was ascertain whether binary descriptors could be used for this specific problem effectively. The results show that the binary descriptors in addition to being faster than the standard descriptors, can achieve similar accuracies or even outperform their competitors.

5.5 Vocabulary generation

To generate the vocabulary two pipelines were implemented: the first using SIFT keypoints, clustering the keypoints descriptors using k-means with k-means++ centroid initialization. The second pipeline has been implemented in order to extend the bag of keypoints model for binary data. For this, the ORB detector and descriptor was used due to previous results which showed the most stable performance. To cluster binary keypoint descriptors, k-majority (described in Chapter 4 was used. To generate vocabularies 500 keypoints extracted from each of the 264 pages of the Vogue UK (June, 2014) fashion magazine were used which resulted in 132000 sample keypoint descriptors. After that, the keypoint descriptors were clustered using k-means for the SIFT keypoints and k-majority for the ORB keypoints. The computational complexity in generating vocabulary (presented in Figure 5.9a) and the accuracy achieved (presented in Figure 5.9b) for the 2 methods were compared. The plots are generated by varying the number of clusters (K) which is a common input element to the two algorithms.

The vocabulary generation using k-majority to cluster ORB keypoint descriptors can achieve a speed up in the order of 3 times faster comparing to the generation using the standard SIFT keypoints clustered by k-means.



(a) Processing time obtained for k-means and k-majority for different number of clusters - K.

(b) Precision obtained for different vocabularies varying the number of clusters - K.

Figure 5.9: Vocabulary generation tests.

5.6 Vocabulary size

In the previous test were compared two approaches to create vocabularies that allowed to validate the use of k-majority to cluster ORB descriptors. In order to ascertain what is the bag of binary keypoints approach accuracy for larger vocabularies, the number of clusters - K - was increased. The results are presented in Figure 5.10. The retrieval precision on the Fashion Magazine dataset as the number of clusters, K is varied is tested. Using a high number of clusters always seems to help, increasing the precision. However, the time required to quantize each keypoint descriptor also increases with the number of clusters, so it is not desirable to have an unreasonable number of clusters if system's efficiency is a concern. This suggests that, if the computation time for the quantization of an image is short (this obviously depends on the application), clustering using a larger number of clusters is a good idea because it gives better retrieval precision.

Top N Precision In order to improve the precision of the retrieval system, we decided to test if in the cases where the system does not retrieve the correct page, that page is among the best results. For this, instead of considering that the page retrieved by the system is correct only when it is the most similar, we assume that is correct if it is between the N most similar pages according to the system of ranking. As shown in Section 5.4, using a brute force approach achieve good results even if the comparison is made with all the pages of the magazine. So after selecting the top N ranked pages, we compare them using this approach. Figure 5.11 shows the results obtained. As can be seen in the displayed picture, assuming that the correct print fashion magazine page may be among the top 20 ranked pages, system precision increases by more than 20%. However, as the aim was always to choose the correct page and not the nearest set of pages, the descriptors of the top N

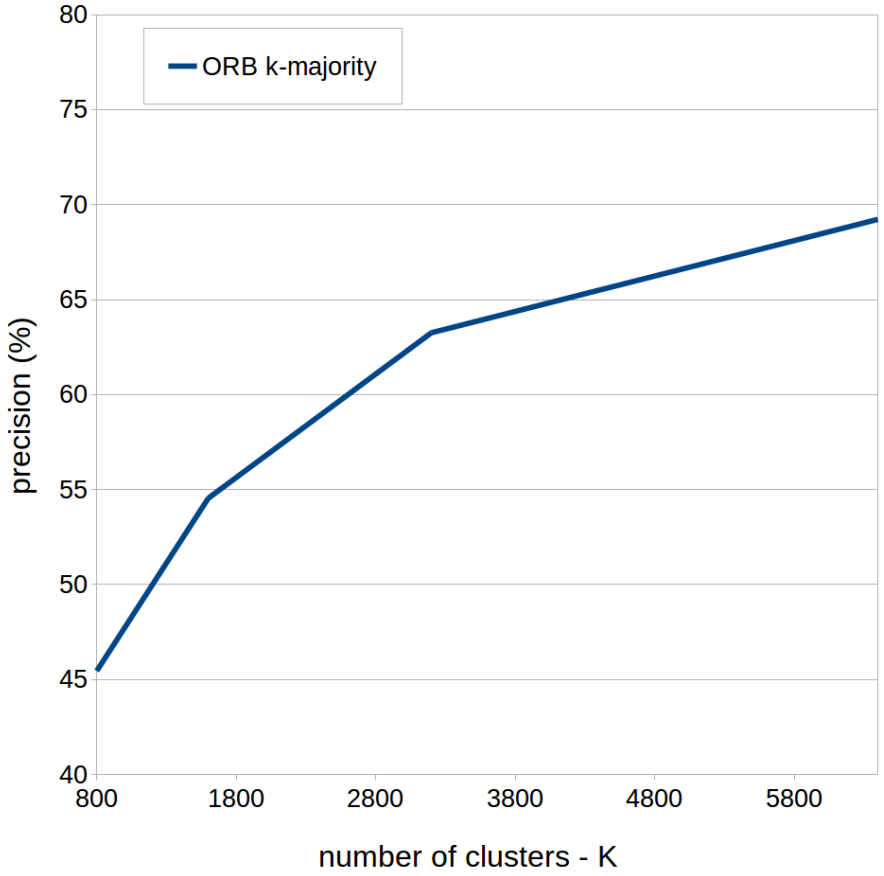


Figure 5.10: Precision values obtained for several vocabularies created with ORB keypoints clustered using k-majority.

pages were compared with the descriptors of the query image. It is then made a re-ranking of the top pages taking into account the number of best matches (matches between keypoint descriptors who pass the ratio test) of the various correspondences between the top N images and the query image. The results obtained after the re-ranking using the brute-force approach and obviously just counting the most similar page as correct match are also shown in the Figure 5.11. Note that for this test the $K = 6400$ visual vocabulary generated in the previous step was used.

5.7 Vocabulary generalization

An important concern in the visual vocabulary is the choice of data used to construct it. Generally the most accurate results are obtained when using the same data source to create the vocabulary as is going to be used for the retrieval task. However, generating a vocabulary whenever new data is indexed in the database may be inefficient. For example, to index the pages of a specific magazine, the vocabulary made from a sample of those pages would be most accurate; using a second magazine to form the vocabulary should still produce meaningful results, though likely weaker accuracy. The generalization of the retrieval system is then tested using a fashion magazine

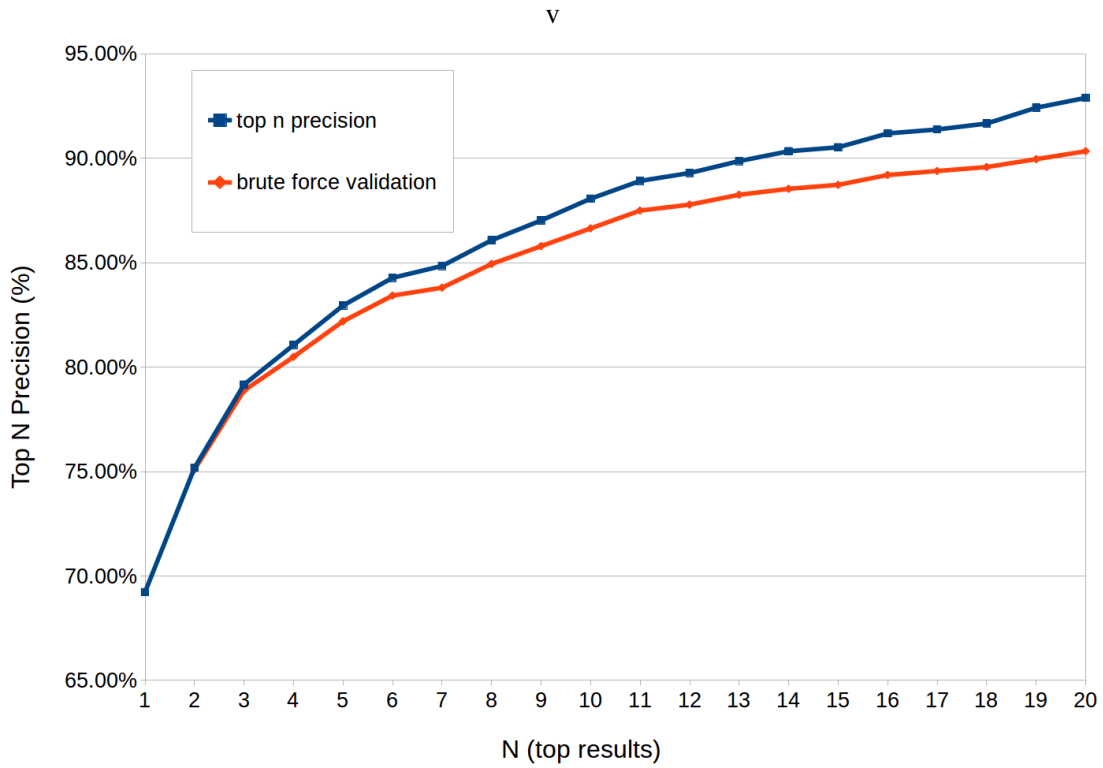


Figure 5.11: Top N precision and the validation results using the brute force approach.

to generate a vocabulary of K = 6400 visual words - Cosmopolitan UK (June, 2014). Then, image signatures are created for all the pages of Vogue UK magazine (June, 2014) using the vocabulary generated from the Cosmopolitan fashion magazine and the system precision is tested using the query images of the Fashion Magazine dataset. Results are shown in the Table 5.4

Table 5.4: Generalization test comparing the precision of the queries from the Fashion Magazine dataset (Vogue UK magazine) when using a vocabulary generated from images of the same fashion magazine with a vocabulary generated from another fashion magazine (Cosmopolitan UK).

	Vogue	Cosmopolitan
Precision	69.22%	56.44%

Although the results obtained using the the most generic vocabulary are worse, these results are more stable and can be achieved for other fashion print magazines indexed. In addition the results may be improved by using brute force re-ranking of the top *N* pages retrieved as shown in the tests performed in the previous section.

5.8 Confidence System

Until this step, the fact that the system could retrieve wrong pages of fashion magazines did not matter. However, in real world applications it is necessary to have confidence in the results returned

to the user even if it is necessary to reduce significantly the number of times the system returns what is intended. Thus, the purpose of this test was to achieve a strategy to validate the results which consists in the system retrieving only the result when there is a 100% confidence that fashion magazine page returned will be correct. Therefore there will be cases where the system despite having a page with ranking superior than all the other pages, will not have enough confidence to return it to the users. For this to be possible, it was used a threshold ratio between the number of best matches between the highest ranked page by the system and the query image and the number of best matches between the second highest rank page with the query as explained in the Section 4.2.4. To perform these tests we used the vocabulary generated from the Cosmopolitan UK fashion magazine with $K = 6400$. The results are shown in the Tables 5.5, 5.6 and 5.7.

Table 5.5: Precision obtained for different values of the confidence ratio and the top N pages used to perform the brute force.

	Ratio = 2.0	**Ratio = 2.1**	**Ratio = 2.2**	**Ratio = 2.3**	**Ratio = 2.4**	**Ratio = 2.5**
N = 5	65.44%	64.49%	63.26%	61.74%	60.13%	59.00%
N = 10	66.67%	65.44%	64.11%	62.41%	60.51%	59.09%
N = 15	67.80%	**66.10%**	**64.68%**	**62.50%**	**60.51%**	**58.71%**
N = 20	**67.90%**	**65.81%**	**35.32%**	**37.50%**	**60.42%**	**57.86%**

Table 5.6: Error percentage obtained for different values of the confidence ratio and the top N pages used to perform the brute force.

	Ratio = 2.0	**Ratio = 2.1**	**Ratio = 2.2**	**Ratio = 2.3**	**Ratio = 2.4**	**Ratio = 2.5**
N = 5	0.76%	0.57%	0.57%	0.57%	0.38%	0.38%
N = 10	0.38%	0.19%	0.09%	0.09%	0.09%	0.09%
N = 15	0.19%	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**0.00%**
N = 20	0.19%	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**0.00%**

Table 5.7: Percentage of times the system will not have the confidence to answer to the query inserted by the user.

	Ratio = 2.0	**Ratio = 2.1**	**Ratio = 2.2**	**Ratio = 2.3**	**Ratio = 2.4**	**Ratio = 2.5**
N = 5	33.81%	34.94%	36.17%	37.69%	39.49%	40.63%
N = 10	32.95%	34.38%	35.80%	37.50%	39.39%	40.81%
N = 15	32.01%	**33.90%**	**35.32%**	**37.50%**	**39.49%**	**41.29%**
N = 20	31.91%	**34.19%**	**35.14%**	**37.23%**	**39.58%**	**42.14%**

5.9 Final prototype

To develop the final prototype all parameters have been tuned so that the system did not return any incorrect result, as it was intended. Each time a new query image is introduced into the system, a ranking is generated taking into account the number of visual words in common between the query image and all images in the database. If the ratio of the highest ranked page and the second best is not less than 1.5 the highest ranked page is selected, otherwise it is held the brute force approach in the top 10 results, ie are calculated the number of best matches between the query image and the 10 pages that had better scores on the results returned by the inverted index. Then a re-ranking step based on the results of the brute force approach in the top 10 results is conducted. In this case, if the ratio between the best rated page is double or more compared to the second best rated, the best rated page is chosen, otherwise the system does not return any page and it is assumed that the page requested by the user does not belong to the database.

The results obtained by entering all the images from the Fashion Magazine dataset are shown in Table 5.8.

Table 5.8: Percentage of matches between the query image and the corresponding magazine page in the database.

Query type	**Correct Match (%)**
1	69.70%
2	59.85%
3	67.42%
4	38.64%
5	17.42%

From the results it is concluded that the system is very robust when introduced query images of the first three types. However, when the difficulty increases (query types 4 and 5) the number of correct matches is reduced. Despite the reduction in the percentage of correct matches, the prototype is still able to recognize some fashion magazine pages from challenging query images as shown in Figure 5.12.

In order to easily demonstrate the prototype built, a proof of concept consisting in an interface was developed to make possible to test the introduction of a picture taken by an user in the system. Figure 5.13 shows screenshots of the developed interface.

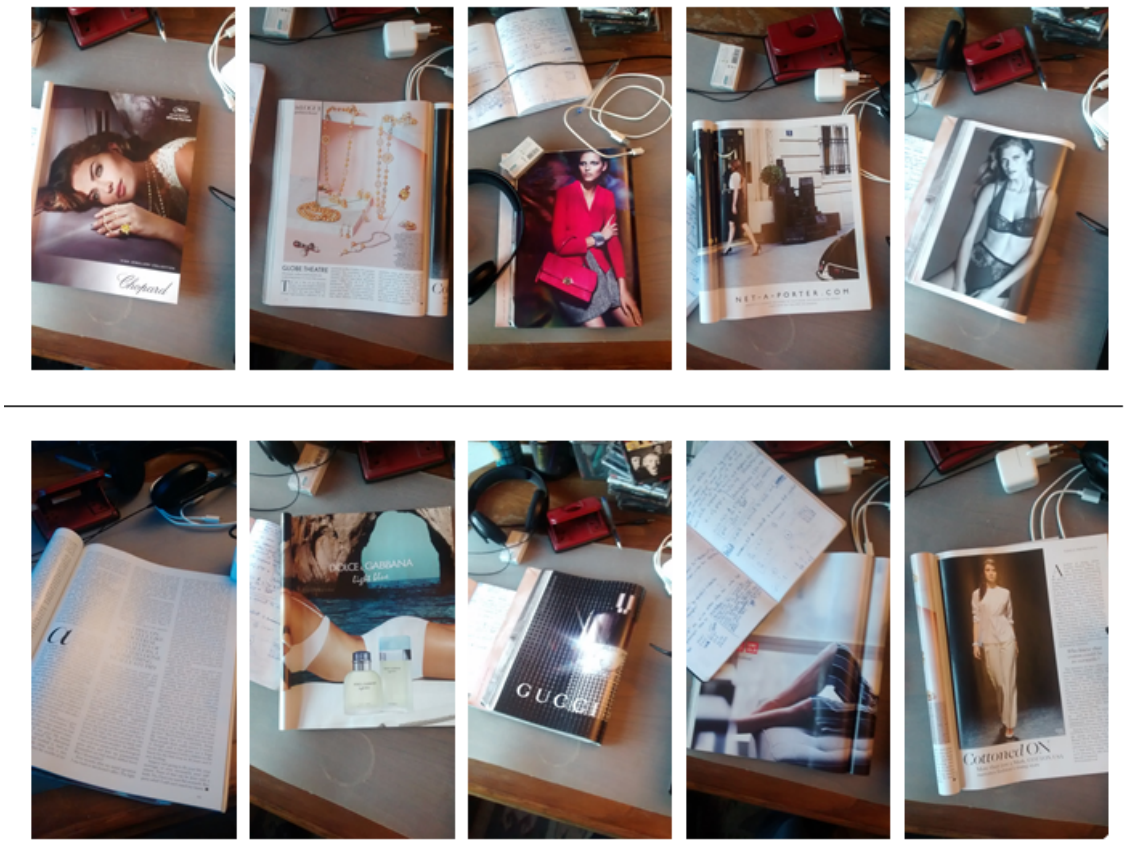
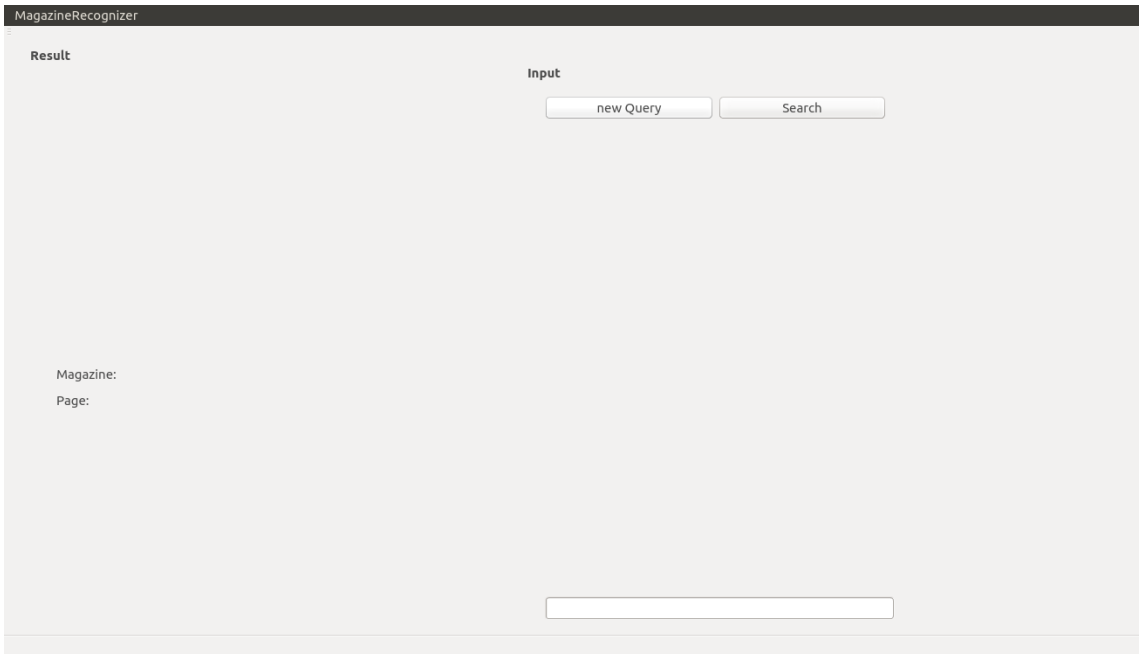
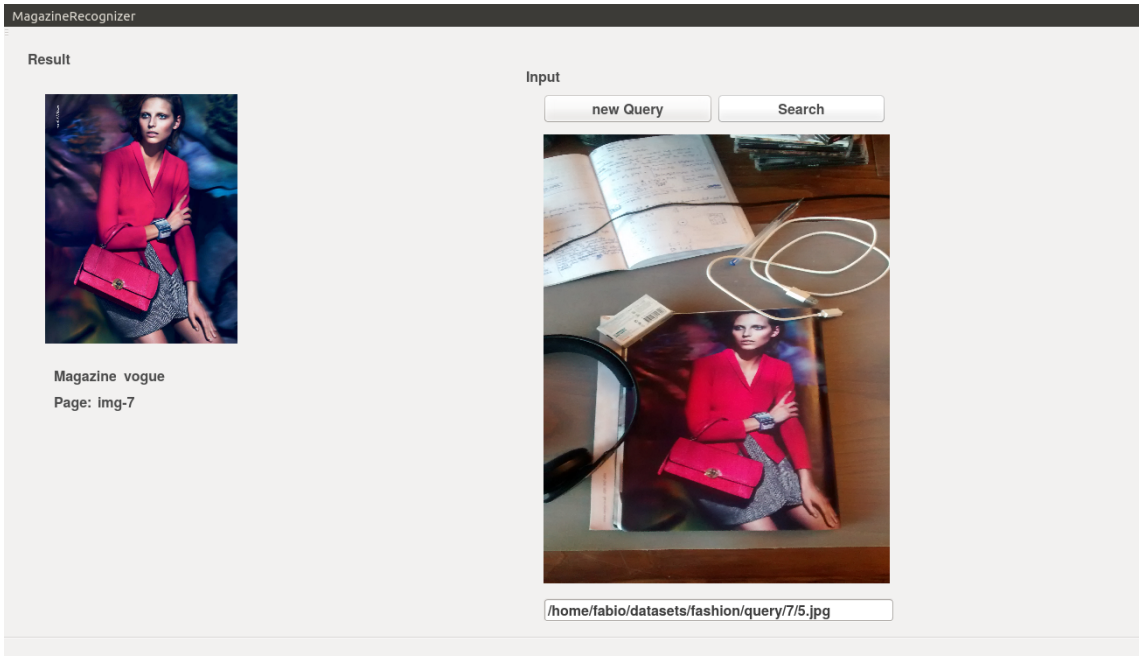


Figure 5.12: Some examples of query images of the type 5. The top line shows images that the system recognizes successfully, returning the correct page despite the difficulty associated with such images. Below examples of query images for which the system can not return a reliable result are presented.



(a) Interface developed for the system prototype.



(b) Sample result returned by the prototype. On the right is showed the photograph entered by the user and on the left the search result returned by the system.

Figure 5.13: Interface developed to be used as system's proof of concept.

Chapter 6

Conclusions and Future Work

During this dissertation a content-based image retrieval application was developed. Several works have focused on this specific matter. For the comparison of images in terms of similarity, features had to be extracted from both the query image and the database image. Various feature detectors and descriptors performed this task successfully. As demonstrated in the literature, early approaches, such as SIFT and SURF remain as standards in terms of system accuracy, however, newer approaches based on binary descriptors, such as ORB, BRISK and FREAK have shown to achieve similar results of accuracy while outperforming the standard ones in terms of efficiency. This aspect is critical when using lower powered devices, such as smartphones. To improve performance even further, keypoint descriptors extracted from the images can be clustered into bags of keypoints, and these image signatures obtained by quantizing the descriptors into visual words can be matched directly and efficiently instead of matching the keypoint descriptors.

The binary descriptors were used to solve the problem of image retrieval. The experiments on the fashion magazine dataset showed that, while simple and compact, the binary descriptors are very discriminative and close in performance to much higher dimensional descriptors such as SIFT. They especially provide a boost in the system's computational efficiency. The results showed that binary descriptors have their role to play in image retrieval systems using the bag of keypoints approach since they achieve similar performance than traditional descriptors.

In order to achieve the main goal of this thesis which consisted in the development of a content-based image retrieval application for fashion magazine recognition and retrieval, a study to determine how representative could be vocabularies built from binary keypoint descriptors were made in the fashion magazine retrieval task. Also the factors that influence system performance using this type of vocabularies were studied. Grounded on this ideas it was implemented a proof-of-concept system to validate the use of binary keypoint descriptors in the bag of keypoints approach. The results are encouraging but need continuous research in order to be refined.

6.1 Future Work

To further improve results, different image matching techniques can be applied to the most similar images (*top N*) to remove false positive results more effectively (images that appear as similar but actually depict different magazine pages). A homography calculation between the query image and the *top N* images in the database, using for example the *Random Sample Consensus* (RANSAC) or other similar techniques, can refine the results. System scalability should be studied more thoroughly in order to understand if it is possible to index several magazines without losing much accuracy. Besides this the motivation to find strategies to generalize the vocabulary is large and must then be improved Also, as it was part of the objectives of this dissertation, the end-to-end web or mobile based application can be finalized.

References

[1] Andrew Ng. The k-means clustering algorithm, 2014.

[2] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: a survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008.

[3] Edward Rosten and Tom Drummond. Machine learning for high–speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, pages 404–417. Springer, 2006.

[5] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[6] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010*, pages 778–792. 2010.

[7] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[8] Stefan Leutenegger, Margarita Chli, and Roland Yves Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[9] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.

[10] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, 2005.

[11] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003.

[12] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, 2004.

[13] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[14] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3304–3311. IEEE, 2010.

[15] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[16] Josef Sivic, Bryan C Russell, Alexei A Efros, Andrew Zisserman, and William T Freeman. Discovering objects and their location in images. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 370–377. IEEE, 2005.

[17] Frederic Jurie and Bill Triggs. Creating efficient codebooks for visual recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 604–610. IEEE, 2005.

[18] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Manchester, UK, 1988.

[19] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *Computer Vision—ECCV 2002*, pages 128–142. Springer, 2002.

[20] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International journal of computer vision*, 60(1):63–86, 2004.

[21] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Computer Vision–ECCV 2010*, pages 183–196. Springer, 2010.

[22] Stuart Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.

[23] Edward W Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[24] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[25] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 525–531. IEEE, 2001.

[26] Paul L Rosin. Measuring corner properties. *Computer Vision and Image Understanding*, 73(2):291–307, 1999.

[27] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[28] Tony Lindeberg. Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116, 1998.

[29] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[30] Mustafa Ozuysal, Michael Calonder, Vincent Lepetit, and Pascal Fua. Fast keypoint recognition using random ferns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):448–461, 2010.

[31] Vincent Lepetit and Pascal Fua. Keypoint recognition using randomized trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(9):1465–1479, 2006.

[32] Jared Heinly, Enrique Dunn, and Jan-Michael Frahm. Comparative evaluation of binary features. In *Computer Vision–ECCV 2012*, pages 759–773. 2012.

[33] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. Evaluation of interest point detectors. *International Journal of computer vision*, 37(2):151–172, 2000.

[34] Dagmawi Bekele, Michael Teutsch, and Tobias Schuchert. Evaluation of binary keypoint descriptors. In *ICIP*, pages 3652–3656, 2013.

[35] Costantino Grana, Daniele Borghesani, Marco Manfredi, and Rita Cucchiara. A fast approach for integrating orb descriptors in the bag of words model. In *IS&T/SPIE Electronic Imaging*, pages 866709–866709. International Society for Optics and Photonics, 2013.