

THE UNIVERSITY OF ADELAIDE

School of Computer Science

Efficient and Robust Image Ranking for Object Retrieval

Yanzhi Chen

December, 2013

SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN THE
FACULTY OF ENGINEERING, COMPUTER & MATHEMATICAL SCIENCES

INTRODUCTION

CHAPTER I

1.1 Problem statement

This thesis focuses on efficient and effective object retrieval from a large scale image dataset. The goal of object retrieval is to, given a query image depicting an object, return the dataset images containing that same object quickly and accurately. For example, to find the “Eiffel Tower” from large number of unordered images, as illustrated in Figure 1.1. More specifically, we aim to search for a query object based on the *visual content* of query and dataset images, instead of text words or tags used by most search engines, *e.g.* Google.

Throughout this thesis, we are looking for a specified object within an image, instead of an entire image. Therefore, the query is usually given as a ROI (region of interest) within the query image. Moreover, we search for a particular object, *e.g.* an individual car, instead of trying to match a general car. Conversely, this thesis is not concerned with classifying images into categories with some pre-defined or online learnt labels.

An object retrieval system is required to rank images containing the target object higher than those do not. Thus, by truncating the ranked result, images containing the target object are returned. The results should also be returned in real time, *i.e.* in a couple of CPU seconds. Therefore, this thesis aims at accurate, scalable and efficient object retrieval system.

1.2 Why is it difficult ?

Accurate, scalable and efficient visual search has challenged computer scientists for many decades, although in the early years researchers thought it was an “undergraduate student’s summer work” [15]. The difficulty arises from the fact

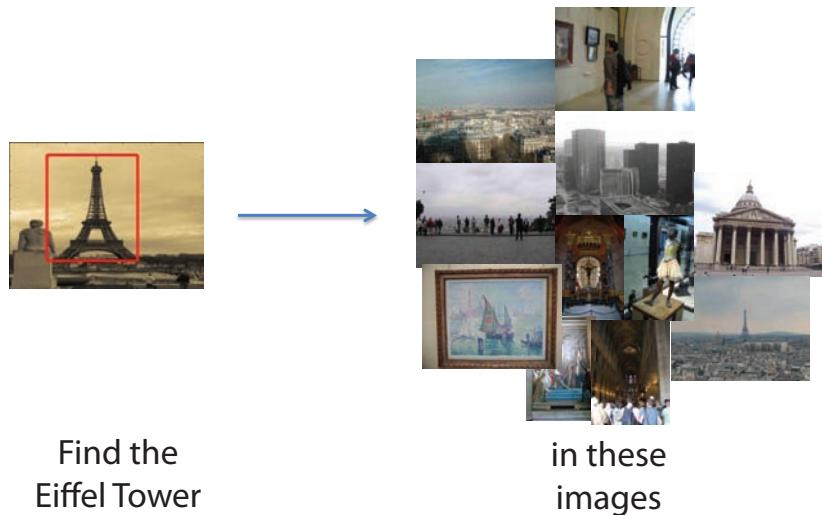


Figure 1.1: Example of object retrieval: to find the “Eiffel Tower” from large number of unordered images.

that the appearance of an object significantly changes in various image conditions: scale, viewpoint, lighting and partial occlusion of objects, as shown in Figure 1.2. The retrieval results, as shown in Figure 1.3, should return images containing the query objects at the top ranks. Making accurate retrieval results requires a model that captures the relevant image content. Equally important is the image similarity measurement, which determines the orders of the results returned by a search engine. These involve a number of techniques in computer vision, *e.g.* feature detection and description and nearest neighbor search, as well as techniques borrowed from related research area, such as information retrieval.

The proposed techniques, however, also need to scale to large numbers of images. The problem of object retrieval has become more relevant in recent years, because of the availability of image collections from the Internet. FLICKR, Facebook and other image sharing websites provide high volumes of visual data available online. A particular object, *e.g.* the “Eiffel Tower”, is associated to large amount of images in these websites. Therefore, an object retrieval system needs to organize over millions of images and return the retrieval results in a few CPU seconds.

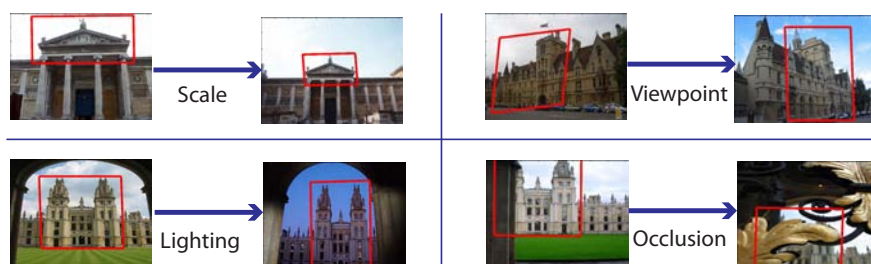


Figure 1.2: Examples of image condition changes in scale, viewpoint, lighting and partial occlusion of objects.

1.3 Contributions

Most successful object retrieval systems are built on a Bag-of-Words model, as proposed in [130]. Typically, each image is represented as a weighted vector of quantised features, known as visual words; dataset images are sorted according to their dot product similarity to the given query. Although the BoW retrieval system is efficient, the extraction and quantisation of local image features introduces errors into the retrieval results. This thesis aims to improve the quality of retrieval results. The contributions of this thesis are divided into three aspects: refinement of image representation, refinement of similarity measure and result re-ranking.

- Visual thesaurus structure and its applications. A visual thesaurus is able to store spatial relatedness of visual words. Based on the visual thesaurus, the image can be represented in a way that is more robust to appearance changes and the image similarity measure can be refined.
- Cross-word image similarity measure. A cross-word matching scheme is proposed to modify the original word-to-word image similarity such that features corresponding to the same world location, but assigned to different words, can be matched.
- Context based re-ranking. The initial retrieval results are further examined by contextual information embedded in their ranks. Images belonging to a coherent context will be promoted in the ranks, otherwise they will be demoted.
- Ranking verification. A result verification method is helpful to remove false positives from the initial retrieval results. We examine the retrieval results

only by their ranking information, and with no need to recover low level feature information.

- Group-query with a collection of images. In group-query, the query object is described by a small collection of related images. This overcomes the sensitiveness of single query instance.

1.4 Thesis outline

In Chapter 2 we review recent work on image retrieval with visual content, which is known as content-based image retrieval (CBIR) in literature. Section 2.1 describes a general content-based image retrieval system, while Section 2.2 focuses on object retrieval system built on a “Bag-of-Words” (BoW) model. We describe the detailed architecture of our system implementation, dataset organization, and evaluation in Section 2.3. The review of recent improvement and applications of the standard BoW retrieval system is discussed in Section 2.4 and Section 2.5, respectively.

In Chapters 3 and 4, we discuss several changes to the standard pipeline to improve the retrieval performance. The focus in these two chapters is the quantisation error that occurs in the BoW model, in which low level feature information is ignored. Chapter 3 presents a *visual thesaurus* structure to capture the latent spatial information of visual words. A spatial expansion method is proposed to refine the original query by including latent spatially related words. In this manner, the image representation is refined. Chapter 4 describes three methods to improve the image similarity measurement. Firstly, we present a visual word re-weighting scheme such that weights of reliable visual words are increased to improve the precision. Secondly, we associate spatial expansion and visual word re-weighting to further improve the BoW representation, which boosts both recall and precision. Thirdly, we derive a cross-word matching method to refine the dot product similarity, such that matched features are considered in the similarity measure even if they are mapped to different visual words.

The methods presented in Chapter 3 and Chapter 4 work well when foreground information can be well captured. However, the dataset might lack foreground information or it may be too expensive to explore these information. Retrieval techniques then pay attention to re-ranking the initial retrieval results with low level feature information or high level semantic information. In this

thesis, we use ranking information collected online, without knowledge learnt offline. Chapter 5 presents a novel re-ranking method based on contextual information obtained from the initial retrieval results. In contrast to previous methods, the re-ranking process only needs information about rank purity of initial retrieval results. It takes advantages in two aspects *i)* the re-ranking method does not need to change the offline module of the standard retrieval system. *ii)* the re-ranking method does not rely on estimating a semantic relationship between dataset images.

Chapter 6 also uses ranking information in the retrieval system. In contrast to Chapter 5, we propose a re-ranking method based on examination of ranking consistency between a pair of query/dataset images. The examination of ranking consistency leads to an improved similarity measure, which only considers the ranking information when each image is used as a query. Therefore, our ranking verification method is not restricted to rigid object retrieval, instead it can be applied to various kinds of image datasets. Moreover, the ranking verification results can be applied to a number of related problems: query expansion and dataset image mining.

In Chapter 7, we revisit the problem of query types. In previous chapters, the query is defined as “query-by-example”, in which the input query to the retrieval system is a single query instance. In this chapter, we define the query as group-query, a collection of images depicting the same object, and the retrieval system aims to find all images having the given object. Two types of ranking functions are proposed in Chapter 7: the averaging ranking function and the discriminative ranking function. Each of them plays a different role in ranking dataset image according to the group-query.

Chapter 8 summarizes the contributions of our work in object retrieval. We also discuss some future directions of research in object retrieval.

1.5 Publications

The visual thesaurus structure in Chapter 3 is proposed in [28]. The works in Chapter 3 and Chapter 4 have already been presented in [28, 26], as well as is under journal version revision [27]. The ranking consistency presented in Chapter 6 is also under journal revision [29]. The group-query retrieval in

Chapter 7 was published at [30]. The work of context based re-ranking in Chapter 5 is about to be submitted to publication.


























































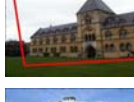








	Query images	Top ranked results					
All souls							
Ashmolean							
Balliol							
Bodleian							
Christ church							
Cornmarket							
Hertford							
Keble							
Magdale							
Pitt river							
Rac. camera							

Figure 1.3: Examples of query images of Oxford buildings. The query objects are indicated in a bounding box in the left column, and the corresponding retrieved results are listed in the right column.

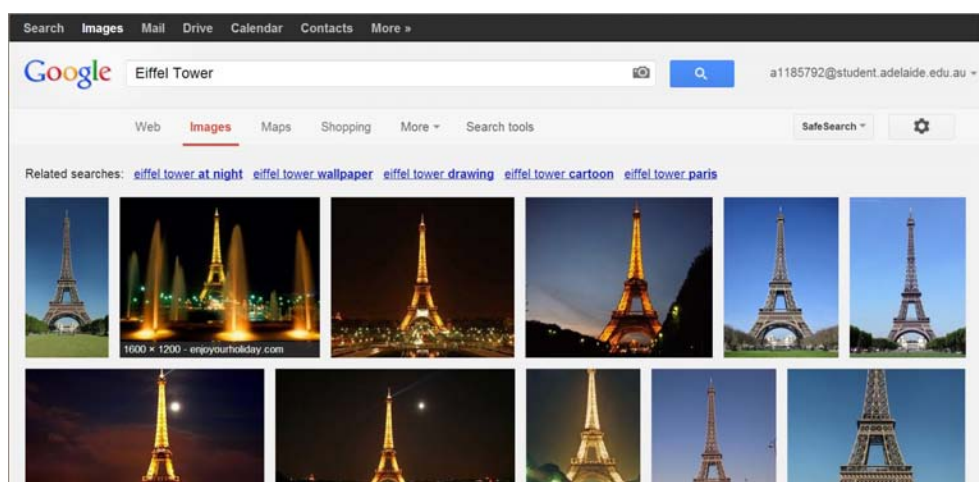
LITERATURE REVIEW

CHAPTER II

This chapter reviews recent research in retrieval methods based on visual content. This is known as content-based image retrieval (CBIR), which is a vast field of research. Typically, it is a query-by-example problem: the user inputs an image as a query and the CBIR system returns a set of images similar to the query. The retrieval results are ordered, *i.e.* similar images should be returned earlier than those are dissimilar to the query. Moreover, if an object is given in the query (user-specified ROI), the retrieval system needs to find all dataset images containing the query object and locate position of the object in images (if possible). Query with visual content is challenging, because images alter in viewpoints, scale, lighting, and the target object might be occluded, as shown in Figure 1.2.

Most commercial image search engines, *e.g.* Google, primarily use text tags as query: the user inputs some text words to describe the object for retrieval and the search engine returns results that match annotation labels given in the query. However, query with text tags does not return accurate retrieval results when there is inconsistency between human perception and annotation accuracy, as shown in Figure 2.1. For example, the retrieval results of “Eiffel” is mostly correct, while the retrieval results of “Napier” include many images that do not contain the target object (the “Napier” building). As a result, the retrieval performance is affected when the text words are ambiguous to describe the query object.

To overcome the gap between text tag description and human perception, content-based image retrieval (CBIR) system was introduced in the early 1980s and since then has attracted increasing interest from both academia and industry. In a CBIR system, images are compared and searched by their visual content. The input query is usually an image instead of text tags. Some early CBIR systems, *e.g.* IBM’s QBC [42] and Photobook [104], are developed on relatively small datasets. With the increasing availability of images shared on the Internet, it is possible to collect millions of images uploaded by users worldwide. Thus, the



NOTE:
This figure/table/image has been removed
to comply with copyright regulations.
It is included in the print copy of the thesis
held by the University of Adelaide Library.

Figure 2.1: Image retrieval results by text-based query. The input query is text tag: “Eiffel Tower” and “Napier Adelaide University”. The retrieval results of “Eiffel Tower” is accurate, while the retrieval results of “napier” building in Adelaide University is random, which includes many images that do not contain the target object.

challenges facing a CBIR system are that: *i*) the dataset images vary in a number of challenging image conditions, as illustrated in Figure 1.2. Therefore, the CBIR system needs to be robust to the image condition changes. *ii*) image dataset size is enlarging. The CBIR system faces the problem of balancing efficiency and effectiveness.

Although many methods have been proposed in building a CBIR system, searching with visual content is still a difficult problem because of the above two challenges. Moreover, most CBIR systems are based on low level features such as corners and gradients, which leads to the gap between human and feature described image content. The CBIR system needs to narrow the “semantic

gap” [131].

Comprehensive surveys on CBIR systems can be found in literatures [118, 131, 82]. In this chapter, we recap the most relevant work in image retrieval in the past decade, but pay special attention to methods aiming for object retrieval, where an object appears as foreground in an image, and the rest of the image is to be ignored (background). We begin this chapter by introducing a general CBIR system in Section 2.1 and then focus on the BoW based retrieval system (Section 2.2). Datasets and evaluation criterion are introduced in Section 2.3. We also review some widely used improvements on the standard retrieval system in Section 2.4. Finally, we discuss the applications of retrieval results to a couple of related problems (Section 2.5).

2.1 Content-based image retrieval

A CBIR system is built on visual content. A fundamental issue for a CBIR system is to compare the visual content between pairwise images. This is usually achieved by two components: the description of visual content and analysis of visual content. Both of them need to consider the scalability to a large scale dataset. Thus, the retrieval techniques used in a CBIR system must address the following issues:

1. **Description of visual content:** the mathematical representation of an image. It aims to convert an image into a mathematical representation. An image is composed of raw pixels. Thus it is hard to directly compute the matches between pairwise images. To address this, images are represented by various features, as discussed in Section 2.1.1.
2. **Analysis of visual content:** the ranking function to produce result lists. It compares the similarity between a pair of query/dataset images. The search engine then ranks the dataset images by these similarity scores, and usually returns a shortlist of ranked results, as discussed in Section 2.1.2.
3. **Scalability of a retrieval system:** a CBIR system needs to make a balance between the efficiency and effectiveness when choosing the techniques of description and analysis of visual content. For example, a system needs to return the retrieval results from millions of image in a few seconds.

Below we review some commonly used techniques in a CBIR system.

2.1.1 *Description of visual content*

The description of visual content involves a variety of feature detectors or/and descriptors. With these steps, images are represented by some feature-specific structures in the image for computational task. Typically, the image features can be obtained globally or locally.

Global features describe the image content as a whole, *i.e.* each image is assigned a single feature vector. Visual representation by global features reflects the global statistical characteristics of an image [79]. However, it can not distinguish foreground from images [143]. In contrast, local features describe the image content as a collection of image patterns which differ from their immediate neighborhood [143]. Local feature is usually taken from a local invariant region and then converted into a feature descriptor. Each image is represented by a collection of local feature descriptors. This involves detection of local invariant region, whose location should be robust to image condition changes. Therefore, local feature is suitable for describing foreground. Details of the evaluation of local region detectors can be found in the survey [143]. In this section, we discuss the usage of various features in content-based image retrieval. Note that the choice of feature type is determined by both the memory cost and the variety of image dataset, and vice versa.

Memory cost of image dataset The size of image datasets has increased dramatically in the past decade. For example, QBIC dataset [42], which is built in 1995, contains 7.5K images; “INRIA Holiday” dataset [63] contains 10M (and then 100M) images in 2010. Typically, dataset built on global features requires a single image feature of each dataset image, while dataset built on local features needs hundreds (or thousands) of features per image. For example, the Oxford 5K building dataset [106] needs 2500 local feature per image on average. Therefore, a CBIR system needs to deal with the increasing availability of images when choosing to use global or local features.

Image dataset A CBIR system also needs to take account of image dataset. The system built on global feature aims at image-level search, *i.e.* to search an entire

image as a whole. In contrast, the system built on local feature pays attention to object instance. Therefore, local features are more suitable for a retrieval system when the object of interest does not dominate the image.

In summary, global features are widely used to retrieve an entire image and are economical in memory cost, while local features are used to retrieve a specified object within an image and usually are more expensive in memory cost than global features. Below we review some commonly used image detectors and descriptions in a CBIR system.

Survey of global features

Early image retrieval systems represent images with some global information, for example, color, texture or shape [131]. Some commonly used global features in CBIR system are discussed below.

Color feature is one of the most widely used global features in CBIR systems. Color reflects the raw pixel information of an image. The color content of an image can be defined by the distribution of pixels in a selected color space [136]. The detail description of different color space can be found in [112]. Commonly used color features in CBIR system includes color histogram, color moments, and color sets. Among them, color histogram is the widely used color feature representation. For example, Bradski [18] uses a histogram in HSV space for object representation. Swain and Ballard [136] indexes images by color histograms, which count the number of discrete color occurring in the image. They use histogram intersection as the similarity measure for the color histogram. However, these color features are not directly related to high-level semantic information.

Texture is another global information used in image retrieval systems. Texture describes the content of many real world images, *e.g.* the material of imaged appearance. Textures can be captured under different lighting and viewing conditions. In order to obtain discriminative patterns, there have been a number of feature extraction methods. The earliest method is statistical features of gray levels, for example, gray level co-occurrence matrices (GLCM) [53] are used in texture classification. Tamura *et al.* [137] defined six textural features: coarseness, contrast, directionality, line-likeness, regularity and roughness. The Tamura features characterize textures in terms of local statistical measure. Other popular methods include Gabor filters [87] and wavelet transform [148]. These

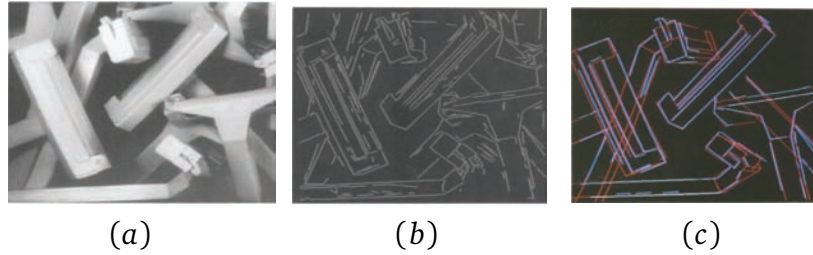


Figure 2.2: Example of a successful recognition of object boundaries. Taken from [85]. (a) Original image. (b) Result of line segmentation. (c) Recognized model of image (a).

enable filtering in the frequency and spatial domain. A full comparison of them used in image retrieval can be found in the survey [55].

Shape features are important image features although they are not as widely used as color and texture features in CBIR system. These include ratio, circularity, Fourier descriptors, moment invariants, and boundary segments [90], etc. Among them, the commonly used shape feature is the object boundaries, e.g. literatures [85, 58] search objects by correspondence between an object model (e.g. 3D wire-frame) and detected image primitives (e.g. points and line segmentations). Example of a successful recognition of object boundaries is shown in Figure 2.2. The correspondence search is usually addressed by geometric methods: interpretation trees [49, 51], pose clustering [135] and RANSAC [43]. However, these methods are unreliable in detecting of object contours in real world images.

Super pixel is a segmentation-based visual representation, which aims to group pixels into perceptually meaningful regions. “Blobworld” [23] performs object retrieval with a small set of image regions (super pixels) as query, instead of with an entire image in the previous work. The “Blobworld” representation groups pixels into regions, thus “objects” were extracted from images in a bottom-up manner. However, the object obtained in [23] highly depends on the results of segmentation with super pixels.

Survey of local features

In recent years, detecting interest points invariant to scale and affine transformation has become more popular. The key idea is to detect regions invariant to a class of transformations, such that they are well suited to matching wide-

baseline images [13, 89, 113, 145], reconstruction of camera sets [122], image stitching and building panoramas [20]. The difficulty is that the detected regions, extracted from various image conditions, should correspond to the same pre-image for different viewpoints.

Typically, the extraction of local feature proceeds in two steps. Firstly, local invariant regions are detected from each image. The term *region* refers to a neighborhood of pixels to be analyzed. There are several popular affine invariant region detectors in literature, for example: *Harris-affine* [93], *Hessian-affine* [93], *Maximally stable extremal regions* (MSER) [89] and Salient region [69]. Figure 2.3 shows examples of detected regions on one pair of images captured from slightly different viewpoints. Each detected region is represented by an elliptical region, which shows the original detection size. As seen in these results, affine invariant region detectors are able to correspond to the changes of regions which are locally affine between two views. Secondly, a vector descriptor is associated with each region, computed from the intensity pattern within the region. *e.g.* SIFT [83], or more efficient SURF [14]. Besides descriptors extracted from local regions, dense descriptors are also revisited recently, *e.g.* DAISY [140]. It retains the robustness of SIFT, but can be computed quickly at each image pixel.

Based on these methods, an image can be represented by a collection of local feature descriptors. Images are matched by the correspondences between feature descriptors. This is a fundamental component of a CBIR retrieval system. Early attempts of local appearance models [123, 124, 144, 40, 83] directly base their retrieval techniques on the local features. Compared to the retrieval methods built on global feature, these methods are better able to find reliable matches among a variety of real world objects because they do not require whole image to be similar. The increase of image dataset size requires more efficient matching between images, therefore, local feature is quantised into some visual word IDs, and the images can be compared according to the appearance of these reduced information. We discuss this in Section 2.2.

2.1.2 Visual content analysis

The analysis of visual content is complicated because of the gap between low level visual content and high level user concept. Human interpret images with high-level features, such as identity of an object. However, the features extracted

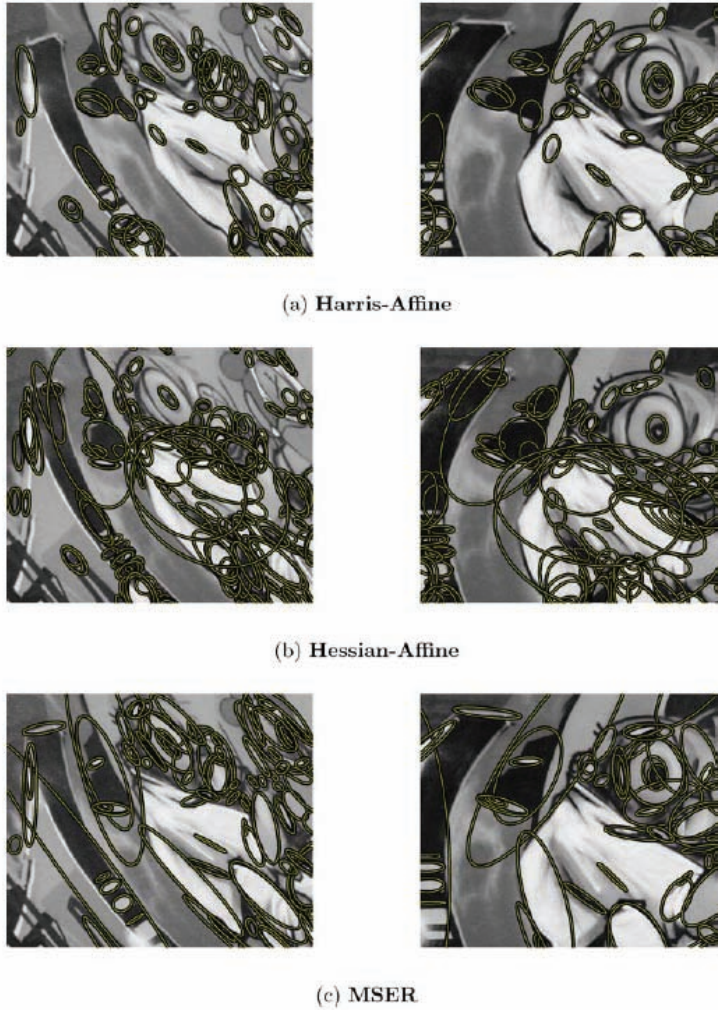


Figure 2.3: Example of affine invariant regions detected by various detectors. Taken from Mikolajczyk *et al.* [95]. The ellipses show the original detection size. (a) Harris-affine. (b) Hessian-affine. (c) MSER.

by computers are low-level features, *e.g.* color, texture, shape or super pixels. Therefore, it is hard to link the high-level user concepts and low-level visual content [82].

In order to meet the requirements of user concept, the query in CBIR is usually categorised into different types. In this thesis, we aim to search with specific objects. This is a *query-by-example* problem, where visual content is measured by pairwise image similarity measurements. As a result, a retrieval system needs to rank each dataset image according to its visual distance to the given query. Other types of query includes query by *object-class* or query by *visual*

attributes. The query by object-class requires the retrieval system to classify dataset images into different categorisations; while the query by visual attributes describes a query object by some physical traits, *e.g.* the facial attributes of a person (hair color, presence of sunglasses, etc). In this section, we review methods used for query-by-example methods, while other types of query is not the focus of this thesis.

Query-by-example retrieval

In the “query-by-example” retrieval, the analysis of visual content directly compares the similarity between query and dataset images. The retrieval system then uses a ranking function to order dataset images, according to their similarity to the query. Typically, the ranking function adopts an image similarity measure to assign a score to each dataset image d , while the image similarity measure depends on the type of image description, as discussed in Section 2.1.1.

The search of images built on global feature is based on the single feature vector of an image. Therefore, it can be converted into a nearest neighbor (NN) search problem: to find the element $NN(q, d)$ in a finite dataset \mathbf{V} by minimizing the distance to a given query q :

$$NN(q, d) = \arg \min_{d \in \mathbf{V}} \text{dist}(q, d) \quad (2.1)$$

The widely used distance between images q and d is Euclidean (L2) distance [130]. However, such exact NN search becomes expensive when in large scale high dimension space, *i.e.* the image dataset becomes large and image vectors are high dimensional data.

Alternatively, a great deal of research focuses on approximate nearest neighbor (ANN) search. In contrast to exact NN search, ANN search aims to return the nearest neighbors with a high probability while with improved speed and memory savings. ANN search finds ε -error neighbors for a given query q from dataset \mathbf{V} :

$$ANN(q, d) = (1 + \varepsilon)R \quad (2.2)$$

where R is the distance between query q and its true nearest neighbor, $\varepsilon > 0$ is the approximate factor and $d \in \mathbf{V}$ is the approximate nearest neighbor. Note that Eq. (2.1) is a special case of Eq. (2.2). The goal of ANN search is to reduce the

query time to a sub-linear function of the size of the dataset, while the memory usage and preprocessing time grows reasonably (*i.e.* exponential not acceptable).

For search of images built on local feature, the similarity measure is based on matching feature descriptors. It mostly uses Minkowski metric to define the distance between descriptors. Let $X = \{x_1, x_2, \dots, x_n\}$ and $Y = \{y_1, y_2, \dots, y_n\}$ be two descriptors having n dimensions, the Minkowski distance of order p is defined as:

$$\text{dist}(X, Y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}} \quad (2.3)$$

Usually, the order is $p = 1$ (Manhattan metric), $p = 2$ (Euclidean metric) or $p \rightarrow \infty$ (Chebyshev metric). Weighted Minkowski distance is a variant of Eq. (2.3), in which important features are weighted. By using the Minkowski distance, the similarity between two images can be *one-to-one* match or *many-to-many* match. The one-to-one match is to find the nearest (best) descriptor in the response image of each query descriptor [83]. The many-to-many match allows cross-descriptor match between pairwise images. A widely used method for many-to-many match is the earth mover's distance [117](EMD). The EMD method is originally proposed for color and texture histograms. It is a cross-bin similarity measure, which computes the distance between two probability distributions in a region. The histograms are therefore measured as a *transportation problem*. The EMD similarity between two images is the least amount of work to transport the units in one image to another.

2.1.3 A basic CBIR system

In CBIR system, the retrieval results are typically presented as a (truncated) list of ordered images, according to their visual similarity to the given query. A CBIR system needs to consider the following issues when generating lists from a large scale dataset:

- Accuracy. A retrieval system should generate accurate retrieval result, *i.e.* query-relevant images should be ranked higher than irrelevant ones.
- Run time. It should return the retrieved results for a given query image in real time, *i.e.* less than 1 second.
- Memory cost. It should store small constant amount of data per image.

Algorithm 1 General pipeline of a CBIR system.

1. Offline process

- Image processing: to obtain the description of visual content, *e.g.* feature detectors and descriptors, of all dataset images.
- Index in storage: to store description in file and will be accessed if required.

2. Online process

- Image processing: to obtain the description of visual content of given query.
 - Similarity measure: to analyse the visual content between query and dataset images.
 - Ranking: to sort the dataset images to generate a shortlist of candidate images.
 - Result re-ranking: to optionally refine the initial shortlist results.
-

In order to achieve these, a retrieval system is usually separated into two processes: offline and online. The offline process is independent from the given query, therefore usually involves expensive computation to process the whole dataset. This process is normally slow, but only needs to compute once. The online process should quickly compute query results, which should compute the similarity of dataset images efficiently and rank them to generate a (truncated) list of results. Each process is composed of several components to address different tasks in the retrieval system, which are described in Algorithm 1.

2.2 Visual image retrieval built on “Bag-of-Words” model

This thesis implements visual image retrieval built on quantised local features, because we are interested in efficient retrieval of foreground objects. We build our retrieval system on a “Bag-of-Words” (BoW) model. The BoW is a global feature vector built from local features. Thus we get the advantages of local and global features.

The BoW retrieval system is based on successful retrieval techniques developed in the past decade. The early object retrieval systems consider the problem of object retrieval as task of instance matching. These works aim to find geometric invariants from boundaries of the objects [116, 84]. However, this kind of retrieval system lacks efficiency because it needs to match each image feature to every possible model. Another kind of methods use an appearance based model for object retrieval. Murase and Nayar [98] represents the space of images using a low-dimensional eigenvector representation, thus the search of a query is to find the nearest neighbor in the eigenspace. This kind of method is promising in robust

recognition of objects, and lead to increasing attention to local appearance model in [123, 124, 144, 40, 83].

Based on these previous work, text-retrieval based approaches are applied in image domain to achieve efficient recognition of instances [40, 130]. The main contribution of [40, 130] is to reduce local features to discrete IDs (visual words), similar to text words in a document. This is known as a “Bag-of-Words” (BoW) model originally from information retrieval. Following them, visual object retrieval borrows a number of techniques from text-retrieval.

2.2.1 Why quantisation of local features?

The BoW model aims to simplify the computation of image comparison. Direct matching of local features is not computationally tractable. As discussed in Section 2.1.1, there have been a number of affine invariant detectors, *e.g.* Hessian-affine, Harris-affine and MSER, as illustrated in Figure 2.3. Each of them produces hundreds to thousands of local interest regions, each of which is described by a feature descriptor. In this thesis, we use Hessian-affine detector and SIFT feature descriptor as suggested by [106]. This leads to, for example, around 12.7 million 128D SIFT features being used to represent the relatively small Oxford 5K dataset. For each feature in a query, it needs to compare to all other features in the space. Therefore, search with raw features is not reasonable with respect to both computation cost and memory storage.

The “Bag-of-Words” (BoW) model further reduces each local feature descriptor, *e.g.* 128D SIFT descriptor, to a single ID number. It aims that matching features should be assigned to the same ID. Conversely, non-matching features should be assigned to different IDs. As illustrated in Figure 2.4, these cluster IDs are obtained by quantising the descriptors into cluster[s]. In Sivic and Zisserman [130], these cluster IDs are named as “visual words” (corresponding to “codewords” in text-retrieval) and the collection of them are known as “visual vocabulary” (corresponding to “codebook”). The quantisation successfully reduces an image from a set of high dimension local features to a set of 1D indexes. This enables an image to be considered like a document and each indexing ID like a text word.

The visual vocabulary building requires a scalable quantisation, as discussed in Section 2.2.2. After quantisation, the match of local features is reduced to finding those sharing the same visual word ID, while an image can be represented

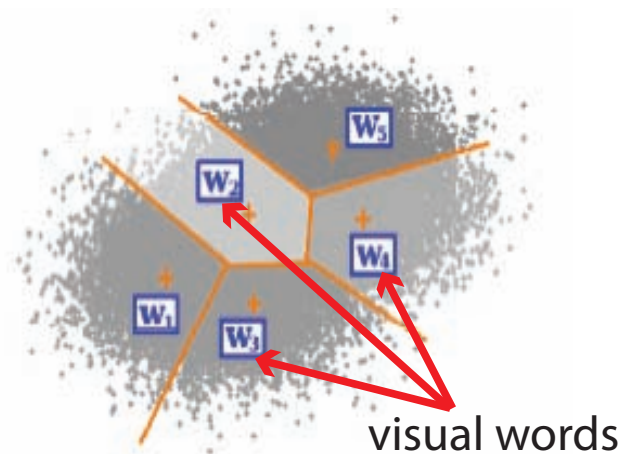


Figure 2.4: Illustration of quantisation in the BoW model. The visual words define a partition of the feature space, which is a Voronoi diagram.

by a frequency histogram of these visual words. This is analogous to word counting in text documents, *i.e.* given an image (document) we simply count the number of each visual words (codeword) occurs in the image (document). As a result, the visual words are orderless and the spatial information of raw features is ignored. Based on the BoW representation, text-retrieval methods can be directly applied in the image domain. The details will be discussed in Section 2.2.3.

2.2.2 Visual vocabulary building

The visual vocabulary partitions continuous descriptor space into discrete tokens. This fixes the search region of similar features into those sharing the same word IDs, instead of search in the whole descriptor space. However, the discretization of descriptor space is difficult when image dataset size becomes large. This section reviews the usage of visual vocabulary as a nearest neighbor search method, and the techniques to address scaling clustering problem.

Nearest neighbor search based on visual words

The visual vocabulary defines a partitioning of descriptor space, which is related to a Voronoi diagram in mathematics. The Voronoi diagram achieves the NN search by dividing continuous space into a number of discrete regions. The regions are called Voronoi cells, which are specified beforehand by a set of data points (seeds).

A corresponding region is assigned to each seed, where all points located in this region are closer to this seed than to any other seeds. Each feature descriptor is therefore assigned to a closest visual word. As a result, the nearest neighbor search of a local feature is reduced to a set of features indexed by the same visual word ID.

There are a great amount of methods to specify the Voronoi regions. Among them, the k-means clustering is the most commonly used method. It iteratively proceeds by two steps: *i*) assign each data point to the cluster whose mean is closest to it; *ii*) update the mean of clusters to be the centroids of the data in the new clusters. After k-means clustering, a high dimensional data can be represented as an ID in the diagram. This is the *quantisation* process used in many data mining problems: a D -dimensional feature \mathbf{x} is mapped to an integer index c_i by a quantizer q :

$$q(\mathbf{x}) = \arg \min_{i \in [1, N]} d(\mathbf{x}, \mathbf{c}_i) = c_i \quad (2.4)$$

where c_i is the index of its corresponding cluster centroid in $\mathcal{C} = \{\mathbf{c}_i\}_{i=1}^N$ resulted by clustering the descriptor space and N is the pre-defined number of cluster centroids. The resulting N cluster centres form a visual vocabulary $\mathbf{W} := \{w_1, w_2, \dots, w_N\}$.

The quantizer maps the high dimensional data to an integer, by finding the closest L2 distance of a feature to a cluster centroid. The BoW model thus acts as approximate nearest neighbor search by the assumption that two features \mathbf{x} and \mathbf{y} that are close in the feature space satisfy $q(\mathbf{x}) = q(\mathbf{y})$ with a high probability [63]. Therefore, a set of vectors \mathcal{V}_i mapped to the same index c_i is defined to be matched:

$$\mathcal{V}_i = \{\mathbf{x} : q(\mathbf{x}) = c_i\} \quad (2.5)$$

in which \mathcal{V}_i is referred as a Voronoi cell.

The effectiveness of quantisation depends on the vocabulary size. Early retrieval system uses flat k-means [130], in which the number of clusters is relatively small (10^4). The increase of dataset size also increases the probability to find similar but irrelevant image to a given query. Large vocabulary (10^6) is used to address this problem, so that irrelevant features are less likely to be assigned to the same word IDs [106]. The retrieval performance is significantly improved, however, flat k-means is too expensive to run on large scale dataset and vocabulary.

Note that flat k-means even is hard to scale Oxford 5K dataset, in which both number of features and cluster centres are required to be very large. Therefore, large scale visual vocabulary is built on an approximate manner, *e.g.* hierarchical k-means (HKM), approximate k-means (AKM) and Hamming embedding (HE) as discussed as follows.

Scaling clustering

Hierarchical k-means Nistér and Stewénus [100] proposes a tree structured scheme: Hierarchical k-means (HKM). The method organizes the large scale clustering as a tree of small clustering problems. Instead of clustering the high dimensional feature space into K final number of clusters (K is a very large number), hierarchical k-means defines the branch factor B (number of children of each cluster) of the tree and recursively partition each cluster as follows. The first level groups all the data points and runs standard k-means with a small number of cluster centres, *e.g.* $B = 10$. On the following levels, k-means is run within each partition and generate B cluster centres as well. The method stops when reaching the bottom of the tree and results in B^n clusters at n -th level. HKM is parameterized by tree height L and the branching factor B . The height of the tree is $L = \log K$, therefore the complexity of HKM is $O(N \log K)$ for N data points, when the base $B = 10$.

Approximate k-means Approximate k-means (AKM) [106, 97] solves the scaling problem in a different way. In standard k-means, the majority of computation is spent on NN search between the data points and clusters. Instead of exact NN search, AKM replaces this step with the approximate nearest search based on randomized kd-tree [97]. The key idea of AKM is to use randomized kd-trees (implemented in FLANN [97]) to build a forest of 8 randomised k-d trees over the cluster centres before iterations. During each iteration, a new data point is assigned to the approximate closest cluster centre. The randomised tree causes overlap of the feature space, and thus data points will be more likely to be located near the boundaries, as noticed in [107].

The complexity of AKM is the same as HKM. The classical kd-trees is a binary tree in which every node is a k -dimensional point. Each non-leaf node generates a splitting hyperplane such that the space is divided into two parts. The left

(right) subtree of that nodes represents the left (right) of this splitting hyperplane. Therefore, the kd-tree splits the data in half at each level of the tree. The NN search can be done efficiently by using the kd-tree structure in low dimensional space, however, its accuracy decreases in high dimensional space. The inefficiency in high dimensional space is due to the search needs very large number of nodes. Based on the classical kd-trees, a randomised kd-trees is built by randomly choosing the split dimension from among the set of dimensions with the highest variance [127]. The depth of each tree in the forest is $\log K$, therefore the complexity of AKM is $O(N \log K)$ for N points.

Hamming embedding In k-means clustering and its variations (HKM and AKM), the trade-off between accuracy and recall of nearest neighbors is managed by the number of clusters K . In order to achieve high accuracy, K is set as a very large number (e.g. 10^6) as in [100, 106]. However, the large vocabulary size also causes a lot of recall loss. Hervé *et al.* [63] extends the BoW model with a combination of coarse clustering methods (k-means) and a binary code that refine the descriptor. The combination takes the advantages from both sides: *i*) a rough partition with small visual vocabulary, which generates high recall rate. *ii*) binary encode of the local feature descriptors to the cluster centres, which leads to high accuracy. A relative small visual vocabulary used in the first stage, such that Hervé *et al.* [63] can use flat k-means to generate more balanced cluster than AKM. Based on these cluster results, Hamming Embedding (HE) maps the Euclidean distance between two descriptors \mathbf{x} and \mathbf{y} located in the same cluster centre into the Hamming distance:

$$h(b(\mathbf{x}), b(\mathbf{y})) = \sum_{i=1}^{d_b} \|b_i(\mathbf{x}) - b_i(\mathbf{y})\| \quad (2.6)$$

where $b(\cdot)$ is a d_b -dimensional binary signature. This mapping is from the Euclidean space into Hamming space. After that, a descriptor is represented by both $q(\mathbf{x})$ and $b(\mathbf{x})$. The HE matches two features into the same word ID if they are located in the rough cluster centre and their hamming distance is under a fixed threshold.

In this thesis, we use AKM for scaling clustering in the baseline system, as suggested by [106]. It achieves higher accuracy than HKM, as well as attains efficiency in offline clustering.

2.2.3 Efficient indexing by using text-retrieval methods

Under the BoW model, each image can be treated as a “document” that contains a collection of visual words. In text retrieval, each document is represented by a vector of weighted word frequencies [120]. Similar to that, a text-based image retrieval system can index each dataset image by a weighted image vector. Thus, image retrieval borrows a number of techniques from text retrieval.

Vector-space model Most modern text retrieval systems are built on the vector-space model [121]. The vector-space model [121] builds a text retrieval system in three stages: indexing, weighting and similarity measure. Firstly, document indexing represents each document d as a high dimensional vector $\mathbf{d} = \{\tau_i\}_{i=1}^N$, where each element τ_i is the weight of term¹ in document d , and N is total number of distinct words in the corpus. If a term i occurs in a document d , its value τ_i is nonzero. The frequency of term is varying in dataset images. Like text document, some words, *e.g.* “the”, “and”, “is”, have a high frequency in a document but do not describe the content. Therefore, the second stage is to weight the terms to discriminate one document from the other, and which usually considers both local and global information. The weight scheme is known as the term frequency inverse document frequency (tf-idf) model. For document d , its weighted vector is defined as:

$$\mathbf{d} = \{\tau_1, \tau_2, \dots, \tau_N\}^T \quad (2.7)$$

in which each element τ_i is weighted by two components: its frequency in both the individual document (tf) and the corpus (idf):

$$\tau_i = \underbrace{\frac{n_{id}}{n_d}}_{tf} \cdot \log \underbrace{\frac{V}{N_i}}_{idf} \quad (2.8)$$

where n_{id} is the number of occurrences of word i in document d , n_d is the number of visual words in image d , V is the dataset size and N_i is the number of documents containing word i . The term-frequency (tf) weights the word i occurring more often in an individual document higher (local parameter), while the inverse document frequency (idf) down weights the word i occurring more often in the corpus (global parameter). For the given query vector \mathbf{q} and any

¹Term can be a single word or longer phrase. In this thesis, we define term as a single word.

dataset image vector \mathbf{d} , the image similarity is defined as the cosine angle between these two vectors [130]:

$$\text{dist}(\mathbf{q}, \mathbf{d}) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\|_2 \|\mathbf{d}\|_2} \quad (2.9)$$

Eq. (2.9) is a dot product of the query vector \mathbf{q} and the dataset image vector \mathbf{d} divided by magnitudes of both vectors. The images are then ranked by descending scores of Eq. (2.9). Note that the original paper directly use L2 distance similarity between two BoW vectors [130, 106]: $\|\mathbf{q} - \mathbf{d}\|_2$. The images are then ranked by ascending L2 distances. However, the L2 distance and dot product distance is interchangeable (one is monotonic function of the other). When \mathbf{q} and \mathbf{d} are normalized (L2 norm), the L2 distance can be written as $(\mathbf{q} - \mathbf{d})^T(\mathbf{q} - \mathbf{d}) = \|\mathbf{q}\| + \|\mathbf{d}\| - 2\mathbf{q}^T\mathbf{d} = 2(1 - \mathbf{q}^T\mathbf{d})$, where $\mathbf{q}^T\mathbf{d}$ is the dot product between two BoW vectors. The BoW vectors are very sparse, thus the computation of dot product is efficient, as it only needs to compute the entries that are both non-zeros in a pair of image/dataset images.

Inverted file An inverted file data structure helps the retrieval system quickly to access the target documents [161]. An inverted file is composed of a collection of lists, one per term, recording the pointers that maps terms to the documents that contain them. An example is the Keeper dataset illustrated in [160], which contain 6 documents and 19 terms as shown in Table 2.1. The corresponding inverted file is illustrated in Table 2.1. When search for the terms “big”, “gown” and “house”, the inverted file would return the intersection of the lists: $\{2, 3\} \cap \{2\} \cap \{2, 3\} = \{2, 3\}$. As a result, the retrieval system only needs to consider the second and the third documents in the Keeper dataset, other documents are irrelevant. Similar to the inverted file built on text document, image dataset can also be indexed in the same structure. Each visual word is assigned with an image list, in which images containing the same visual word are mapped to the same ID.

2.2.4 The BoW retrieval system architecture

The BoW retrieval system follows the general outline of a CBIR system given in Algorithm 1. The BoW retrieval system is separated into two processes: offline and online. The offline process builds visual vocabulary and inverted file for dataset

- 1 The old night keeper keeps the keep in the town
- 2 In the big old house in the big old gown
- 3 The house in the town had the big old keep
- 4 Where the old night keeper never did sleep
- 5 The night keeper keeps the keep in the night
- 6 And keeps in the dark and sleeps in the light

Table 2.1: The Keeper dataset [160].

term t	frequency f_t	Inverted list for t
and	1	<6, 2>
big	2	<2, 2>, <3, 1>
dark	1	<6, 1>
did	1	<4, 1>
gown	1	<2, 1>
had	1	<3, 1>
house	2	<2, 1>, <3, 1>
in	5	<1, 1>, <2, 2>, <3, 1>, <5, 1> <6, 2>
keep	3	<1, 1>, <3, 1>, <5, 1>
keeper	3	<1, 1>, <4, 1>, <5, 1>
keeps	3	<1, 1>, <5, 1>, <6, 1>
light	1	<6, 1>
never	1	<4, 1>
old	4	<1, 1>, <2, 2>, <3, 1>, <4, 1>
sleep	1	<4, 1>
sleeps	1	<6, 1>
the	6	<1, 3>, <2, 2>, <3, 3>, <4, 1>, <5, 3>, <6, 2>
town	2	<1, 1>, <3, 1>
where	1	<4, 1>

Table 2.2: Inverted file for the Keeper dataset [160]. Each term t in the corpus as a list of points to map each term to the documents containing it.

images, while the online process searches the dataset based on quantised visual words and ignores the original local features. Each process is composed of several stages as illustrated in Figure 2.5.

Typically, the offline process proceeds as follows:

- Feature detector and descriptor: The first step of BoW retrieval system is to extract local interest points from each dataset image. Each interest region is represented by a feature descriptor (Section 2.1.1). This proceeds on all dataset images and image input as query.
- K-means clustering and quantisation. The second step is to generate a visual vocabulary. The feature descriptors are clustered by approximate k-means to

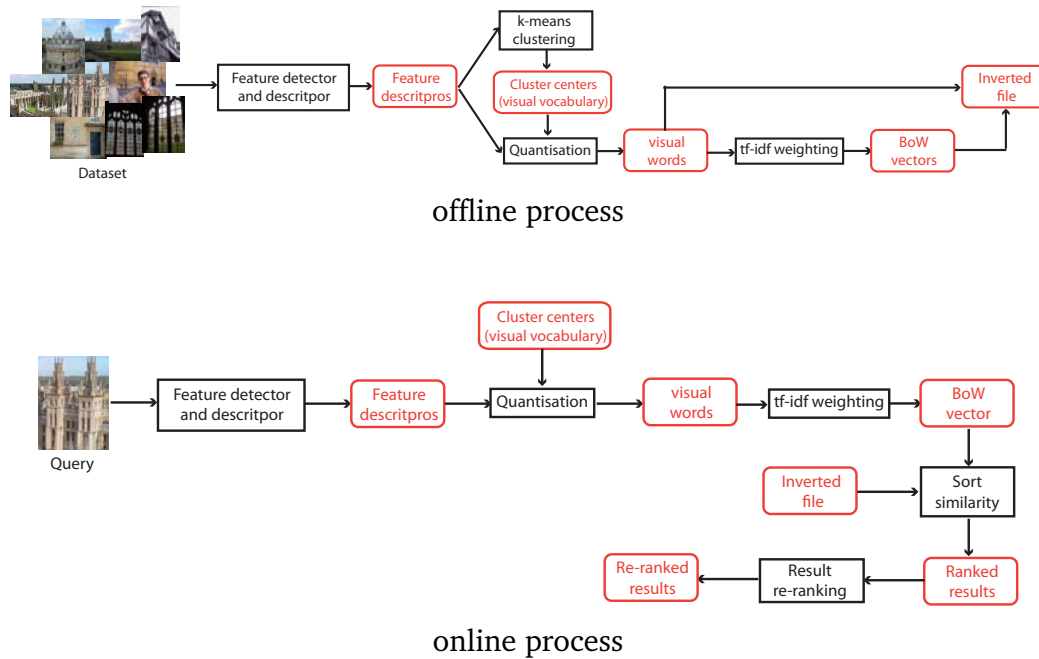


Figure 2.5: The baseline framework of BoW based retrieval system, in which the black boxes are the computation functions and the red boxes are the corresponding input/output data.

generate K cluster centres, known as visual words. Each feature descriptor is then assigned to the closest cluster centre, and labeled with a visual word ID. The original feature descriptors are ignored after this step (Section 2.2.1).

- Visual word weight scheme. The third step is to represent each dataset image as a weighted vector. This is similar to weighting words in a document in the text-retrieval methods. It uses the term frequency inverse document frequency (tf-idf) [156] weight scheme (Section 2.2.3, Eq. (2.8)).
- Inverted file structure. Finally, the dataset can be stored as an inverted file indexing structure. As illustrated in Table 2.2, the inverted file stores mapping from visual words to dataset images. This helps the retrieval system quickly to access the candidate images (Section 2.2.3).

The online process searches the dataset according to a given query. For a new query image, feature detector and descriptor are applied to obtain the visual word IDs. A BoW vector is generated for image similarity measure and ranking:

- Image similarity sorting. The images are ranked by sorting the normalized dot product between the given query vector and corresponding dataset

	Dataset	# images	# features	vocabulary size	# feat / # img
A	Oxford 5K	5062	12.7M	1M	2.5K
	Paris 6K	6412	20.7M	1M	3.2K
	Rome	2337	4.7M	200K	2.0K
	Holiday	1491	4.6M	200K	3.1K
B	Caltech Categories	3639	3.5M	500K	1.0 K
	ImageNet (animal)	8311	7.8 M	500K	0.9K
C	Oxford 105K	105K	170M	1M	1.6K
	Oxford 1M	1M	1589 M	1M	1.6K

Table 2.3: Evaluation dataset summary. The features are obtained by Hessian-affine feature detector [95] (with threshold $t = 200$) and 128-D SIFT descriptor [85].

image vectors. The image ranking is efficient because it only needs to calculate the visual words in both images (q and d).

- Result re-ranking. The retrieval results returned by the retrieval system can be further examined by some similarity criteria, *i.e.* spatial consistency recovered from raw features [106]. As a result, some highly ranked false positives can be moved lower in the ranked results.

2.3 Datasets, evaluation and the baseline system

This section introduces the datasets used in this thesis for evaluating various retrieval methods, evaluation criteria borrowed from information retrieval and reports details of our implementation of the baseline method [106], which is based on the architecture described in Section 2.2.4.

2.3.1 Evaluation datasets

We evaluate the retrieval performance on some commonly used datasets, which can be categorized into three groups, as outlined in Table 2.3. The datasets in **Group A** contain rigid, unique objects, *e.g.* famous landmarks, such as can be crawled from Internet.

- Oxford 5K dataset [4] consists of 5062 image collected from FLICKR by searching 11 distinctive Oxford landmarks. Each landmark contains 5 different queries, each of which is within an associated bounding box. Each image is annotated with one of the 11 ground truth buildings as listed in

Table 2.4. It also is annotated with one of the four kinds of visual condition: “good” (clear image of the object), “ok” (more than 25% of the object is clear visible), “junk” (less than 25% of the object is clear visible) or “unseen” (the object is not present). The “good” and “ok” images are treated as positive samples, and the “unseen” images are treated as negative samples. In addition, the “junk” images are not considered in evaluating the retrieval performance. Examples of some images labeled as “good” and “junk” are illustrated in Table 2.5.

- Paris 6K dataset [4] is organized in a similar fashion to Oxford 5K dataset, containing 6412 images by searching 11 distinctive landmarks in Paris. The text tags used in organizing Oxford 5K and Paris 6K datasets are listed in Table 2.4.
- Rome dataset is a subset of scene summarization [128], containing 2337 images. Each image is also annotated with a landmark.
- Holiday dataset contains 1491 images of a wider variety of objects, of which 500 are queries [62].

The datasets in **Group B** contain less geometric information than Group A: Caltech Categories [6] (3639 images) and a subset of animal images from ImageNet [3] (8311 images). Each of them uses 20 images randomly chosen as queries. Some of the queries are shown in Table 2.6. The datasets in **Group C** are used to test the scalability of our method: Oxford 105K and Oxford 1M. These two datasets are composed of Oxford 5K dataset images and 100K/1M FLICKR images downloaded from [1].

In this thesis, Oxford 5K and Paris 6K datasets are commonly used for parameter setting and retrieval results comparison, while other datasets are used either test the scalability or flexibility.

2.3.2 Evaluation criterion

Many evaluation criteria used in image retrieval are borrowed from information retrieval. Among them, the main tool is the precision and recall measures: **Precision** is the fraction of retrieved documents that are relevant, while **recall** is the fraction of relevant documents that are retrieved. Mathematically, they are

Oxford		Paris	
Landmark	# images	Landmark	# images
All souls	78	Defense	117
Ashmolean	25	Eiffel	289
Balliol	12	Invalides	198
Bodleian	24	Louvre	152
Christ church	78	Moulinrouge	237
Cornmarket	9	Museedorsay	72
Hertford	24	Notredame	119
Keble	7	Pantheon	126
Magdalen	54	Pompidou	51
Pitt river	7	Sacrecoeur	149
Radcliffe camera	221	Triomphe	281

Table 2.4: Text queries used to annotate each image in Oxford and Paris datasets.





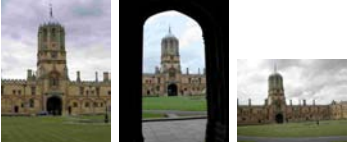

Query	“good” images	“junk” images
		
		






Table 2.5: Examples of “good” and “junk” images from the Oxford 5K dataset.

defined as follows:

$$\text{Precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \quad (2.10)$$

$$\text{Recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \quad (2.11)$$

Given a query document, the retrieval system returns a set of ranked results. A precision-recall (PR) curve describes precision and recall for every possible truncation of the ranked result list. The PR curve is generated as follows: the initial precision value is set as 1 and recall value is set as 0; the k th precision and recall are calculated by the truncated k results and plotted on the curve, as shown in Figure 2.6. The accuracy of a query results is defined as the area under the PR curve, known as average precision (AP). An AP score ranges between 0 and 1. An

Caltech Categories		ImageNet	
airplane		animal 1	
motorbike		animal 2	
face		animal 3	

NOTE:
 This figure/table/image has been removed
 to comply with copyright regulations.
 It is included in the print copy of the thesis
 held by the University of Adelaide Library.

Table 2.6: Query examples of Caltech Categories and ImageNet datasets.

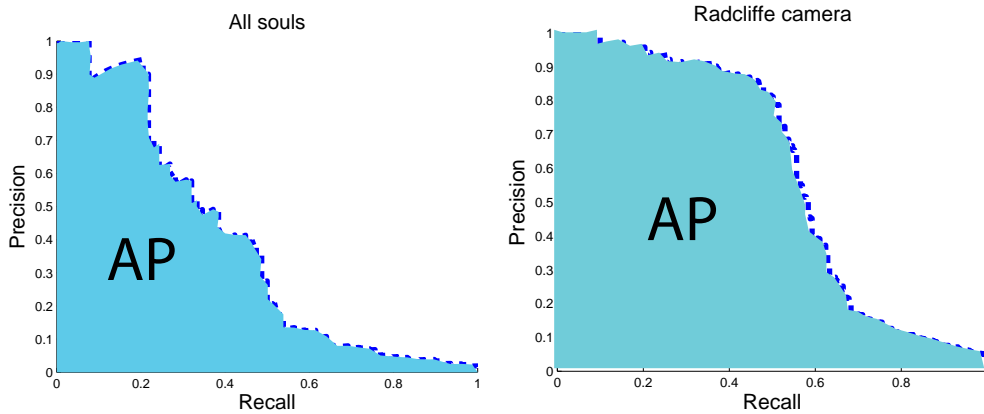


Figure 2.6: Illustration of precision-recall (PR) curve. The area under the PR curves is the average precision (AP). As a result, the accuracy of *Radcliffe camera* is greater than the accuracy of *All souls*.

ideal PR curve has precision as 1 over all recall levels, such that the AP score is 1 (maximum). As illustrated in Figure 2.6, the accuracy of *All souls* is lower than *Radcliffe camera* because the area under the PR curve of *All souls* is smaller than the one of *Radcliffe camera*. In practice, evaluation runs multiple kinds of queries to test the stability of the retrieval performance, and takes the mean average precision (mAP), as the final measurement of the retrieval performance.

Algorithm 2 The baseline retrieval system outline.

1. Vocabulary construction (offline)
 - 1.1 Local region detection on each dataset image (Section 2.1.1).
 - 1.2 Extract SIFT features from each region (Section 2.1.1).
 - 1.3 Cluster all SIFT features into N cluster centres (Section 2.2.2).
 - 1.4 Assign each SIFT feature to its closest cluster centre (visual word ID) (Section 2.2.2).
 2. BoW representation (offline)
 - 2.1 Compute tf-idf vector for each dataset image (Section 2.2.3).
 - 2.2 Construct inverted file for the dataset (Section 2.2.3).
 3. Object retrieval (online)
 - 3.1 Local region detection on query image (Section 2.1.1).
 - 3.2 Extract SIFT feature from query image (Section 2.1.1).
 - 3.3 Compute tf-idf vector for query image (Section 2.2.3).
 - 3.4 Retrieve image candidates using inverted file (Section 2.2.3).
 - 3.5 Compute image similarity between pairwise query and dataset images (Section 2.2.3).
 - 3.6 Rank dataset images by sorting the image similarity.
 - 3.7 (Optional) Spatially verify the top ranked results (Section 2.4.4).
-

2.3.3 The baseline retrieval system

We consider the searching system with the BoW model and a standard tf-idf weighting as our baseline. The system is constructed under the framework discussed in Section 2.2.4. The implementation exactly follows the experiments configuration reported in [106]. The pipeline of the system is detailed in Algorithm 2, which is separated into three parts: *i*) vocabulary construction; *ii*) BoW representation; and *iii*) object retrieval.

We report the details of our implementation of Algorithm 2 as follows. In the vocabulary construction, we use Hessian-affine invariant detector [94] for local region detection. The threshold is set as $t = 200$, for example, it generates on average 2500 local features of each image on Oxford 5K dataset². The clustering of SIFT features uses approximate k-means implemented by [5], where the number of visual words is $N = 10^6$ by default. The quantisation of raw features uses approximate nearest neighbor search (FLANN) [97]. In this step, the baseline system assigns each feature to its closest cluster centre, *i.e.* each raw feature only has one visual word ID. The BoW representation weights each visual word in an image by the tf-idf scheme. After that, images are stored in an inverted file structure. In online object retrieval, the dataset images are ranked by descending

²Note that the average number of local features will vary on different datasets.

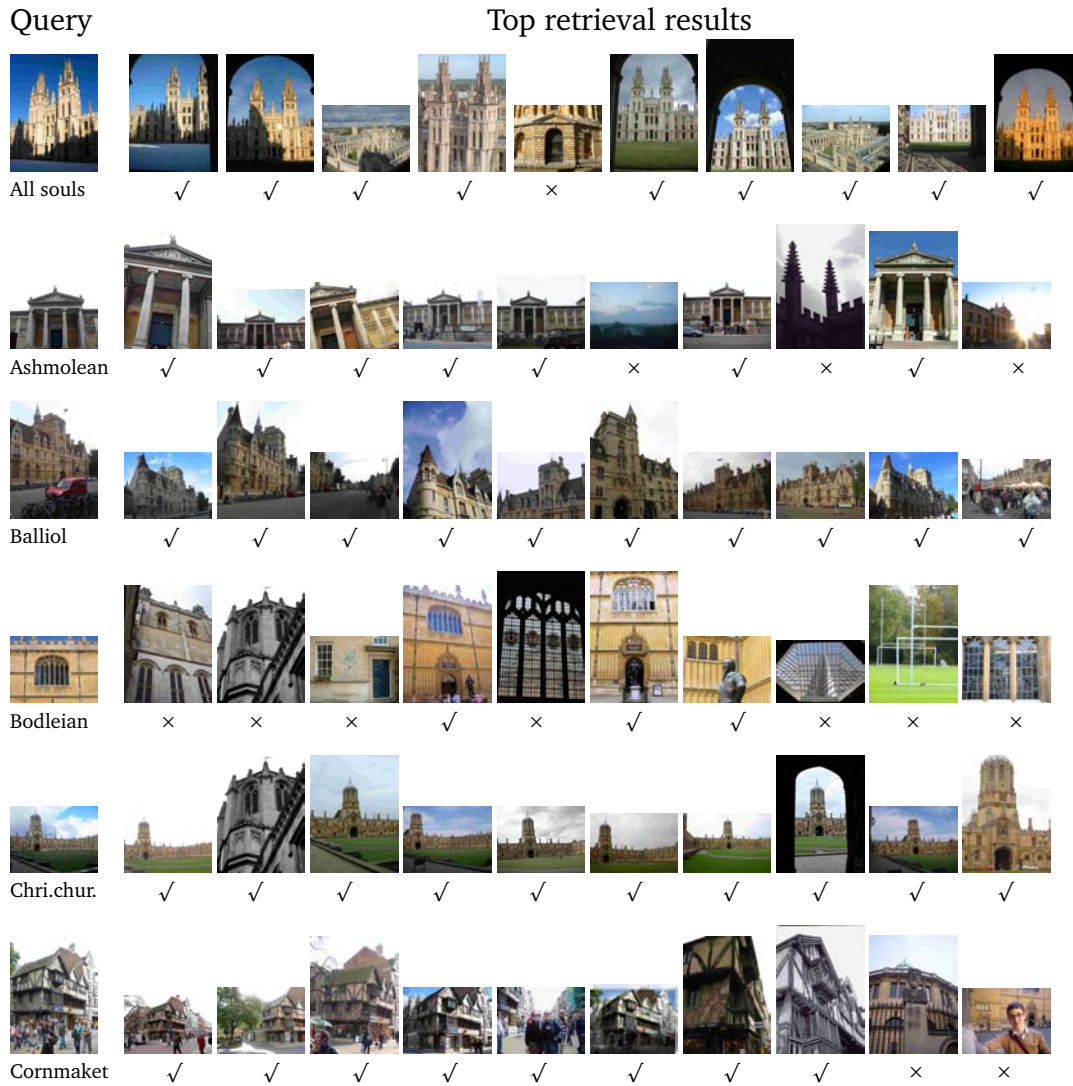


Figure 2.7: Top retrieval results of the baseline method.

normalized dot product similarity [130] to the query image.

The retrieval performance of our implementation is very close to a number previous implementations [106, 108, 37]. Under our implementation, the Oxford 5K dataset achieves 0.612 in mAP score, while the Paris 6K dataset achieves 0.639. The difference between these mAP scores and the ones reported in [106, 108, 37] is less than 1%. We show some top ranked retrieval results of baseline in Figure 2.7, and indicate true and false positives below each row of the retrieval results. The following chapters will compare to the same query instances as shown here.

2.4 Algorithms for retrieval performance improvement

Although the BoW retrieval system is efficient, it might introduce errors during each module as illustrated in Figure 2.7. These errors can happen in the following ways:

- **Features:** local features do not capture all variation resulting from viewpoint and lighting changes, and detection is error-prone.
- **Vocabulary building and feature quantisation:** in order to scale to large collection of images, approximate k-means is adopted in building the vocabulary. As a result, matching features may be assigned to different visual words, and conversely features assigned to the same word may represent different content, due to the approximation.
- **Image similarity measure:** typically the dot product is used to measure image similarity. Thus, only co-occurrence of the same visual word in both images contributes to similarity. As a result, different words are considered infinitely distant even though they may be neighbors in feature space.

These failings have received attentions over the past years, leading to many refinements of the BoW model. These methods have achieved more accurate results than the baseline. It is well known that the performance gain strongly depends on the modules of the retrieval pipeline [25]: feature extraction, vocabulary construction, quantisation of raw features and image similarity measure. Each of them plays an important role in the retrieval system, and its effect influences the following modules. A detailed evaluation can be found in the literature [25]. In this section, we discuss methods aiming to increase the accuracy of retrieval results. Typically, they can be categorized according to improvement of different modules of the baseline system: *i*) local region detector; *ii*) local feature descriptor; *iii*) vocabulary building; *iv*) BoW representation; *v*) similarity measure. We begin by following the order of the standard pipeline (Figure 2.5).

2.4.1 Improvement of local region detector

The BoW object retrieval system is based on local feature descriptors, such that foreground information can be captured. The detection of local features usually

proceeds in two steps: local interest regions, which are obtained by one of the region detectors summarized in [95]; local feature descriptors, which describe each detected region.

This section discusses improvement of local feature detector in an object retrieval system, while the improvement of descriptors is discussed later. The baseline system adopts affine invariant region detector (Hessian-affine region detector) [95]. This kind of methods has been proved to be invariant to transformations: viewpoint changes, zooming, rotation, image blurring, compression or lighting changes. However, the detected regions still may not correspond to the same pre-image, especially when viewpoint changes dramatically. For example, the repeatability of Hessian-affine regions is less than 30% when viewpoint angle between two images is greater than 60, as reported in [95].

There is little work aiming to improve the performance of affine invariant region detectors. Sivic *et al.* [129] proposes to use more than one kind of region detector because a particular type of detector might not work at all. In [129], an affine invariant region detector [93] (focusing on signal variation in more than one direction) works together with MSER [89] (focusing on high contrast extended regions) in tracking objects in video shots. This enables correct matches between video frames. In this thesis, we follow [106] to use Hessian-affine detector and do not focus on the choice of feature detector and its improvement.

2.4.2 Improvement of local feature descriptor

There is some research focusing on reducing the error by learning better visual descriptors (*e.g.* than SIFT) before the quantisation process. Early descriptor learning methods can be found in literatures [78, 152, 126, 11, 151], and are mostly designed for object classification. Among them, Winder and Brown [152] introduces a framework to automate parameter tuning in descriptors algorithms, *e.g.* SIFT and GLOH. It learns good choice for parameters by a 3D ground truth training dataset, which has accurate match/non-match related to interest points from multiple images. Based on [152], Winder *et al.* [151] develops a new training set on patches centered on real interest points. It learns the optimal descriptors with a DAISY configuration [139]. A PCA dimension reduction is applied further in [139] to increase the discriminative power of visual words, while reducing the memory storage.

Although these methods are efficient in a number of classification problems, they need to know clean, labeled data to indicate pairs of points belonging to the same class or not. Hua *et al.* [57] presents method to generate training set from unlabeled data to be matched by the Photo Tourism system [133]. Similar to [151, 57], a descriptor learning method is presented in [108] for object retrieval. In contrast to previous methods, the method presented in [108] enables people to learn discriminate local features from training data automatically collected. By using the training data, the descriptors are projected into a new Euclidean space such that matching descriptors are more likely to be assigned in the same cluster while non-matching descriptors are assigned to different clusters. As a result, it benefits the following vocabulary building and quantisation steps to generate more accurate BoW representation of an image. In this thesis, we use SIFT feature descriptor without dimension reduction.

2.4.3 *Improvement of visual vocabulary*

In a BoW retrieval system, vocabulary construction follows local feature detection and description. In this stage (as shown in Figure 2.5 offline process), data clustering aims to partition the feature space into informative regions, such that matching features are mapped to the same visual word, while non-matching features are mapped to different visual words. The encoding of feature descriptors is important, which requires a discriminative visual vocabulary.

K-means clustering and its variants are commonly used in constructing the visual vocabulary. As discussed in Section 2.2.2, these methods mainly focus on the scalability of the vocabulary constructing, including hierarchical k-means (HKM) and approximate k-means (AKM) presented in [100, 106]. They successfully reduce the computation cost, but lead to inaccurate clustering results as well. In these k-means clustering methods, the distance between two feature descriptors is measured by Euclidean distance. The product quantisation [64] addresses the clustering problem in an alternative way. It decomposes the feature space into low-dimensional subspace and then quantizes each subspace separately. In order to do that, a product quantisation borrowed from source coding is used to map a given vector (split into some distinct sub vectors) to a set of centroids. As a result, the distance between two feature vectors is estimated efficiently by vector-to-centroid distance.

Moreover, matched descriptors might be located in different clusters (mapped to visual words). Literatures [62, 63] present a Hamming embedding (HE) method to more precisely represent matching descriptors by combination of k-means clustering and binary vector signatures. Other methods [65, 66, 10] compress local feature descriptors to vectors of locally aggregate descriptors (VLAD), which have lower dimensions compared to the standard vocabulary constructing results. The reduced vectors keep the discriminative power of original local features by jointly optimisation of dimensionality reduction and indexing.

Recent work aims to embed spatial information in constructing the visual vocabulary [96, 88]. These methods adopt inter-feature relationships discovered by some pre-processes [96, 88]. The inter-feature relationships can also be discovered by tracking and matching features. For example, corresponding patches are tracked [96] in an unsupervised manner, similar to [7]. The visual words are generated by fine partitioning of descriptor space, whose process is data dependent. For each partition (visual word), the method also learns the probability to contain descriptor of matching features. Similar to [96], matching features are tracked offline and a track-graph is constructed to encode feature transformation observed from image sequences [88]. The quantisation is improved by voting a feature to its multiple closest words found in the tracked graph.

In this thesis, we use AKM to build the visual vocabulary and do not investigate the scalability of vocabulary construction.

2.4.4 Improvement of BoW representation

The retrieval system measures the distance of dataset images to the given query using their BoW vectors. However, errors might occur in feature quantisation, *i.e.* mapping matching features to different visual words, and non-matching features to the same visual word.

There have been a large number of methods proposed to address the quantisation errors. Typically, these methods can be categorized into two groups: *i)* methods mapping a feature to multiple visual words (soft-assignment). *ii)* methods adding spatial information in the BoW representation.

Soft-assignment in quantisation

The standard BoW model computes image similarity between pairwise images on a word-to-word basis [130]: a visual word contributes to the similarity score when it occurs in both query and dataset images. However, each feature is hard assigned to a visual word ID in the standard method; features that are not assigned in the same visual word ID will not contribute to the similarity score. This causes intolerance of varying image condition. Moreover, the hard-assignment based on ANN search [97] (FLANN) is likely to refer features near the cluster boundary to far away cluster centre, instead of the one close to the feature, as noticed in [107].

Most techniques address the hard-assignment problem by multiple mapping. This step happens after clustering is executed. Soft-assignment [107] maps each feature (*e.g.* 128-D SIFT descriptors) to multiple neighboring visual words, in which the weight of each visual word is proportional to the L2 distance between the feature and the candidate cluster centres. This allows some uncertainty in quantisation. The visual word ambiguity [146] extends the soft-assignment method by investigating the distribution of visual word after quantisation. It uses kernel density estimation to replace the histogram accounting in order to smooth the local neighborhood. However, this kind of method used in image representation leads to the increase in both memory usage and run time.

Adding spatial information in the BoW representation

Although spatial information is ignored in the standard BoW representation, it can be re-introduced after quantisation to enhance the BoW representation. Spatial pyramid [77] is one of the earliest methods to pay attentions to the spatial information after quantisation. The method is based on a global geometric correspondence, where features are grouped within increasing sub-region of the image space. Thus, the orderless visual words are matched with different levels of spatial histograms. Followed by [77], a number of methods consider the spatial layout information of each feature in the image space. The spatial pyramid is generalized by [16] in two kinds of extensions: a pyramid histogram of visual words (PHOW) for appearance and a pyramid HOG (PHOG) descriptor for shape. In these methods, the features used are dense SIFT.

Recent methods organize the spatial information as “visual phrases” struc-

ture, which is a meaningful high-order pattern of visual words. The method proposed in [158] organizes visual phrases according to spatial co-occurrence pattern of visual words. Spatial-bag-of-words [22] captures the invariance of object translation, rather than the simple spatial frequency. It encodes ordered bag-of-feature representation by projection of local features into different directions. The visual phrases are selected from K nearest neighbors in a fixed spatial region [138, 46]. In these methods, visual phrases are based on spatial co-occurrence of visual words in dataset images. The visual phrase can also preserve geometric information of visual words, *e.g.* the geometry-phrase [159], which organizes groups of visual words as “phrases” if these visual words have similar spatial offset in dataset images.

A spatial re-ranking process is applied to re-introduce the spatial information in the post-process, as shown in Figure 2.5 (result re-ranking). The spatial verification [106] is widely used to recover spatial location of target objects, as well as improve the retrieval results by removing highly ranked false positives. It is noticed that pairwise images are very likely to contain the same object if there is an affine transformation between a number of the corresponding features. In order to examine that, spatial verification utilizes RANSAC [54] to estimate homography transformation between a pair of query/dataset images [106], and assumes that a common object exists if there are minimum number of inliers returned by RANSAC. The detailed process is described in Algorithm 3, where a homography transformation can be 3 dof, 4 dof, and 5 dof as listed in Table 2.7, allowing increasing degree of freedom.

The spatial verification is a powerful tool to distinguish true/false positives, however, it also increases the memory cost because geometry information of raw features is recovered, as well as run time cost because it needs extra computation of spatial consistency examination. Local geometry [105] reduces the memory required to represent each feature while without drop in accuracy. The spatial verification is also used for a number of query expansion methods, as discussed below.

Query expansion

The initial retrieval results can be used to construct a new, richer query model, which is known as *relevance feedback* in text retrieval [119, 21]. These methods

Transformation	dof	Matrix
Translation and isotropic scaling	3	$\begin{bmatrix} a & 0 & t_x \\ 0 & a & t_y \\ 0 & 0 & 1 \end{bmatrix}$
Translation and anisotropic scaling	4	$\begin{bmatrix} a & 0 & t_x \\ 0 & b & t_y \\ 0 & 0 & 1 \end{bmatrix}$
Translation and vertical preserving shear	5	$\begin{bmatrix} a & 0 & t_x \\ b & c & t_y \\ 0 & 0 & 1 \end{bmatrix}$

Table 2.7: Homography transformation types used in spatial verification [106] (degree-of-freedom (dof)): 3 dof, 4 dof and 5 dof.

Algorithm 3 Spatial verification outline.

- 1: **Input:** Minimum number of inliers t , top-K retrieval results, and query image q .
 - 2: **Output:** Re-ranked top-K results.
 - 3: **Initialize:** $T := \{t_k\}_{k=1}^K$ are sorted top-K dot product similarity, $S := \{s_k\}_{k=1}^K$ are all zero entries, $i \leftarrow 0, j \leftarrow 0$.
 - 4: **while** $i < K$ and $j < 20$ **do**
 - 5: Obtain tentative correspondences between image v_i and query q .
 - 6: Use elliptical shape and (x,y) position of raw feature to calculate the homography matrix in RANSAC.
 - 7: **if** # of inliers $\geq t$ **then**
 - 8: $s_k = \#$ of inliers
 - 9: **else**
 - 10: $j \leftarrow j + 1$
 - 11: **end if**
 - 12: $i \leftarrow i + 1$
 - 13: **end while**
 - 14: Re-rank top-K retrieval results by sorting new similarity $U := \{u_k\}_{k=1}^K$, where $u_k = t_k + s_k$.
-

usually use highly ranked (relevant) results to enrich the original query, then the dataset is re-queried by the refined query model to obtain more accurate results. In image domain, these methods extend the query by adding query relevant visual words. This can be done by *blind* query expansion or *automatic* query expansion [37]. The blind query expansion method selects m top ranked results (e.g. $m = 5$) from initial retrieved results. The new query averages the tf vectors and is added to the original query vector. This is defined as the “QE baseline” [37], in which dataset images are re-queried by the combined query vector $\mathbf{d}_{QEbaseline}$:

$$\mathbf{d}_{QEbaseline} = \mathbf{d}_0 + \sum_{i=1}^m \mathbf{d}_i \quad (2.12)$$

where \mathbf{d}_0 is a normalized tf-idf vector of the query and \mathbf{d}_i is the corresponding BoW vectors (tf-idf weighted) of the top ranked results. The automatic query expansion method collects spatial verified visual words corresponding to the given query region. A new query is computed by averaging the verified results of the original query:

$$\mathbf{d}_{AQE} = \frac{1}{m+1}(\mathbf{d}_0 + \sum_{i=1}^m \mathbf{d}_i) \quad (2.13)$$

where \mathbf{d}_i is a normalized tf vector of verified region and m is the number of verified images found by spatial verification (usually at most 50). In this method, known as average query expansion (AQE), the spatial information in detected regions (estimated by the Homography transformation) is combined with the query region. In both QE baseline and AQE, the dataset images are ranked by the normalized dot product similarity, as for the baseline method [130, 106].

Discriminative query expansion The image similarity in most BoW retrieval systems is defined as the dot product distance between two tf-idf vectors, and the dataset images are ranked by the descending dot product scores. Instead of computing image similarity as a metric, discriminative query expansion (DQE) [9] ranks dataset images by a trained classifier. It trains a linear-SVM by positive data collected from the (up to) top-50 retrieved results from AQE and negative data collected from bottom-200 retrieved results. The dataset images are ranked by their signed distance to the decision boundary found by linear-SVM.

Dataset side feature augmentation Dataset side feature augmentation is a useful tool for increasing recall [88, 9]. This kind of method can be thought of as query expansion on each dataset image, augmenting the BoW vector with relevant image features learnt offline. Turcot and Lowe [142] collect feature counts for related images. However, this method is unstable because there are a significant number of features that may not be present in the original query image. Spatial feature augmentation [9] (SPAUG) addresses this shortcoming by taking visibility of the augmenting features into account. The method spatially verifies the features by a homography in the augmented images, instead of all neighboring features. However, SPAUG also increases storage requirements.

Automatic tf-idf failure recovery in query expansion It is noticed in [34] that the effect of AQE method depends on the quality of the shortlist generated by spatial verification. The retrieval performance of AQE will not be improved significantly if the shortlist does not have enough true positives. An alternative way to perform query expansion is proposed in [34]. The method aims to eliminate confusers [72] (features that correlate with both foreground and background information) in a query model, instead of adding query related features as in previous QE methods. Based on this, two improvements of spatial re-ranking are presented in [34] such that an object that is not clearly present in the initial query can be recovered: *i*) an incremental spatial verification (iSP) accounts for agreement with all previously verified images. *ii*) query expansion incorporating spatial context is then able to grow the query model with content outside the user-specified ROI.

We address the image representation improvement in Chapter 3.

2.4.5 Improvement of similarity measure

One of the fundamental issues in a retrieval system is to rank dataset images according to their visual similarity to the query. There is intensive study on visual similarity measure in literature. Typically, they can be categorized into two groups [155]: individual image similarity measure and semantic concept measure. Individual image similarity measures are usually based on feature descriptors, such as those in [131, 82]. Other image similarity measures aim to narrow the semantic gap occurring in retrieval, *e.g.* the semantic distance similarity [153, 12]. Conceptual similarity usually describes the semantic relationships between related images, for example learning the visual correlation between concepts [154, 155].

In this thesis, we focus on local features that are used to compute a BoW vector for each image. The BoW vector is very sparse and high dimensional data, *i.e.* only thousands of entries are nonzero in the 10^6 length vector. Therefore, image similarity measure needs to consider both accuracy and scalability. Typically, a dot product similarity [130] is used to measure similarity between query/dataset image pairs. Although it yields efficiency, the dot product similarity might fail in varied image conditions [130, 106]. Below we discuss some commonly used similarity measures to achieve accuracy or scalability.

Improvement of accuracy

The dot product similarity computes image similarity as a word-to-word match of the BoW vectors. However, it is intolerant to image condition changes. Note that the inter-word relatedness has not been considered in the dot product similarity. Intuitively, a cross-word comparison leads more accurate similarity measure, similar to the earth mover's distance (EMD). The EMD was first used to measure the distance between intensity images [103], and then for color-based similarity [117]. Grauman and Darrel [50] first uses it in comparing image similarity on local features, without quantisation. The efficiency is limited when the dataset size becomes large. The cross-bin comparison on quantised features (visual words) is then proposed in [67], which is based on the observation that the semantic similarity of visual words are available from image content. The semantic similarity is similar to text-based ontology WordNet [2]. An earth mover's distance (EMD) like constraint is proposed to model different semantic similarity of visual words. As a result, images are matched by cross-word similarity (CEMD).

Improvement of scalability

The dot product similarity also lacks efficiency when there are increasing number of queries or larger dataset. This is because the retrieval system needs to compute the score between each pair of query and dataset images returned by an inverted file. Instead of pairwise comparison, approximation of image similarity is an alternative. The approximation thus involves various approximate nearest neighbor (ANN) search methods.

Locality Sensitive Hashing (LSH) ANN has been intensively studied for decades, and has proposed a large amount of algorithms, which is not the focus of this thesis. Instead, we focus on Locality Sensitive Hashing (LSH) [48, 59, 41, 74] used for nearest neighbor search, which is proposed for high dimensional data. The LSH methods can quickly return the nearest neighbor data with a high probability. It is based on a simple idea that should remain close after random projection to a lower dimensional subspace. The projected data points are inserted into buckets in a hash table if they are located in the same bin in the lower dimensional subspace. For a query data point, the LSH search only needs to test the data points that are

located in the same buckets together with the query points. Data points that are not located in the same buckets will be ignored. The computational cost is reduced. Moreover, the LSH search stops after searching a maximum number of candidates from the hash tables. In summary, the hash tables used in LSH search are created as follows: *i*) form L independent projections (hash functions) in parallel. *ii*) each data point is associated with L hash keys produced by the hash functions. *iii*) pool each hash key to a bucket in the hash table. Note that the LSH family covers a variety of methods to construct the hash functions, *e.g.* bit sampling for hamming distance [59], random projection to approximate the cosine distance, p -Stable distributions [41], and min-wise independent permutation [19].

Near duplicate image search with LSH In the image domain, accurately ranking the whole dataset of images is expensive. Instead, there are many applications only need to find images that to be identical or very similar to the same scene or object [38], known as *near duplicate images*. Efficient indexing for near duplicate search is proposed in [70, 86, 36]. These methods search nearest neighbors based on either color histogram [86, 36] or local features (PCA-SIFT) [70]. The hash functions used in these methods are random projections [36], hamming space [70], and multi-probe [86]. These hash functions work well on relatively low dimensional space, *e.g.* PCA-SIFT and color histogram, but fail in effectiveness on the BoW representation (10^6 dimensions). Other methods learn short descriptors for hashing method [141] or extend LSH to Mahalanobis distance [60].

The application of near duplicate image search to BoW representation needs to consider sparsity. Min-Hash algorithm [19], therefore, is designed for the BoW representation [36]. Chum *et al.* [36] is among the first to apply min-Hash in near duplicate image search. It represents each image as a sparse set of visual words, and the image similarity is measured by the set overlap (the number of common visual words shared in a pair of images) It enables efficient retrieval of the BoW vectors as follows: *i*) a hash key is extracted from each dataset image by random permutation of the visual vocabulary. *ii*) each dataset image is composed of a set of hash keys by repeating *i*). *iii*) the image similarity is estimated by the number of common hash keys in two images divided by total number of hash keys. *iv*) Optionally, the min-Hash keys are grouped into a sketch and then search images from hash table by identical sketches.

This method successfully reduces the run time of retrieval process, but some relevant information is not preserved in the selected set of visual words. The method is extended in [38] to more accurately measure the image similarity by taking the advantage of tf-idf weighting in min-Hash algorithm, without increasing computational cost. The similarity between two images is proportional to weighted set overlap (tf) or approximate histogram intersection (tf-idf). Moreover, the geometry information is added in min-Hash in [35], where each visual word is hashed not only by the ordered visual words but also depends on the feature's scale and spatial location. The geometry min-Hash leads to more accurate results than the previous min-Hash [36, 38].

We address the image similarity measure improvement in this thesis in Chapter 4.

2.4.6 Retrieval performance summary

Table 2.8 summarizes the retrieval performance of improvement methods discussed above³. These methods are separated based on methods that modify the baseline before the query is executed (pre-process), or after (post-process). The first group of methods (Group A, methods that modify the baseline before the query is executed) focus on improvement of the BoW representation and similarity measure, while the second group of methods (Group B, methods that modify the baseline after the query is executed) focus on improving the initial retrieval results with a re-ranking or re-query process.

The retrieval results in Table 2.8 are based on our implementation, except the mAP scores cited from literature. We implement the following methods in Table 2.8: the baseline, soft assignment, spatial verification, QE Baseline, AQE and DQE. Our implementation of baseline (as described in Section 2.3.3) follows [106], thus obtains very close mAP scores to the results reported in [106]. The improvement methods are therefore also slightly but consistently different to the results reported in the original paper. The retrieval results of other methods are cited from the original papers.

³Note that we only include retrieval results run on three public datasets: Oxford 5K, Paris 6K and Oxford 105K.

Methods		Oxford 5K	Paris 6K	Oxford 105K
Baseline [106]		0.612	0.639	0.515
A	Descriptor learning (non-linear) [108]	0.662 [108]	0.678 [108]	0.541 [108]
	Soft-assignment [107]	0.673 [107]	0.660	N/A
	Geometry-Preserving [159]	0.696 [159]	N/A	0.604 [159]
	Fine vocabulary [96]	0.742 [96]	0.749 [96]	N/A
	AUG [142]	0.776 [9]	N/A	0.711 [9]
	SPAUG [9]	0.785 [9]	N/A	0.723 [9]
B	Spatial verification [106]	0.649	0.655	0.571
	QE Baseline [37]	0.708	0.736	0.679
	iSP [34]	0.741 [34]	0.769 [34]	0.649 [34]
	Local geometry [105]	0.788 [105]	0.634 [105]	0.725 [105]
	AQE [37]	0.806	0.769	0.767
	DQE [9]	0.798	0.783	0.809
	Hello neighbors [114]	0.814 [114]	0.803 [114]	0.767 [114]
	Total recall II [34]	0.827 [34]	0.805 [34]	0.767 [34]

Table 2.8: Retrieval performance summary of various improvement methods (mAP). Group A: retrieval results of methods that modify the baseline before the query is executed (pre-process). Group B: retrieval results of methods that modify the baseline after the query is executed (post-process). Note that we cite the retrieval results of AUG [142] from literature [9].

2.5 Application of object retrieval

Large scale object retrieval has attracted more and more interest in recent years, because many computer vision and pattern recognition tasks are cast as an image matching problem. The retrieved images can be used in a variety of related applications.

2.5.1 Image dataset mining

Intuitively, dataset images can be linked with a matching graph [111], in which each node represents a dataset image and an edge links two nodes if they contain the same object. The weight of an edge is proportional to image similarity between the corresponding images, for example, the number of inliers detected by spatial verification of pairwise images. As a result, a variety of methods used in data mining can be applied to discover spatially related images in the dataset. The matching graph is used in the following dataset mining problems.

Firstly, the matching graph enables automatic organization of unordered images. Images containing the same object can be grouped by some clustering

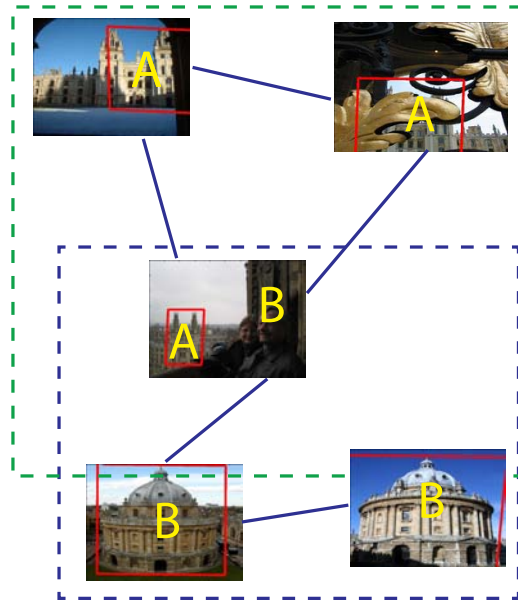


Figure 2.8: Example of connected image groups: All souls and Radcliffe camera. The image in the intersection of two groups connects two buildings: All souls and Radcliffe camera.

techniques, *e.g.* graph cut [111, 26] or mode-seeking [31]. This results in similar scene summarization [128] such that users can see “canonical views” of each object (building), as illustrated in Figure 2.8. Note that the automatic organization by a matching graph is different from [128] in the manner of modeling linked images.

Secondly, topics (themes) of the dataset can be extracted based on the matching graph. A geometric latent Dirichlet allocation (gLDA) [109, 110] classifies features by latent topics, which considers both the visual word frequency in corpus and the spatial layout information.

In contrast to the methods built on the matching graph, the data mining method proposed in [33] uses min-hash to detect near duplicated images as visual queries. The cluster of images are obtained as transitive closures of sets of spatially overlapping images. Moreover, the method is adaptive to the growing size of dataset.

2.5.2 3D reconstruction from retrieved images

With the increasing power of Internet image sharing websites, *e.g.* FLICKR, it is possible to collect large amount of photographs of a city, popular site, facade, and so forth, at home. A fast object retrieval system is useful in location recognition and 3D reconstruction of buildings from these Internet collected images.

Photo tourism system [133] uses the BoW retrieval system to browse large collection of photographs in 3D. The system automatically returns the locations and annotates image details. Because the photographs are usually taken by different people sharing on the Internet, viewpoints often cluster along certain paths. A system presented in [132] discovers a range of such paths and automatically computes the optimal paths between these views. Location recognition [80] uses the recovered structure to explore useful information about image and image features. It can predict which view will be the most common, which feature point is most reliable, etc.

The project described in [7] reconstructs entire cities from images retrieved from the Internet. It richly captures and explores the 3D shape of the city from millions of photographs. The challenge is to find image matches using common points, such that these points can be used to compute the 3D structure of registered images. In this project, a number of object retrieval related methods are applied, *e.g.* spatial verification to find the location of target objects. An efficient system is described in [44], in which dense 3D reconstruction is completed on a single PC in a day.

2.6 Summary

In this chapter, we have reviewed the general content-based image retrieval (CBIR) system and pay attention to object retrieval. We first discuss the various visual descriptions used in the CBIR system, as well as the corresponding visual analysis methods. In order to recognize a target object from images, we build the object retrieval system on local features and a BoW model. We have described the definition of baseline BoW retrieval framework, evaluation datasets and criteria, and the state-of-the-art improvements that we will benchmark our contributions against for the rest of the thesis. Some applications of object retrieval results are also discussed in this chapter.

The BoW retrieval system is a powerful tool for efficient search of target objects from large collection of images. However, the retrieval accuracy is affected by each step of BoW model. In the following chapters, we aim to improve three steps of the BoW retrieval system: image representation, similarity measure and a re-ranking process.

BUILDING A VISUAL THESAURUS

CHAPTER III

In Chapter 2 we saw that an image retrieval system is built on a BoW model, where features are quantised into visual word IDs. Thus, each image is represented by a collection of visual word IDs instead of raw features [130]. The standard pipeline is efficient, however, quantisation also causes low level feature match errors. Thus, some false positive images (that do not contain the target object) would be ranked highly.

There has been intensive study on improving the retrieval performance proposed in the past decade. The improvement of the standard retrieval system needs to overcome several shortcomings introduced by quantisation:

- Loss of recall: information about the raw features is lost, so matching features may be assigned to different visual words. This can decrease the matching score of images that contain the same object and therefore reduce recall.
- Loss of precision: features that do not correspond to the same location may be assigned to the same word. This can increase the match score of images that do not contain the same object, and therefore reduce result precision.

In this chapter, we aim to address the loss of recall and precision by improving the image representation. In the standard BoW representation, the quantised visual words ignore the spatial relatedness between visual words as described in Section 2.4.4. We recover the spatial information of visual words to explore those have not been matched with the same visual word IDs. As a result, the query is enriched.

The spatial information of visual words is usually re-introduced as a post-process (*e.g.* spatial verification [106]), such that query/dataset image pairs are more accurately matched. In contrast, we aim to uncover the spatial relationships between visual words when forming the vocabulary. In this chapter, we propose

a novel spatial co-occurrence measure that can be embedded into the standard retrieval pipeline. We do this by creating a **visual thesaurus** that records spatial co-occurrence information for each pair of words in a vocabulary. Our method aims to introduce information as to how the features are spatially related to each other, and to utilize this spatial information to discover synonym relationships between words. This is analogous to a text thesaurus in that it describes a broad set of equivalence relationships between words.

Based on that, we explore a voting based method to include spatial latent visual words by performing a **spatial expansion** process: a query can be expanded by including latent words, which are highly correlated with those already present in the query. The correlation is deduced from the degree of spatial co-occurrence of the visual words. In this way, the query recall can be boosted and the retrieval accuracy is increased. The framework of spatial expansion method is shown in Figure 3.1, which is a standard BoW retrieval system with additional steps (indicated in the dash box) introduced by our spatial expansion method. Compared to the standard framework (Figure 2.5), we only need to form visual thesaurus offline and expand query words online. The other modules are unchanged. The details are discussed as follows.

3.1 Building a visual thesaurus

Regardless of the feature detector that is used, some features can be more reliably detected at different viewpoints than others. It is observed that while some features occur repeatedly in images of an object, others do not. Figure 3.2 shows a pair of features reliably detected throughout the dataset images. Although these features are assigned to different visual words (indicated by different color in Figure 3.2), each of them indicates presence of the other in an image. Visual thesaurus uses the stable features to learn the relationships between the unstable ones. These relationships are useful when the query patch has a large number of features, but critical when only a small number of features are present.

The visual thesaurus captures the spatial relationship among pairs of visual words. Assume that there are N visual words $\mathbf{W} := \{w_i\}_{i=1}^N$ in the vocabulary. The co-occurrence of a pair of words (w_i, w_j) can be recorded by the frequency of them appearing in the same image throughout the dataset. The outline of building a

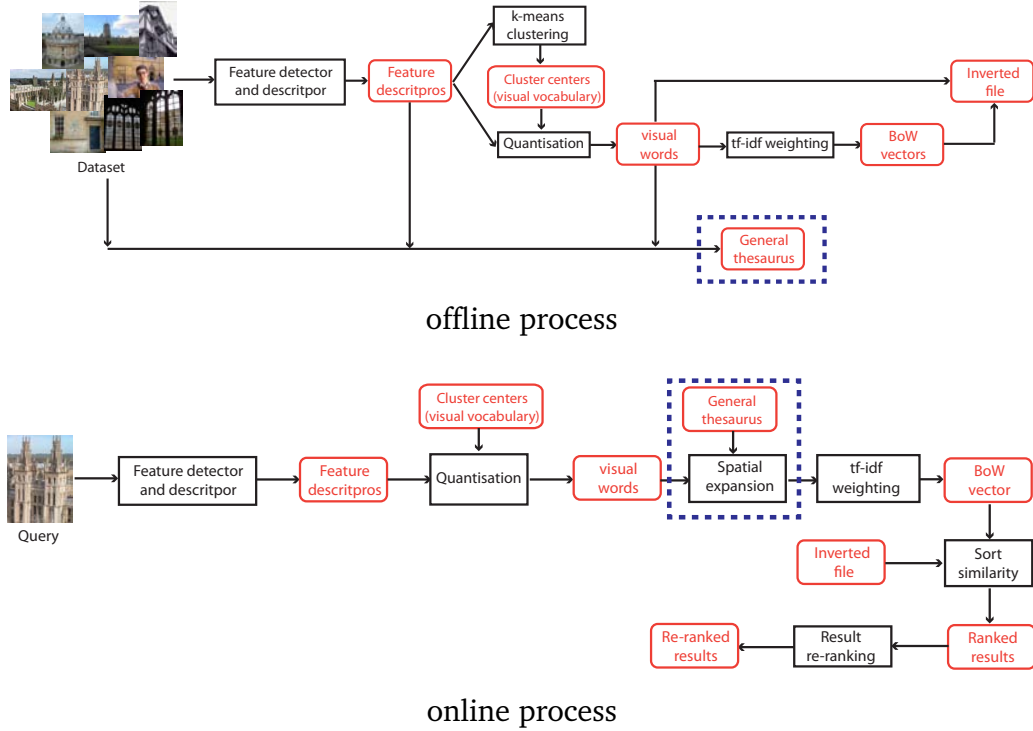


Figure 3.1: System framework of spatial expansion. This is the standard BoW retrieval system with new modules (indicated in dash box) introduced by our spatial expansion method.

visual thesaurus is described as follows:

1. For a given word w_i , we can recover its raw feature location in image space and obtain up to n nearest neighbors in a single image as $\mathbf{D}_i := \{w_j\}_{j=1}^n$, such that $\| \text{loc}(w_i) - \text{loc}(w_j) \| < \rho$, where $w_j \in \mathbf{W}$, ρ is the distance threshold limiting the nearest neighbor region and the function $\text{loc}(w_i)$ returns the (x, y) location of visual word w_i in the image space.
2. For all the images in the dataset \mathbf{V} , a visual thesaurus comprises of N histograms: $\mathbf{H} := \{h_i\}_{i=1}^N$. In each histogram h_i , the j^{th} entry, $h_i(j)$, indicates the frequency of co-occurrence between the word w_i and its nearest neighbor $w_j \in \mathbf{D}_i$ collected from the whole dataset. Note that the length of each histogram is the same as the vocabulary size N , but in general the histograms are sparse.

We generate two kinds of visual thesaurus in this thesis: a general thesaurus, \mathbf{F}_n , is based on all visual words in the vocabulary, as discussed below; an

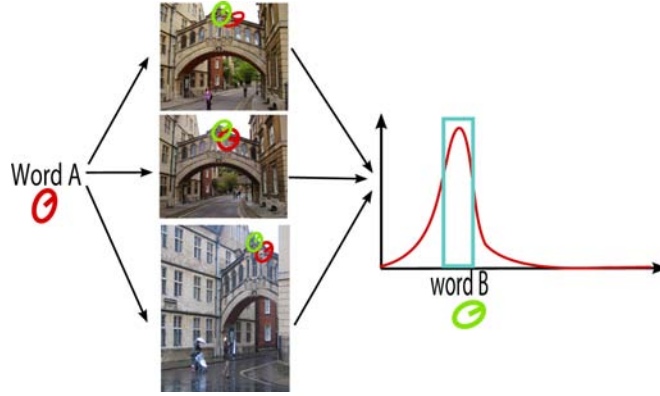


Figure 3.2: Example of a pair of frequently appearing visual words found by a general thesaurus. A pair of features frequently co-occur throughout the dataset images, but assigned to different visual word IDs (indicated by different color).

object-based thesaurus, F_n , only considers a subset of images to explore some discriminative words, which will be discussed in Section 3.5.

Building a general visual thesaurus We define a general thesaurus, F_n , as the set of histograms \mathbf{H} based on (up to) n nearest neighbors. To build F_n , the thesaurus histograms are computed from all the images in the dataset $\mathbf{V} := \{v_i\}_{i=1}^V$, where V is the dataset size. The spatial region for a feature (indexed by visual word ID w_i) in the image domain is defined as a n nearest neighbor set \mathbf{D}_i . The pseudo code for creating F_n is given in Algorithm 4. It proceeds as follows: the general thesaurus uses a set of N histograms to record the spatial co-occurrence of visual words, where N equals to the vocabulary size; the histograms are initialized beforehand, and are incremented when a co-occurrence is detected. Since the vocabulary is usually large (10^6), the histograms are sparse (most of the entries in the histograms will be empty). However in some cases, pairs of visual words can be found to often occur together, as shown in Figure 3.2. This leads to some peaks in the histograms, which indicates frequent co-occurrence of the words in the dataset. The simplest case is an F_1 thesaurus which records a pair of visual words that often occur as nearest neighbors (*i.e.* the words A and B shown in Figure 3.2).

Algorithm 4 Forming an F_n general thesaurus.

```
1: Input: Dataset images  $V$ .
2: Output: A general thesaurus  $F_n$ .
3: Initialize:  $F_n := \{h_i\}_{i=1}^N$  are all zero entries.

4: for all Image  $v \in V$  do
5:   for all Word  $w_i$  in image  $v$  do
6:     Obtain up to  $n$  nearest neighbors  $D_i$  of word  $w_i$ .
7:     Retrieve the histogram  $h_i$ .
8:     for all Word  $w_j \in D_i$  do
9:       Increment the element  $w_j$  in  $h_i(j)$ .
10:    end for
11:  end for
12:  for all Word  $w_i \in W$  do
13:    Normalize s.t.  $\sum_{w_i} h_i = 1, \forall w_i \in W$ .
14:  end for
15: end for
```

3.2 Spatial expansion based on a general thesaurus

In this section, we present a **spatial expansion** method to recover query relevant visual words that are not present in the original query. The recovery of query relevant visual words is based on the observation that matching features are sometimes assigned to different words due to viewpoint changes and quantisation errors, and therefore do not contribute to the similarity measure. This can cause true positive results to be omitted from the top ranks.

There has been much research on improving the query recall, which has been discussed in Section 2.4. For example, soft assignment [107] expands the query by mapping features to multiple cluster centres (visual word IDs) during forming the visual vocabulary. The expansion leads to more accurate retrieval results, however, with the expensive cost of run time and memory usage. Query expansion [37] applies relevance feedback from initial retrieval results to refine the query.

Inspired by these methods, we propose a spatial expansion method to effectively collect query related visual words during run time. Our spatial expansion method is different from query expansion [37] in the following aspects: *i*) the expansion of the query is based on visual words that are spatially related to the query, instead of geometric consistency [37]; *ii*) it mines the spatial information offline, instead of determining spatially consistent words online. The spatial expansion differs from soft assignment in that it is based on spatial co-occurrence of features in images rather than neighbourhood relationships in

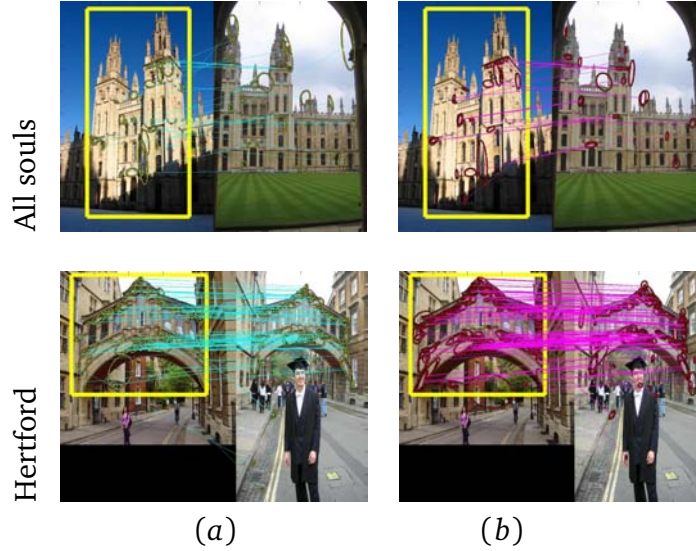


Figure 3.3: Examples of feature correspondences returned by spatial expansion. Examples of query landmarks: *All souls* and *Hertford*. (a): feature correspondences with query words \mathbf{Q} themselves. (b): extra feature correspondences with the spatial latent visual words (\mathbf{W}_T) expanded by \mathbf{F}_1 .

Algorithm 5 Spatial expansion based on a general thesaurus.

- 1: **Input:** Query words \mathbf{Q} , general thesaurus \mathbf{F}_n and threshold t .
 - 2: **Output:** Refined query words by spatial expansion: \mathbf{Q}' .
 - 3: **Initialize:** $\mathbf{W}_T = \emptyset$.
 - 4: **for all** Word q_i in the query \mathbf{Q} **do**
 - 5: Retrieve the histogram h_i .
 - 6: **for all** Entry k in histogram h_i **do**
 - 7: **if** $h_i(k) > t$ **then**
 - 8: Add word w_k to \mathbf{W}_T .
 - 9: **end if**
 - 10: **end for**
 - 11: **end for**
 - 12: Query with $\mathbf{Q}' \leftarrow [\mathbf{Q}, \mathbf{W}_T]$.
-

feature space. Thus although both methods are a form of query expansion, they operate in different ways. The visual thesaurus also has the advantage that it can work directly on the visual words rather than requiring that the features be clustered again in order to record multiple nearby cluster centres.

The outline of our spatial expansion method is described as follows:

1. A query image is represented as a set of visual words: $\mathbf{Q} = \{q_i\}_{i=1}^n$, in which $q_i \in \mathbf{W}$ and $\mathbf{Q} \subset \mathbf{W}$.

2. For each visual word $q_i \in \mathbf{Q}$, a thesaurus histogram h_i can be retrieved from general thesaurus.
3. Spatially expand the original query \mathbf{Q} with relevant words collected from the general thesaurus \mathbf{W}_T .
4. Query the dataset with the refined query words: $\mathbf{Q}' \leftarrow [\mathbf{Q}, \mathbf{W}_T]$.

Note that words in \mathbf{W}_T are chosen based on their co-occurrence with the visual words in the query, and correspond to peaks in the thesaurus histograms, as illustrated in Figure 3.2. During the spatial expansion, these co-occurrence relationships are useful for a query containing a large number of visual words (*Hertford* in Figure 3.3 (a)), and critical for a query where only a small number of visual words are present or the object’s appearance changes significantly (*All souls* in Figure 3.3 (a)). The image query is then formed by concatenating the query words \mathbf{Q} and latent words \mathbf{W}_T together:

$$\mathbf{Q}' = [\mathbf{Q}, \mathbf{W}_T] = [q_1, q_2, \dots, q_n, w_1, w_2, \dots, w_k] \quad (3.1)$$

As shown in Figure 3.3 (b), more visual word matches can be found by expanding the query words \mathbf{Q} .

Algorithm 5 describes the details of generating the latent visual words \mathbf{W}_T . The threshold t used to choose spatially related words is defined by an automatic mechanism, as described in Section 3.3. As a result, we obtain an enriched query \mathbf{Q}' to search the dataset. The image similarity is computed using the baseline measure (Eq (2.9)).

3.3 Experimental results

In this section, we investigate the effects of spatial expansion on three public datasets: Oxford 5K, Paris 6K and Oxford 105K. The effects examined as follows:

Effects of general thesaurus type Tables 3.1 and 3.2 investigate the retrieval performance on six types of general thesaurus: $\mathbf{F}_1, \mathbf{F}_2, \dots, \mathbf{F}_6$, which correspond to different numbers of nearest neighbors in a spatial region. The distance threshold

	Ground truth		mAP						
	Good + OK	Junk	Baseline	F_1	F_2	F_3	F_4	F_5	F_6
All Souls	78	111	0.544	0.603	0.653	0.665	0.673	0.693	0.711
Ashmolean	25	31	0.617	0.619	0.627	0.642	0.638	0.640	0.645
Balliol	12	18	0.563	0.626	0.630	0.601	0.636	0.644	0.644
Bodleian	24	30	0.456	0.578	0.594	0.601	0.622	0.616	0.621
Chris. Chur.	78	133	0.589	0.691	0.722	0.718	0.723	0.725	0.724
Cornmarket	9	13	0.563	0.583	0.571	0.573	0.584	0.591	0.579
Hertford	24	31	0.762	0.816	0.816	0.824	0.843	0.847	0.788
Keble	7	11	0.700	0.663	0.774	0.696	0.730	0.802	0.805
Magdalen	54	103	0.186	0.163	0.174	0.173	0.156	0.152	0.134
Pitt River	7	9	0.995	1.00	1.00	1.00	1.00	1.00	1.00
Radc. Camb.	221	348	0.609	0.747	0.775	0.764	0.814	0.826	0.833
Total	539	838	0.612	0.644	0.66	0.662	0.674	0.685	0.689

Table 3.1: Retrieval performance of spatial expansion on the Oxford 5K dataset. The top line characters represent the methods we use in the experiment: the baseline method, and six types of general thesaurus (F_1, F_2, \dots, F_6)

is chosen as $\rho = 10$ neighboring pixels in building general thesaurus, such that the spatial region focuses on the close neighbors of a feature in image domain. As the number of nearest neighbors increases, the general thesaurus is able to include more spatial information. As shown in Tables 3.1 and 3.2, the overall mAPs of F_n increases with the number of nearest neighbors n . However, for individual queries the peak mAP may be obtained for a value of F_n that is less than the maximum. This is because where the object is small or partly hidden in the image, its spatial support will be small. In this case, adding more spatial neighbours adds noisy expanded words which tend to decrease retrieval accuracy. The precision-recall curves of two particular cases *All souls* and *Hertford* are illustrated in Figure 3.5, showing the improvement of the retrieval performance after spatial expansion (corresponding to Table 3.1). The choice of threshold t is crucial in spatial expansion. It should ensure that as the number of nearest neighbors becomes larger, the threshold t also requires a less strong correlation in order to accept words. This is because, the visual words included in a general thesaurus becomes more diverse as the spatial neighborhood increases. Therefore, the threshold t should decrease as the nearest neighbor increase. Initially, the threshold t is set to 0.1, and then monotonically decreases by 30% as the number of nearest neighbors increases (*i.e.* 0.1 for F_1 , 0.07 for F_2 , 0.049 for F_3 , and so on).

Spatial expansion analysis We investigate the effects of spatial expansion by two aspects: *i*) the effects of spatially expanded words W_T ; *ii*) the effects of spatial

	Ground truth		mAP						
	Good + OK	Junk	Baseline	F_1	F_2	F_3	F_4	F_5	F_6
Defense	117	116	0.419	0.439	0.423	0.459	0.460	0.452	0.449
Eiffel	289	473	0.463	0.499	0.507	0.520	0.528	0.526	0.515
Invalides	198	99	0.643	0.679	0.698	0.692	0.697	0.701	0.707
Louvre	152	147	0.380	0.377	0.390	0.373	0.370	0.361	0.352
Moulinrouge	237	103	0.607	0.629	0.650	0.647	0.658	0.665	0.666
Museedorsay	72	15	0.546	0.584	0.587	0.612	0.610	0.619	0.610
Notredame	119	112	0.807	0.847	0.857	0.859	0.864	0.870	0.873
Pantheon	126	191	0.920	0.944	0.947	0.956	0.960	0.961	0.959
Pompidou	51	77	0.916	0.914	0.911	0.908	0.905	0.901	0.897
Sacrecoeur	149	152	0.826	0.864	0.861	0.874	0.874	0.873	0.865
Triomphe	281	164	0.507	0.533	0.536	0.544	0.542	0.543	0.544
Total	1791	1649	0.639	0.664	0.670	0.677	0.679	0.679	0.676

Table 3.2: Retrieval performance of spatial expansion on the Paris 6K dataset. The evaluation is the same as the Oxford 5K dataset.

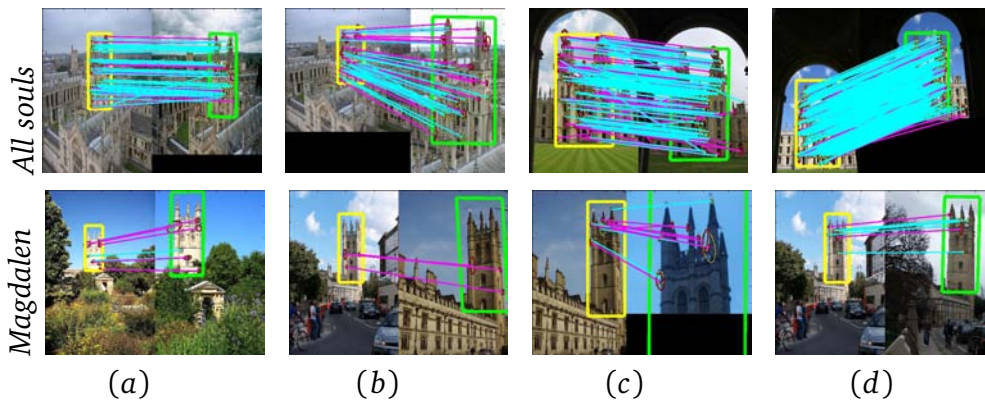


Figure 3.4: Examples of qualitatively examining of spatially expanded words on *All souls* and *Magdalen*. The inlier correspondences with query words \mathbf{Q} are indicated in blue, while the correspondences with their spatially expanded words \mathbf{W}_T are indicated in magenta.

information included in \mathbf{W}_T . Firstly, we evaluate the retrieval performance with the expanded words \mathbf{W}_T only, ignoring the query words \mathbf{Q} . On the Oxford 5K dataset, we obtain the mAP score as 0.656, compared to the baseline (0.612). Therefore, these spatial related words are effective in improving the retrieval performance. Secondly, we use both query words \mathbf{Q} and spatially expanded words \mathbf{W}_T . The retrieval accuracy (mAP) increases for most queries on the Oxford 5K and Paris 6K datasets. As reported in Tables 3.1 and 3.2, performance with spatial expansion on some landmarks becomes worse as n increases, because noisy spatial information is included *e.g.* *Magdalen* ($n \geq 3$). To investigate this, we examine the correspondences between spatially expanded visual words for selected image pairs. The correspondences are obtained by using both the query

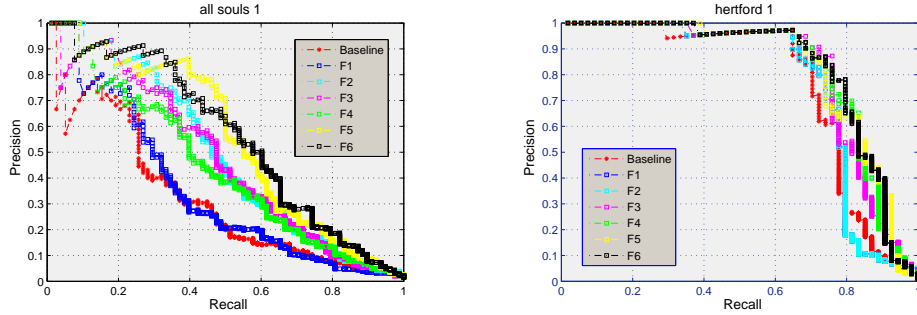


Figure 3.5: Precision-recall (PR) curves of *All souls* and *Hertford* with baseline method and six types of spatial expansion based on general thesaurus.

words Q (blue) and the expanded words W_T (magenta). As seen in Figure 3.4, the *All souls* images mostly feature the building prominently, and thus the query correspondences occur mostly in image regions that are locally consistent. In these cases, spatial expansion reinforces these correct matches with further consistent correspondences. However, in the case of *Magdalen*, the building is often small in the image or partly occluded. There are often not enough matches between query words to retrieve the results shown in Figure 3.4. In this case, the words obtained by spatial expansion account for most of the matches and are vital in boosting recall. However, some matches of them are incorrect because of the small region of spatial support around some query features. This affects the retrieval performance when n increases.

Figure 3.6 illustrates the effects of the general thesauruses F_1 - F_6 by comparing the retrieval accuracy and the number of words included in the query. The horizontal axis shows the number of spatially expanded words that are included increases continuously as the thesaurus includes more neighbours (from F_1 to F_6). However, the improvement in average retrieval accuracy (mAP scores) diminishes as n increases, as discussed above. To compromise between accuracy and efficiency, we set the general thesaurus as F_5 by default in the rest of this thesis.

Comparison to the state-of-the-art methods We compare our spatial expansion method to a number of state-of-the-art methods as listed in Table 3.3. The comparison is separated into two groups, as consistent with Table 2.8. Compared to Group A (offline methods), the retrieval accuracy of spatial expansion F_5

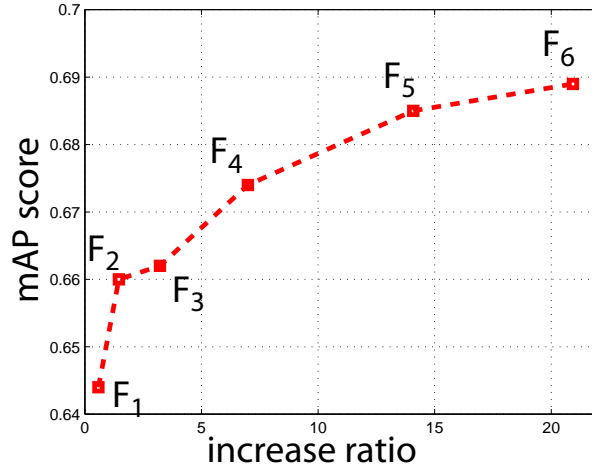


Figure 3.6: Illustration of the accuracy of spatially expanded words according to different general thesaurus F_1 - F_6 on the Oxford 5K dataset. This figure depicts the mAP score as a function of the increase ratio of spatially expanded visual words (expanded / original query words). For instance, by doubling the query visual words (F_2), the mAP score achieves 0.66.

is close to the other improvements, including descriptor learning [108], soft-assignment [107], and geometry-preserving [159] on three datasets. Our spatial expansion method does not outperform methods equipped with dataset side feature augmentation, *e.g.* AUG [142] and SPAUG [9], where spatial related features are examined and linked by a matching graph. However, our spatial expansion method is simpler in implementation than those methods pre-processing of the entire dataset.

Our spatial expansion method costs less memory than soft-assignment [107]. The soft-assignment method requires $n^2 = 9$ times more memory usage and run time than the baseline (where $n = 1$), because each feature is assigned to $n = 3$ visual words. Our spatial expansion method spends less computational cost than the soft-assignment method, as illustrated in Figure 3.6. It can also outperform the spatial verification [106] without a spatial consistency examination. However, it is lower than various query expansion methods in Group B, because query expansion methods use more accurate spatial related words obtained online.

In Group C, we show that our spatial expansion method can jointly work with a number of post-processes and receive further improvement. These include: spatial verification, AQE and DQE. As reported in Table 3.3, the improvement of spatial expansion combined with spatial verification depends on the initial

Methods		Oxford 5K	Paris 6K	Oxford 105K
Baseline [106]		0.612	0.639	0.515
A	Descriptor learning (non-linear) [108]	0.662 [108]	0.678 [108]	0.541 [108]
	Soft-assignment [107]	0.673 [107]	0.660	N/A
	Spatial expansion (F_5)	0.685	0.679	0.622
	Geometry-Preserving [159]	0.696 [159]	N/A	0.604 [159]
	Fine vocabulary [96]	0.742 [96]	0.749 [96]	N/A
	AUG [142]	0.776 [9]	N/A	0.711 [9]
	SPAUG [9]	0.785 [9]	N/A	0.723 [9]
B	Spatial verification [106]	0.649	0.655	0.571
	QE Baseline [37]	0.708	0.736	0.679
	iSP [34]	0.741 [34]	0.769 [34]	0.649 [34]
	Local geometry [105]	0.788 [105]	0.634 [105]	0.725 [105]
	AQE [37]	0.806	0.769	0.767
	DQE [9]	0.798	0.783	0.809
	Hello neighbors [114]	0.814 [114]	0.803 [114]	0.767 [114]
	Total recall II [34]	0.827 [34]	0.805 [34]	0.767 [34]
C	Spatial expansion (F_5)	0.685	0.679	0.622
	Spatial expansion + Spatial verification	0.716	0.689	0.676
	Spatial expansion + AQE	0.815	0.778	0.773
	Spatial expansion + DQE	0.810	0.785	0.798

Table 3.3: Comparison of spatial expansion F_5 to the state-of-the-art methods. Group A: retrieval results of methods that modify the baseline before the query is executed (pre-process). Group B: retrieval results of methods that modify the baseline after the query is executed (post-process). Group C: comparison and combination of spatial verification and various query expansion methods. Note that we cite the retrieval results of AUG [142] from literature [9].

retrieval results returned by the baseline or spatial expansion. However, two query expansion methods, AQE and DQE already include a spatial verification step, and thus combining with spatial expansion results in only a minor improvement.

3.4 Discussion

We show some top ranked retrieval results of spatial expansion F_5 in Figure 3.7. As seen in these examples, spatial expansion F_5 is able to return more true positives than the baseline method among the top-10 results compared to the results shown in Figure 2.7 (baseline). Our spatial expansion method achieves improvement of retrieval results by introducing more query relevant words collected offline. We obtain the mAPs as 0.685 and 0.679 for the Oxford 5K and Paris 6K datasets respectively, which are 11.9% and 6.26% increase over the baseline, respectively. The improvement is more evident on large scale dataset. We obtain the mAP

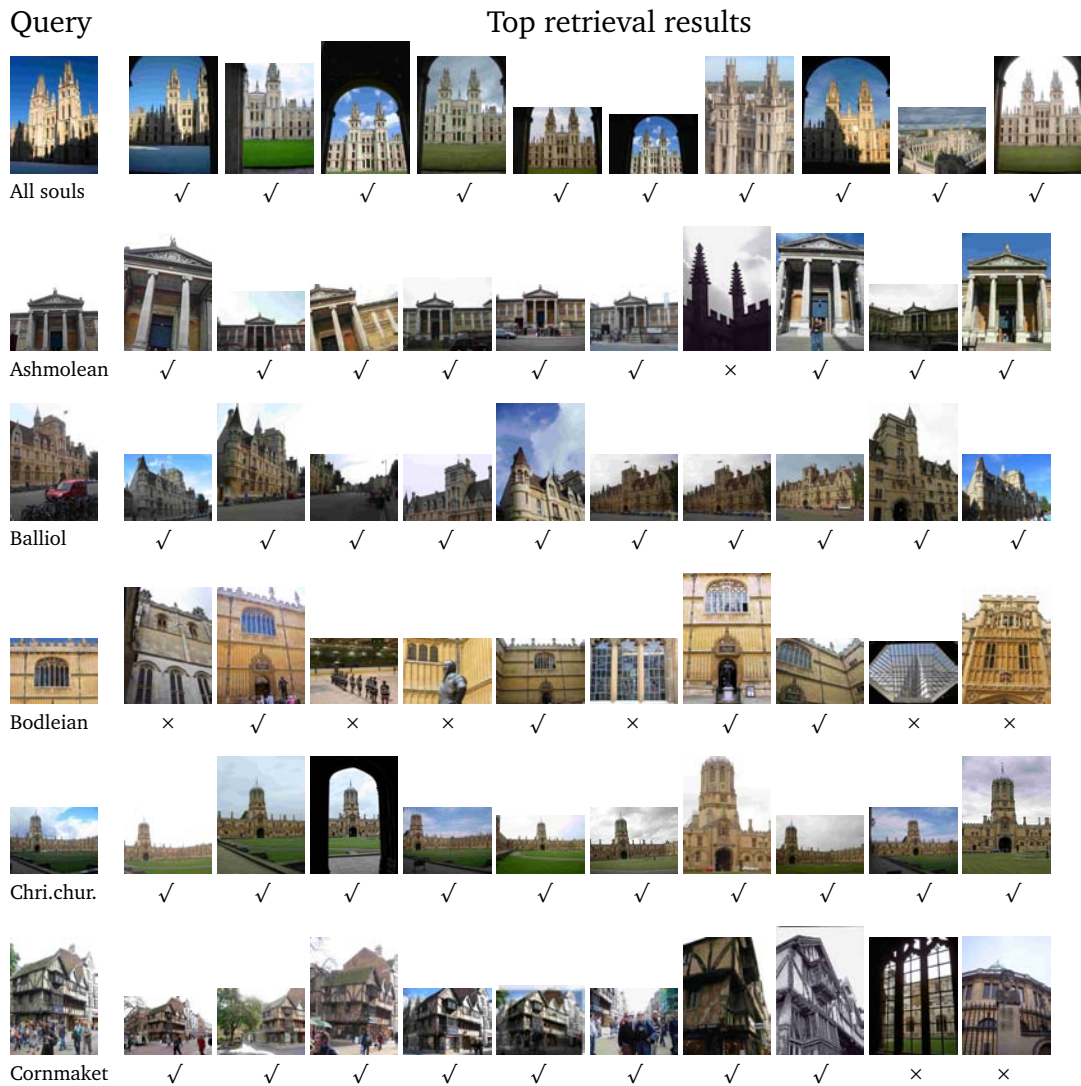


Figure 3.7: Top retrieval results of the spatial expansion method based on a general thesaurus F_5 .

as 0.622 on the Oxford 105K dataset, which is 20.8% higher than the baseline method, as shown in Table 3.3.

Note that the improvement of retrieval performance is mainly due to the enriched query with spatially expanded words. During spatial expansion, however, it is also likely to include noisy features, as discussed in the experimental results (Section 3.3). This is because the visual words collected by the general thesaurus are only constrained to a fixed spatial region, which is not able to ensure these words are correctly associated to the query words. Therefore, the query relevant

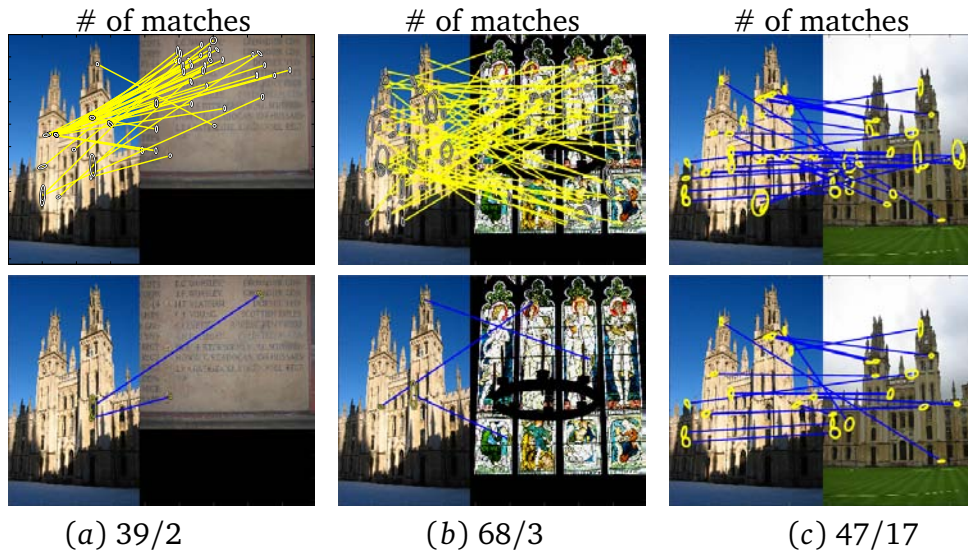


Figure 3.8: Comparison of matched visual words. Numbers beneath each example denote the total number of matches, and number of spatially consistent matches. (a) and (b): False positive image matches, ranked 3rd and 7th. (c): True positive matches, containing more consistent matches.

words found by spatial expansion might contain errors, such that retrieval results is not as accurate as AQE [37]. Recent work [34] has also noticed that there are confusing words for a given query image, which leads the BoW retrieval system fail to select relevant images by spatial verification. Therefore, query expansion benefits from eliminating irrelevant visual words. Inspired by these, we propose an object-based thesaurus to more accurately explore the spatial relationship among visual vocabulary.

3.5 Building an object-based thesaurus

Different from a general thesaurus, an object-based thesaurus, \mathbf{F}'_n , is able to eliminate background features that occur in the dataset images. The parameter n is the number of nearest neighbors to define \mathbf{D}_i , which is same as defined for \mathbf{F}_n . We exploit two properties of the foreground words.

Firstly, similar to [142], we also noticed that there are only a few visual words in the vocabulary essential to represent an object ¹. An object-based thesaurus, therefore, only needs to construct thesaurus histograms on these

¹As reported in [142], object features cover as few as 4% of total features in the dataset images.

foreground words, instead of visual words in the whole vocabulary as done by a general thesaurus.

Secondly, an object-based thesaurus captures the local geometric consistency of these foreground visual words, without the need for post-process. Figure 3.8 ((a) and (b)) shows cases where the spatially expanded word correspondences, based on a general thesaurus, are mostly incorrect in false positive results. In contrast, most correspondences are correct in a true positive result, and appear on the object we are trying to find (Figure 3.8 (c)). The standard method for eliminating these mistakes in matching is spatial verification [106], or one of its variants. These methods overcome this by basing the image ranking on the number of spatially consistent matches found between a pair of query/dataset images. Visual words that appear as *inliers* in the homography estimation [54] are considered correct matches, while images that exhibit a high degree of spatial consistency are true positives. The detected foreground words can be stored in an object-based thesaurus.

Therefore, the scheme of object-based thesaurus is composed of two parts: a training stage to collect foreground words and use of foreground words to construct thesaurus histograms. After that, we use the object-based thesaurus in two kinds of improvement: *i*) spatially expand the query words, as described in this chapter; *ii*) re-weight the importance of visual words, which will be described in Chapter 4.

3.5.1 Automatic training data collection

Collecting foreground words in our method is different from [142], because we do not need to build a matching graph. Instead, we borrow the automatic training data collection scheme from [108]. The data collection requires homography estimation between pairs of images randomly picked up from the dataset [108]. The process repeats until there is no growth of foreground words extracted from the dataset. This can be seen as a spatial verification process executed in an offline stage. The training data collection proceeds as follows:

1. An image pair is randomly chosen from the dataset.
2. Features are matched by their corresponding visual word IDs.

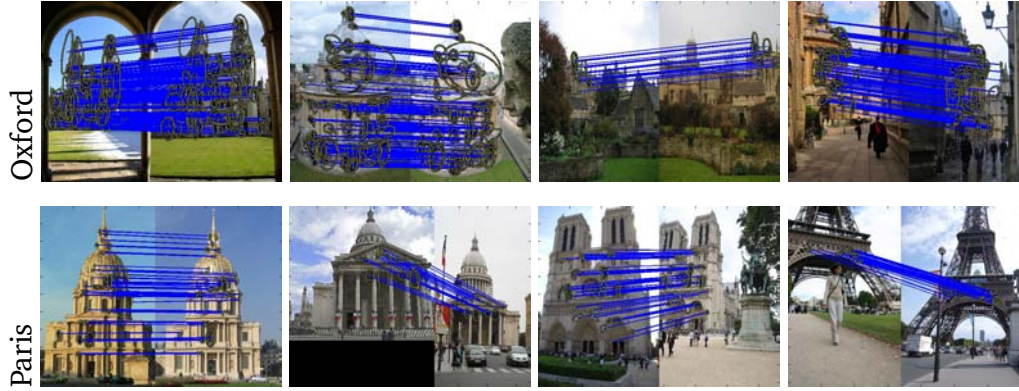


Figure 3.9: Training pairs detected with automatic selection on the Oxford 5K and Paris 6K datasets. All the pairwise images contain at least 20 liners during RANSAC. The inlier correspondences are shown in blue lines.

3. Image pairs matched by a homography transformation with at least 20 inliers found in RANSAC (as set by [108]) are retained.
4. Extract foreground visual words (detected as inliers) from images.

The difference between our method and [108] is in two aspects. Firstly, the homography matrix is calculated with correspondence of visual words, instead of raw SIFT features. That is because visual word matching is much faster, allowing us to process a larger amount of training data at the expense of slightly lower accuracy. In practice this trade-off is beneficial. Secondly, we only use the inliers from the training pairs, visual words that have not occurred as inliers are ignored. Figure 3.9 shows some training image pairs collected from the Oxford 5K and Paris 6K datasets with a high number of geometric inliers.

In practice, approximately 10% of the visual words in vocabulary \mathbf{W} form the foreground words \mathbf{W}_S . The detailed effects of training data amount will be discussed in next chapter.

3.5.2 Association of words in thesaurus

Let $\mathbf{W}_S := \{w_i\}_{i=1}^n$ be the foreground words appearing as inliers at least once in training data. The object-based thesaurus captures the spatial co-occurrence properties of only inlier words \mathbf{W}_S . The process is similar to Algorithm 4. It defines a n nearest neighborhood (\mathbf{D}_l), as used in a general thesaurus. However,

the visual words in the neighborhood need to be inliers appearing in the training data collection, which is not required in building a general thesaurus. The object-based thesaurus is also composed of histograms, which records the frequency of co-occurrence among the inlier words.

3.5.3 Spatial expansion based on an object-based thesaurus

In this section, we illustrate the effects of spatial expansion based on an object-based thesaurus F'_n . The process follows Algorithm 5, except for the way to select spatially expanded words to the given query. In an object-based thesaurus F'_n , the distribution of visual words only records those appearing as inliers (W_S). Therefore, the histograms count the frequency of co-occurrence in F'_n , instead of normalized frequency used in F_n . The experiments investigate the following issues of an object-based thesaurus F'_n used in spatial expansion: *i*) the choice of object-based thesaurus type F'_n ; *ii*) the threshold setting of word selection during spatial expansion.

Firstly, the type of object-based thesaurus F'_n defines the spatial region of the co-occurrence of visual words. Unlike a general thesaurus, the foreground words have been filtered, and so we do not rely so heavily on image location. Thus we can enlarge the neighborhood n in an object-based thesaurus so as to include enough neighboring foreground words in the thesaurus histograms. Therefore, we define the type of object-based thesaurus as F'_{15} in the following experiments.

Secondly, Table 3.4 evaluates the retrieval accuracy with the decreasing threshold number of co-occurrences t in spatial expansion based on an object-based thesaurus. Similar to the results obtained from a general thesaurus, the retrieval accuracy reaches a plateau when t is close to 1, in which almost all the foreground words are included. Figure 3.10 illustrates the number of spatially expanded words when threshold t decreases. The decrease in threshold leads to the increase in amount of spatially expanded words, as well as increase in retrieval accuracy. However, the number of spatially expanded words grows dramatically when t becomes lower than 20. In order to balance the efficiency and effectiveness, we set $t = 20$ by default, as it achieves high retrieval accuracy while using less spatially expanded words.

Moreover, Table 3.5 compares the retrieval results to the state-of-the-art methods on three datasets: Oxford 5K, Paris 6K and Oxford 105K. As seen in

Methods		Oxford 5K	Paris 6K
Baseline		0.612	0.639
Spatial expansion (\mathbf{F}_5)		0.685	0.679
$t = 30$	Spatial expansion (\mathbf{F}'_{15})	0.690	0.673
$t = 25$	Spatial expansion (\mathbf{F}'_{15})	0.690	0.677
$t = 20$	Spatial expansion (\mathbf{F}'_{15})	0.701	0.683
$t = 15$	Spatial expansion (\mathbf{F}'_{15})	0.704	0.691
$t = 10$	Spatial expansion (\mathbf{F}'_{15})	0.708	0.693
$t = 5$	Spatial expansion (\mathbf{F}'_{15})	0.716	0.708
$t = 3$	Spatial expansion (\mathbf{F}'_{15})	0.715	0.715

Table 3.4: Retrieval results of spatial expansion based on object-based thesaurus. The threshold t is set as times of co-occurrence recorded in an object-based thesaurus.

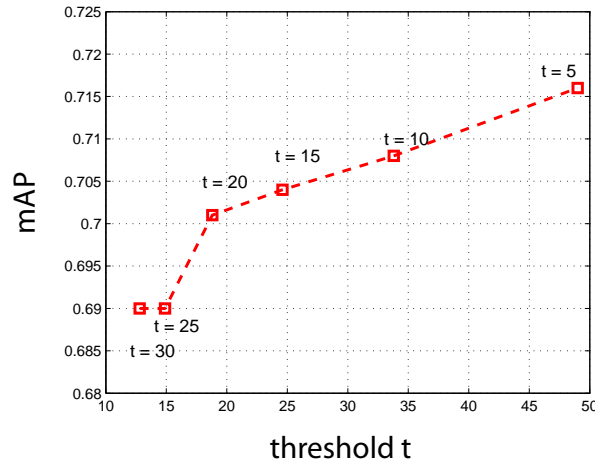


Figure 3.10: Illustration of retrieval accuracy of spatially expanded words (\mathbf{F}') included by various threshold.

Table 3.5, spatial expansion \mathbf{F}'_{15} is able to increase the retrieval accuracy on all three datasets. It increases the retrieval accuracy by 14.5%, 6.9% and 29.5% on the three datasets, compared to the baseline. Figure 3.12 shows some retrieval results of spatial expansion \mathbf{F}'_{15} . As seen from Figure 3.12, the retrieval results have been improved significantly on some queries, *e.g.* Bodleian, in which the top 10 results are all correct. However, using inlier words alone also introduces some false positives, *e.g.* Balliol. The detailed comparison between two kinds of spatial expansion (\mathbf{F}_5 and \mathbf{F}'_{15}) is illustrated in Figure 3.11. As seen in Figure 3.11 (a), the retrieval performance of most queries are improved by spatial expansion (\mathbf{F}_5). However, the retrieval performance of spatial expansion (\mathbf{F}'_{15}) varies dramatically,

Methods		Oxford 5K	Paris 6K	Oxford 105K
Baseline [106]		0.612	0.639	0.515
A	Descriptor learning (non-linear) [108]	0.662 [108]	0.678 [108]	0.541 [108]
	Soft-assignment [107]	0.673 [107]	0.660	N/A
	Spatial expansion (F_5)	0.685	0.679	0.622
	Geometry-Preserving [159]	0.696 [159]	N/A	0.604 [159]
	Spatial expansion (F'_{15})	0.701	0.683	0.667
	Fine vocabulary [96]	0.742 [96]	0.749 [96]	N/A
	AUG [142]	0.776 [9]	N/A	0.711 [9]
	SPAUG [9]	0.785 [9]	N/A	0.723 [9]
B	Spatial verification [106]	0.649	0.655	0.571
	QE Baseline [37]	0.708	0.736	0.679
	iSP [34]	0.741 [34]	0.769 [34]	0.649 [34]
	Local geometry [105]	0.788 [105]	0.634 [105]	0.725 [105]
	AQE [37]	0.806	0.769	0.767
	DQE [9]	0.798	0.783	0.809
	Hello neighbors [114]	0.814 [114]	0.803 [114]	0.767 [114]
	Total recall II [34]	0.827 [34]	0.805 [34]	0.767 [34]
C	Spatial expansion (F_5)	0.685	0.679	0.622
	Spatial expansion+ Spatial verification	0.716	0.689	0.676
	Spatial expansion+ AQE	0.815	0.777	0.773
	Spatial expansion+ DQE	0.810	0.782	0.798
	Spatial expansion (F'_{15})	0.701	0.683	0.667
	Spatial expansion + Spatial verification	0.719	0.689	0.704
	Spatial expansion + AQE	0.806	0.785	0.783
	Spatial expansion + DQE	0.813	0.789	0.818

Table 3.5: Comparison of spatial expansion to the state-of-the-art methods. Group A: retrieval results of methods that modify the baseline before the query is executed (pre-process). Group B: retrieval results of methods that modify the baseline after the query is executed (post-process). Group C: comparison and combination of spatial verification and various query expansion methods. Note that we cite the retrieval results of AUG [142] from literature [9].

i.e. some queries achieve very high accuracy (close to 1) while others drop significantly. This is because the co-occurrences of inliers are likely to be associated with multiple objects, as suggested by [110]. In Chapter 4, we investigate the spatial structure of these foreground words, such that the weights of them are adjusted according to their importance to the query object.

3.6 Conclusion

In this chapter, we have introduced a **visual thesaurus** data structure to capture informative latent visual words in an image database. We use a visual thesaurus structure to record the spatial co-occurrence of pairwise features in a fixed region.

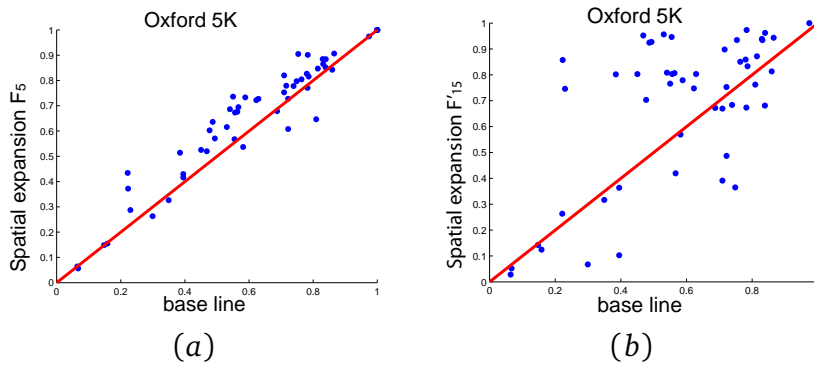


Figure 3.11: Comparison of various spatial expansion to the baseline. Each point represents an individual query of the 55 queries on the Oxford 5K dataset. (a) comparison of spatial expansion F_5 to the baseline. (b) comparison of spatial expansion F_{15} to the baseline. The points above (below) the diagonal illustrate the increase (decrease) of retrieval accuracy.

A **spatial expansion** method, incorporating with either a general visual thesaurus or an object-based thesaurus, is introduced to collect query relevant words which have high spatial co-occurrence with the query but are not present. After the spatial expansion of the query, retrieval is conducted with the standard method, *i.e.* tf-idf weights and dot product similarity. The spatial expansion aims to improve the BoW representation of query image, and thus the overall retrieval performance is improved.

The effects of these two kinds of visual thesaurus are different. A general thesaurus covers any visual word in the vocabulary, which is simple, *i.e.* it does not need training stage. In contrast, an object-based thesaurus is only constructed on foreground visual words and thus background information is avoided. However, it needs a training stage to collection these foreground information. The way of training data collection limits the effects of object-based thesaurus to spatially rigid objects. Therefore, the choice of visual thesaurus type depends on the purpose of different problems.

Because we focus on object retrieval, we use an object-based thesaurus in the following chapters to identify the importance of a visual word and the relatedness between different visual words.

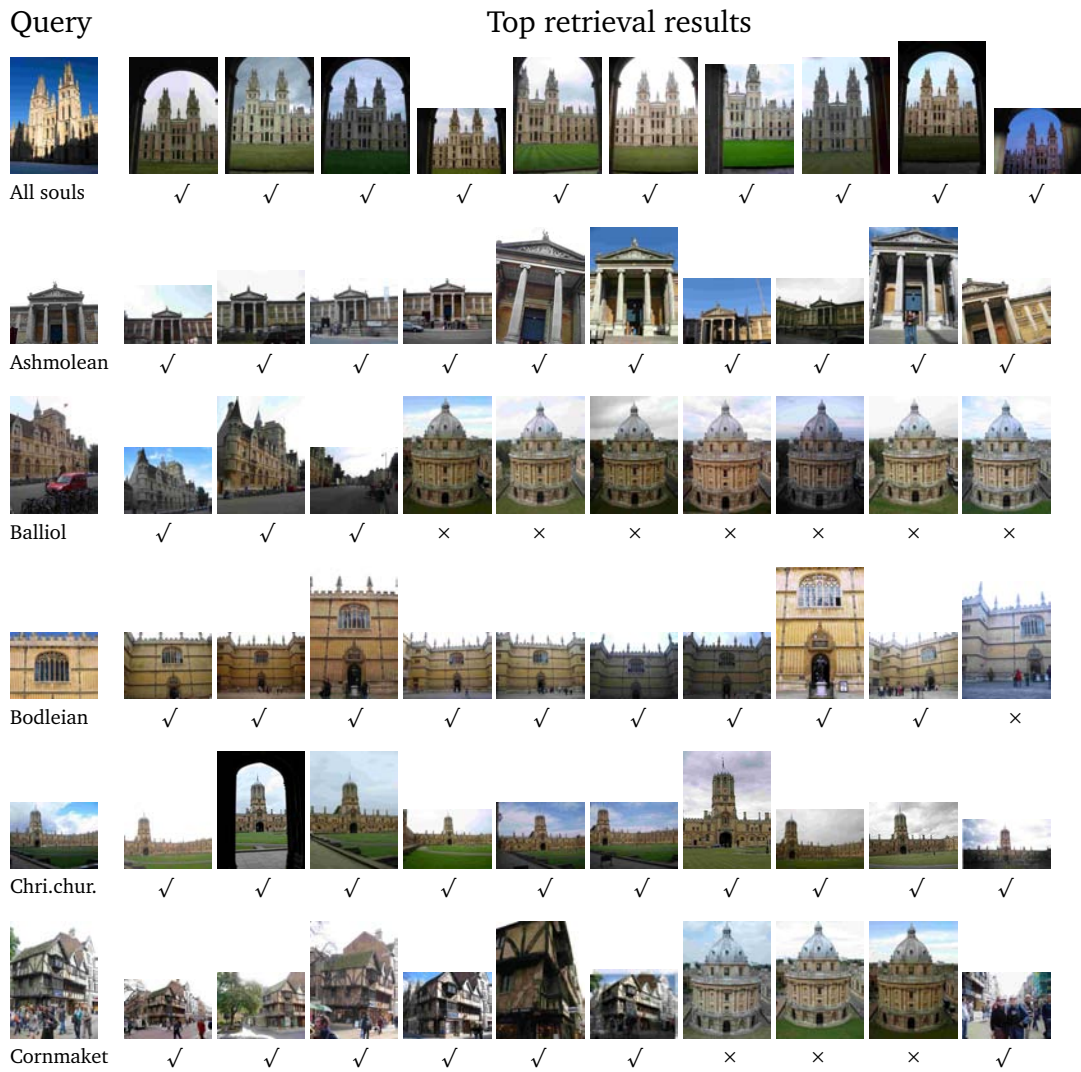


Figure 3.12: Top retrieval results of the spatial expansion method based on an object-based thesaurus F'_{15} .