Dissertation

Aggregating Local Features into Bundles for High-Precision Object Retrieval

Stefan Romberg

Universität Augsburg
Fakultät für Angewandte Informatik

Department of Computer Science

University of Augsburg

Advisor:	Prof. Dr. Rainer Lienhart
Reviewers:	Prof. Dr. Rainer Lienhart
	Prof. Dr. Elisabeth André
	Prof. Dr. Eckehard Steinbach
Thesis Defense:	December 13, 2013

Abstract

Due to the omnipresence of digital cameras and mobile phones the number of images stored in image databases has grown tremendously in the last years. It becomes apparent that new data management and retrieval techniques are needed to deal with increasingly large image databases. This thesis presents new techniques for content-based image retrieval where the image content itself is used to retrieve images by visual similarity from databases. We focus on the query-by-example scenario, assuming the image itself is provided as query to the retrieval engine.

In many image databases, images are often associated with metadata, which may be exploited to improve the retrieval performance. In this work, we present a technique that fuses cues from the visual domain and textual annotations into a single compact representation. This combined multimodal representation performs significantly better compared to the underlying unimodal representations, which we demonstrate on two large-scale image databases consisting of up to 10 million images.

The main focus of this work is on feature bundling for object retrieval and logo recognition. We present two novel feature bundling techniques that aggregate multiple local features into a single visual description. In contrast to many other works, both approaches encode geometric information about the spatial layout of local features into the corresponding visual description itself. Therefore, these descriptions are highly distinctive and suitable for high-precision object retrieval.

We demonstrate the use of both bundling techniques for logo recognition. Here, the recognition is performed by the retrieval of visually similar images from a database of reference images, making the recognition systems easily scalable to a large number of classes. The results show that our retrieval-based methods can successfully identify small objects such as logos with an extremely low false positive rate. In particular, our feature bundling techniques are beneficial because false positives are effectively avoided upfront due to the highly distinctive descriptions.

We further demonstrate and thoroughly evaluate the use of our bundling technique based on min-Hashing for image and object retrieval. Compared to approaches based on conventional bag-of-words retrieval, it has much higher efficiency: the retrieved result lists are shorter and cleaner while recall is on equal level. The results suggest that this bundling scheme may act as pre-filtering step in a wide range of scenarios and underline the high effectiveness of this approach.

Finally, we present a new variant for extremely fast re-ranking of retrieval results, which ranks the retrieved images according to the spatial consistency of their local features to those of the query image. The demonstrated method is robust to outliers, performs better than existing methods and allows to process several hundreds to thousands of images per second on a single thread.

Danksagung

Ich möchte mich bei all denjenigen bedanken, die mir diese Arbeit ermöglicht haben. Ich bedanke mich insbesondere bei Prof. Dr. Rainer Lienhart für seine Betreuung, Begutachtung, Ermutigung und seine Geduld während dieser Arbeit. Ebenso möchte ich mich bei Prof. Dr. Eckehard Steinbach und Prof. Dr. Elisabeth André für die Übernahme der Gutachten bedanken.

Ich danke all meinen Kollegen, insbesondere Eva Hörster, die mich in das weite Feld der Bildsuche eingeführt hat und ihre Erfahrung stets an mich weitergab. Ebenso bin ich Fabian Richter, Thomas Greif, Johannes Schels, Christian X. Ries, Christian Eggert, Christoph Lassner, Dan Zecha, Moritz August, Georg Schroth, Lluis Garcia Pueyo, Michele Trevisiol, Yannis Kalantidis und Roelof van Zwol dankbar für die Zusammenarbeit und die vielen Diskussionen und Anregungen.

Des Weiteren danke ich Christian Steinherr, der durch unkomplizierte technische Hilfe einige rechenintensive Experimente erst ermöglicht hat.

Ich möchte mich ferner bei Fabian Richter und Karin Lutz für ihr akribisches Korrekturlesen bedanken, welches sehr zum Gelingen dieser Arbeit beigetragen hat.

Sehr großer Dank gilt auch meinen Freunden und meiner Familie, insbesondere meinen Eltern. Sie alle haben mich stets unterstützt und mir Studium und Promotion erst ermöglicht.

Nicht zuletzt möchte ich mich ganz innig bei meiner Partnerin Karin Lutz bedanken, die mich durch diese Zeit begleitet hat, mir in jeglicher Hinsicht eine Stütze war und mir auch einen Blick über den Tellerrand vermittelt hat. Ohne sie wäre diese Arbeit nicht möglich gewesen.

To my father,
who always supported me
and who showed me that things in life have many facets.

Contents

1 Introduction

1.1 Motivation

The objective of image retrieval is to find images in a database that are relevant to a given query. In the early days image retrieval systems were almost entirely text-based: The query was supplied by keywords or free text. The relevance of an image with respect to the query was then determined by matching text fragments such as image captions or manually made annotations with the query text. However, such a technique requires the presence of metadata or manual labels and thus strongly limits its scope.

In contrast, this thesis focuses on *content-based image retrieval*, which allows to search for visually similar images without requiring annotations or free text as query. Here, the actual image content is processed to determine whether two images are visually similar. By extracting information from edges, corners and blobs as well as color and gradient distributions the image is eventually described by a higher-level representation than by plain pixels. We follow the query-by-example paradigm where the query itself is supplied as an image without additional text keywords. The retrieval engine then processes the input image, extracts a suitable representation and searches an image database for images that have a similar representation.

It is important to understand that image retrieval does not require image understanding or "image recognition". This is due to the fact that a plain retrieval engine normally lacks a semantic anchor. Just like a child that plays with a toy car but never got told what toy it is playing with, the engine can neither name nor recognize the content of an image. However, it is able compute a similarity measure for two images showing the same yet unknown object.

The retrieval system just maps similar visual content to similar descriptions but has no conception of an underlying high-level concept such as specific objects, scenes or landmarks. The underlying principle is simply that two images that show the same object, scene or concept should have a visual description that is in some notion similar. In contrast, if images are dissimilar, the visual description should reflect this and be dissimilar as well.

The main objective of research on image retrieval is to find a well-working mapping that follows this principle and further allows efficient and effective search for similar descriptions. This alone poses great demands on the robustness of the visual description – ruling out naive visual description such as from plain pixels immediately – as image retrieval should work in a wide range of imaging settings.

While the inability of a retrieval system to recognize image content for a high-level understanding seems unfortunate, it also allows to deal with new concepts without re-training – in contrast to an object recognition or image classification system.

In the following Section 1.2 we briefly sketch several applications for content-based image retrieval. In Section 1.3 we present common challenges for image retrieval system. Section 1.4 outlines the contributions of this thesis, while Section 1.5 explains the structure of this thesis.

1.2 Applications

Humans have always used images to express themselves and to transport information. With the appearance of digital cameras and the rise of the Internet the amount of images stored world-wide has increased tremendously. Thus, the demands for automatic vision-based management and retrieval techniques rise quickly. Here we briefly sketch a few current and future applications, focusing on those where image retrieval is the core technique.

Location Recognition Location Recognition is the localization of the user's current position within a city or a building by visual inspection of his surroundings. This may be done by taking pictures of the surroundings and searching a database of reference images such as Google's Streetview or Microsoft's Streetside. A location recognition system must be able to handle thousands to millions of locations and constantly adapt to changing environments. For that reason pre-trained classifiers are often infeasible and retrieval techniques need to be used. The user may use a mobile device to query such a system by views of his current surrounding. The system may answer the search by providing additional information about his current location such as information about the city district or the building's floor-plan.

Product Search Another scenario already partly realized on nowadays mobile phones is product search. Here, the objective is to quickly fetch information of a product while shopping. By taking a photograph of the product or its packaging users may retrieve information about the product on their mobile phone almost immediately. In contrast to traditional solutions such as scanning bar codes or QR codes, content-based retrieval does not require a special label on the product itself. Consequently, this technique also allows to use posters and printed ads as visual link. Once captured by the mobile application, the retrieval engine points to a web page describing the event or product.

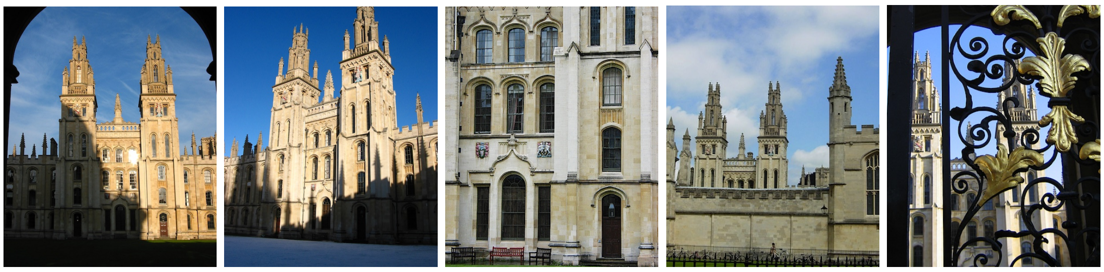
Visual Hyperlinks Quite similar, yet slightly different is the concept of visual hyperlinks. Following the PageRank principle (Page et al. 1999) web pages are linked by hyperlinks and deemed similar if they contain similar keywords and text. This scheme may be translated to the visual domain where pages are linked by the visual content of images. This may be extended to image sub-regions: For instance, an image often shows more than one concept or object. The corresponding image regions describe different concepts and thus may be linked to different images. The resulting image graph will describe the visual relations - just like the graph for web pages. It may be used to enrich the representations of a web page in order to improve search results in a web search engine, which can be queried either by text or image examples.

Visual Dictionary One long-term goal and a very useful application of image search would be a visual dictionary. User may take photographs of scenes, locations, landmarks, products and objects to retrieve information about those based on visual analysis. For example, a user may want to know the name of a flower that is unknown to him. With the help of a mobile device he may query a huge database of images cross-referenced and annotated either by humans or automatically with a photograph of that flower. The retrieval engine searches for the most similar image in the database, processes the corresponding metadata and returns the search result such as flower name and species to the user.

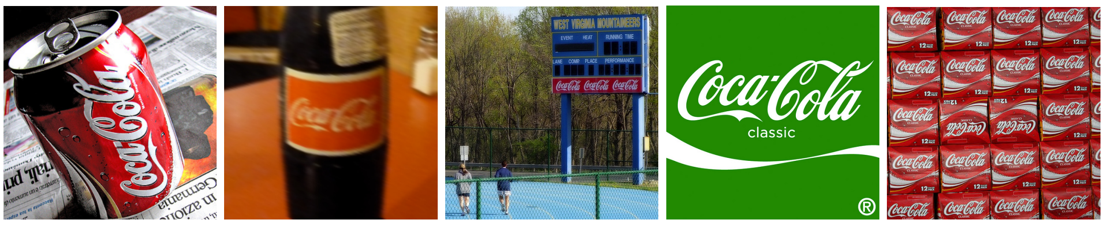
In general, such application provides a convenient and intuitive way to lookup information for arbitrary scenes and objects within a large visual dictionary. As additional information source, time, the user's location or his profile may be further used to augment the query.

1.3 Challenges

Early research mostly dealt with artificial and relatively clean data while nowadays the focus is on real-world images. Consequently, retrieval systems have to deal with a variety of challenges.



(a) Lighting (b) Perspective (c) Scale changes (d) Truncation (e) Occlusion



(f) Rotation (g) Noise & Blur (h) Small Objects (i) Variations (j) Multiple Objects

Figure 1.1: Examples of common challenges for image retrieval systems. The top row shows various views of the All Souls College. The lower row shows various views of the Coca-Cola logo. Images taken from the datasets Oxford (see Section 3.1.2) and FlickrLogos-32 (see Section 3.1.4).

The common challenges for image retrieval and also for computer vision in general include different varying lighting and illumination conditions (see Figure 1.1(a)). This alone basically rules out many color- or intensity-based representations as these image properties vary greatly under different illumination conditions. Photographs from different viewpoints (Figure 1.1(b)) pose a great challenge to the visual description as the underlying image content changes accordingly and rigid geometric descriptions are unable to handle this. Scale changes e.g., caused by camera zoom (Figure 1.1(c)) require robust scale-invariant descriptions. Truncation (Figure 1.1(d)) and Occlusion (Figure 1.1(e)) both partially hide the underlying concept or object. Thus naive global descriptions will break and more sophisticated representations are required.

A robust search for objects with different poses e.g with different orientations (see Figure 1.1(f)) requires a rotation-invariant visual description. Image noise and blur (Figure 1.1(g)) require a robust yet distinctive visual description that deals with those artifacts from difficult conditions. In contrast, small objects (Figure 1.1(h)) or objects with little structure (e.g., the apple logo) pose the challenge to be able to describe them at all. All visual representations must handle reasonable intra-class variations (Figure 1.1(i)) and be robust against minor changes. Eventually, multiple objects (Figure 1.1(j)) are problematic if methods can cope with a single instance only.

1.4 Contributions

The main contributions of this thesis can be summarized as follows:

- **Modality Fusion:** We describe a technique for fusing different modalities such as visual features and text or different visual features into a single description derived from a topic model. The underlying model is learned in a fully unsupervised manner. Experiments show that this method copes well with weak human annotations extracted from community databases. In this field, our contribution is two-fold: (1) We present a multi-level model consisting of independent topic models and describe an efficient heuristic for fast initialization. (2) We further describe a method that globally optimizes the model jointly over all levels and modalities.
- **Geometric Re-Ranking:** We present an extremely fast and effective geometric re-ranking method termed 1P-WGC-RANSAC. Our method is based on establishing 1-point correspondences resulting in a deterministic procedure that is also robust against false correspondences. We especially demonstrate how the re-ranking can be accelerated by omitting the projective re-estimation and by introducing a weak-geometric constraint. Overall, this approach significantly improves the retrieval results and outperforms related approaches in the literature. Due to its speed it is further suited for real-time applications.
- **Feature Bundling for Object Retrieval and Logo Recognition:** We developed two different feature bundling techniques especially suited for retrieval and detection of small objects. We explicitly designed our methods such that the spatial layout of local features is encoded *into* the signature. This is the *key distinction* to most existing approaches and effectively discards a great proportion of false positives.

  – We present a method to encode the spatial layout of feature triples into a compact signature. This signature is then used to access a hash table when indexing or querying. Using this highly distinctive signature we demonstrate an effective logo detection system by means of counting occurrences of distinctive feature configurations.
  – We further developed a second bundling technique to aggregate the spatial neighbors of local features into feature bundles. By employing a locality-sensitive hashing via min-Hash, our method is capable of robust similarity search for similar yet not identical sets of local features.
  – We extensively evaluate these methods on publicly available datasets of real-world images showing small objects – i.e., logos of different brands. We further demonstrate the general applicability of the latter bundling technique for object retrieval and its improvement over existing methods.
  – Finally, we demonstrate a system that exploits our highly distinctive bundle representation to perform logo recognition with state-of-the-art performance.

Parts of this thesis have been published in peer-reviewed conferences and journals: The multi-modal fusion of visual features and tags has been described in Romberg et al. (2009), Lienhart et al. (2009) and Romberg et al. (2012). The automatic discovery of spatially consistent feature pairs and triples and the encoding of their spatial layout into discrete signatures have been presented in Romberg et al. (2011). The re-ranking technique based on a modified RANSAC variant was presented in Romberg and Lienhart (2013b). Our approach for feature bundling based on min-Hashing has been exploited for retrieval in Romberg et al. (2012) and for logo detection in Romberg and Lienhart (2013a;b). See Appendix A for a full list of publications.

1.5 Thesis Overview

This thesis consists of four parts:

Part I introduces the technical and theoretical foundations. In Chapter 2 we introduce visual features, clustering techniques, visual vocabularies and the underlying vector space model we build on. The different datasets and evaluation criteria are described in Chapter 3.

Part II presents an approach for multi-modal image retrieval exploiting both visual and textual cues in a fully unsupervised manner. In Chapter 4 a method is described that fuses information from different domains such as visual features and text, as well as different kinds of visual features into a single representation. We describe a method that learns a global model by jointly optimizing the representation over all levels and modalities, as well as a heuristic for fast initialization of the model – speeding up both training and inference in practice.

Part III focuses on feature bundling methods that aggregate multiple local features into a single description. In particular we target object retrieval, logo retrieval and high-precision image search for small objects in general. In Chapter 5 a high precision Monte-Carlo method for detecting logos in images is presented. A novel RANSAC variant for extremely fast re-ranking of image search results based on their geometric consistency with respect to the query is described in Chapter 6. In Chapter 7 a feature bundling technique based on min-Hashing is presented, suited for robust approximate similarity search for feature bundles. This technique is then exploited for object retrieval as well as logo recognition.

Finally, Part IV concludes this thesis with a brief discussion and outlook.

Part I

Foundations

2
Fundamental Techniques

In this chapter we discuss fundamental concepts and techniques of content-based image retrieval. In Section 2.1 we describe the different visual features used in this thesis. In Section 2.2 we explain different clustering techniques for grouping similar visual features into discrete classes that are used as visual representatives – the visual words. In Section 2.3 we then show how these visual words are used within the bag-of-words framework for a robust and scalable image description.

2.1 Visual Features

One of the most critical components of any computer vision application is the visual description of image content itself. Over the years a vast number of different visual features have been explored. With increasing computational power those features got more sophisticated, robust and were adopted and tuned to application-specific domains. The choice of the actual visual feature has to be made based on its properties, such as its intrinsic design, e.g., color- vs. non-color capturing features, its robustness, memory consumption and speed.

For a long time it was common that the whole image was described by a single *global feature*. Global features encode the visual content into a single description. Thus, these usually have very low memory requirements and are fast to compute. Global features are still being used e.g., for content-based video analysis or video retrieval - mostly due to their small memory footprint. However, in many domains *local features* superseded global features. Local features have been successfully applied to image retrieval, object recognition, image classification and are an important building block for many other computer vision applications. In this thesis, we focus on local features only as their performance for accurate object retrieval has been proven more effective than global features. A comparison of global and local features of various kinds in the context of image retrieval is given in (Deselaers et al. 2008).

By definition a local feature describes an image region ("patch") locally around a chosen point of interest. Such interest points are usually determined automatically by an interest point detector, sampled from a regular grid; or in rare cases one can select them manually. The description of the image patch is carefully designed to be invariant against certain transformations and robust to changes that commonly occur to images. For instance, the description should be unimpaired by image noise, changes in lighting and changes of the perspective viewpoint. At the same time the visual description should be also descriptive and discriminative such that unrelated image patches can be distinguished from each other and a meaningful similarity can be computed from the distance between descriptors.

Each image contains multiple local features; therefore any local feature-based approach implicitly deals with occlusion or partial appearances of an object. A change in lighting conditions or the perspective viewpoint may significantly change the pixel intensities of an image patch around a certain keypoint. An accurate visual description therefore largely depends on the robust detection of interest points and the invariance of the encoding by the descriptor.

Starting with simple naive descriptions of image patches soon more sophisticated visual descriptors were proposed. In combination with the progress on detection of interest points the decade of 2000-2010 became the decade of local features. Lowe (1999) proposed *SIFT* features including a scale- and rotation-invariant detection of interest point as well as a robust description of the local neighborhoods around these points and has become the de-facto standard. Today SIFT and some of its variants are the most widely used visual features in the domain of image retrieval despite the publication of a vast number of other features.

Much research has been dedicated to both the actual visual descriptors and the interest point detectors and yielded detectors like *MSER* (Matas et al. 2004), *Hessian-Laplace*, *Harris-Laplace*, *Hessian-affine*, *Harris-affine* (Mikolajczyk et al. 2005). Research has also focused on improving speed (Bay et al. 2008) and robustness against perspective changes (Mikolajczyk et al. 2005). Recently several detectors based on machine-learned extremely fast corner detection schemes have been proposed such as *FAST* (Rosten and Drummond 2006), *AGAST* (Mair et al. 2010), *BRISK* (Leutenegger et al. 2011) and *ORB* (Rublee et al. 2011).

Among the visual descriptors the original SIFT descriptor (Lowe 2004) remains one of the most used and best-performing descriptors. Countless SIFT variants have been proposed, such as Rank-SIFT (Li et al. 2011), SIFT-Rank (Toews and Wells 2009), π-SIFT (Park et al. 2008), VF-SIFT (Alhwarin and Ristić-Durrant 2010), Eff^2 (Lejsek et al. 2006) or approximated SIFT (Grabner et al. 2006). Each of them tackles a specific weakness of the original SIFT scheme. Other less closely related variants such as *GLOH* (Mikolajczyk et al. 2005), *PCA-*

SIFT (Sukthankar 2004) and *DAISY* (Grabner et al. 2006) rely on improved spatial pooling of image gradients. However, none of these nor any of the former variants replaced the de-facto standard SIFT. *SURF* (Bay et al. 2008); conceptually similar to SIFT - while being much faster - received much attention especially because it was easily portable to GPU hardware. While the aforementioned descriptors above are more or less gradient-based descriptions other descriptors capture shape (Belongie et al. 2002) or local self-similarities (Shechtman and Irani 2007) instead. Commonly local features are extracted from gray-scale images but color descriptors also have been explored (van de Sande et al. 2010). Recently several descriptors have been proposed that exploit a compact binary representation for efficient storage and fast distance computations such as *BRIEF* (Calonder et al. 2010), *ORB* (Rublee et al. 2011), *BRISK* (Leutenegger et al. 2011) and *FREAK* (Alahi et al. 2012).

In contrast to local features that were traditionally computed from sparse interest points, other features were introduced for the purpose of object recognition and later used for retrieval as well. This mainly includes SIFT descriptors computed from a regular grid (often termed *dense SIFT*) or from densely distributed pseudo-interest points (Tuytelaars 2010). In addition, the *Histogram of Oriented Gradients (HOG)* descriptor was specifically designed for object class description and received much attention.

A complete discussion of different local features and their techniques is beyond this thesis; a comprehensive overview is given in Tuytelaars and Mikolajczyk (2008). In the following we briefly describe the feature used in this thesis.

2.1.1 SIFT

Interest Point Detection

The *Scale Invariant Feature Transform (SIFT)* (Lowe 2004) marks an important milestone in computer vision. After a decade of competing local visual features SIFT features still remain among the best-performing and widest-used visual features in the computer vision community. It still is the de-facto standard for image retrieval. In order to make our results easily comparable to other results in literature most of the work in this thesis uses SIFT features. We use a mature implementation that incorporates many minor additions that have been added since Lowe's early publication (Lowe 1999) and were later summarized in Lowe (2004).

Local features describe small image patches locally around certain points in an image. Preferably these points should be in salient image regions e.g., showing structures or having high contrast. The detection and localization of those points is performed by an *interest point detector* - also coined *salient point detector* or more vaguely *region detector*. Such a detector determines *interest points* that can be repeatedly detected across a wide range of viewing conditions at distinctive locations on the underlying scene or object. Ideally, those points should have only small localization errors both regarding in position and scale.

Lowe's SIFT detector follows the scale-space theory to determine interest points across different scales, such that the estimated scales of the interest points are roughly covariant with

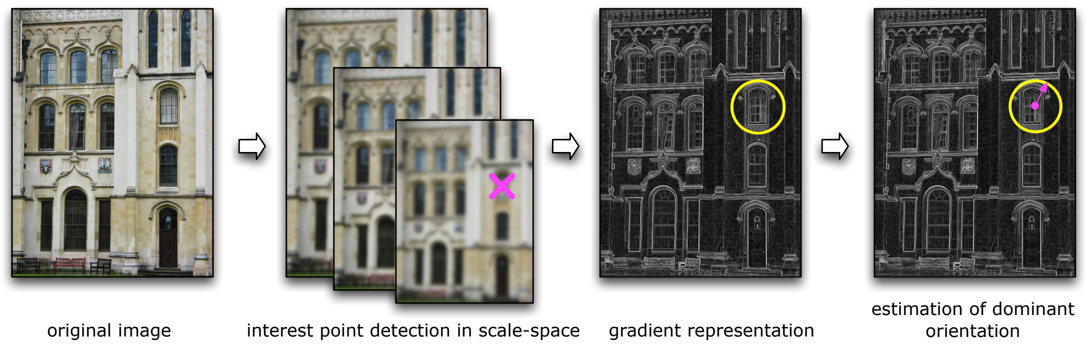

original image
interest point detection in scale-space
gradient representation
estimation of dominant orientation

Figure 2.1: SIFT: Interest point detection - From left to right: First interest points are detected in scale-space by determining the scale for which the DoG has the maximum response with subsequent localization refinement and edge suppression. The dominant orientation is estimated by measuring the strongest gradient directions in the underlying image patch.

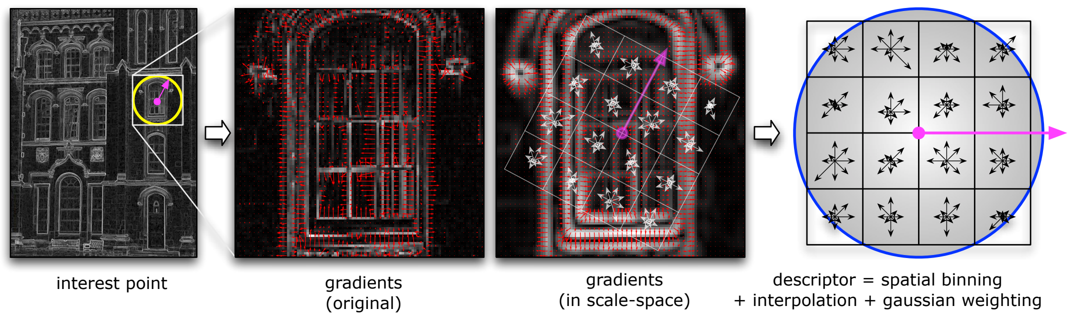

interest point
gradients (original)
gradients (in scale-space)
descriptor = spatial binning + interpolation + gaussian weighting

Figure 2.2: SIFT: Descriptor computation - From left to right: The local region surrounding an interest point is described by aggregating gradient magnitudes into a histogram-like descriptor that encodes gradient orientations, their strength as well as their spatial distribution. The gradient magnitudes are binned into 4×4 spatial cells and 8 orientation bins with interpolation between adjacent bins and a Gaussian weighting by the distance to interest point.

the image size of the underlying object or scene. For efficient computation Lowe adopted the Difference-of-Gaussian (DoG) function as used by Lindeberg (1993; 1994; 2007) that approximates the Laplacian-of-Gaussian function. Its response in scale-space is used to determine extrema locations that are candidates for interest points.

Starting with the original image the scale-space is built incrementally by repeatedly convolving the image with a Gaussian filter resulting in an increasingly blurred sequence of images. This process continues over multiple octaves; after each octave the blurred image is halved to save computations. The pyramid of DoG response maps is then built by computing the difference map between two adjacent levels in the scale-space. Local extrema in the DoG response maps are found by a 8-neighborhood intensity comparison across three adjacent scale levels.

The interest point is initially localized by the pixel position of the extremum and the corresponding scale level in the DoG pyramid. It is then further refined both spatially and in scale-space with sub-pixel accuracy: A 3-D quadratic function is fitted to the location in scale-

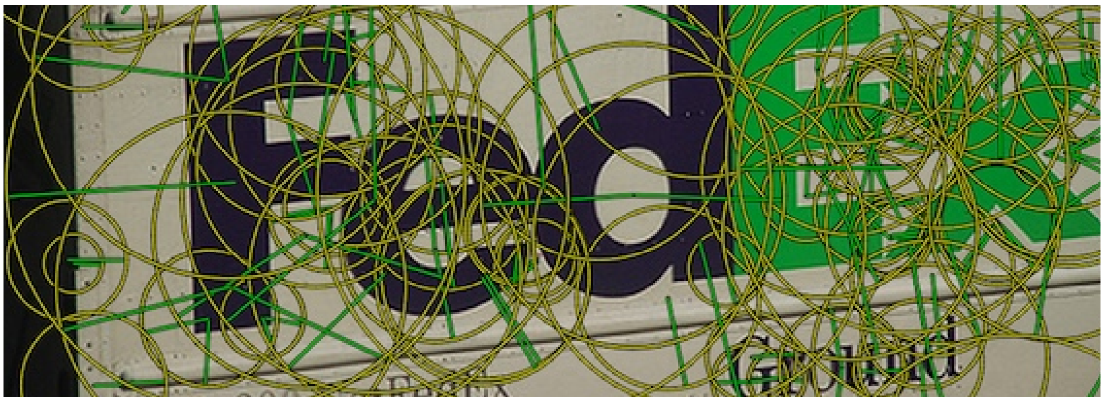


Figure 2.3: Example for sparse SIFT features computed from Difference-of-Gaussian interest points. The image patch captured by the descriptors is shown as yellow circle. The green line denotes the dominant orientation estimated for each interest point.

space (i.e., to x, y, $scale$) by a 2nd-degree Taylor-expansion on the gradient samples (Brown and Lowe 2002). The final interest point location is determined as the extremum of this function. This refinement is important, especially for image registration, since interest point locations detected on different scale levels are back-projected to the original image resolution.

To prevent unstable detections, a threshold on the DoG response at the location of the refined extremum discards spurious detections at points with little contrast. To reduce the responses along edges an additional edge suppression step is applied: Similar as in the Harris corner detector (Harris and Stephens 1988) the ratio between trace and determinant of the Hessian at the interest point is exploited to discard points on edge-like structures.

Once the interest point has been localized, the dominant gradient direction of the underlying image patch is estimated to allow the visual description being transformed into a rotation-normalized canonical frame. The orientation estimation is done by building a histogram of gradient orientations that measures which gradient directions occur most in the underlying image patch. The gradients of the neighborhood around the interest point are sampled at the appropriate scale-space level and added to the orientation histogram whereby each sample is weighted by its gradient magnitude and the distance to the interest point. Last, a quadratic function is fitted to the discretized orientations; the peak of the fitted parabola yields the final orientation estimation.

The SIFT detection scheme is sketched in Figure 2.1. Examples of interest points and their estimated dominant orientation are shown in Figure 2.3.

SIFT Descriptor

The SIFT descriptor represents the image region around an interest point by the spatial distribution of the underlying image gradients. Gradient samples provide both the gradient direction as well as the gradient strength (magnitude). The spatial distribution of these gradients in the descriptor window yields a highly distinctive yet robust visual description.

To reduce the size of the visual description to a relatively small descriptor that is also independent of the patch size, the gradients are quantized both spatially as well as by their direction into a 128-dimensional real-valued histogram of gradient orientations. The spatial information is encoded by a 4×4 grid of cells, whereby each cell encodes underlying gradients by 8 different angles. The size of a single grid cell depends linearly on the scale of the interest point, i.e., the scale of the interest points is multiplied with a magnifier that controls the described patch size and may be chosen depending on application requirements. To obtain a rotation invariant visual description the gradients are encoded into a canonical frame: All gradient orientations are measured relative to the estimated orientation of the interest point.

A minor change on the descriptor location – such as small shifts or minor scale changes – should only have minor effect on the computed visual signature. To achieve this, the gradients are interpolated bi-linearly into the cells of the spatial histogram. Thus, a single gradient sample (at a single pixel) contributes to multiple cells with weights according to its position on the underlying grid. In the same spirit, the gradient directions are interpolated into adjacent orientation bins. Each entry is weighted by the gradient magnitude such that the gradient histograms closely resemble the distributions of strong and weak edges in the underlying image[1]. In addition to the aforementioned interpolation all gradient weights that are added to the histogram are further weighted by a Gaussian on the distance to the interest point. This descriptor computation scheme is illustrated in Figure 2.2.

Eventually, the robustness of the descriptor to changes in lighting is improved by normalization: To achieve invariance to changes in the overall image intensity the descriptor is first L_2-normalized. To further reduce the dominance of extremely strong edges - in contrast to their surroundings - large peaks in the histogram are clipped and finally the descriptor is L_2-normalized to unit length again. This scheme results in a visual description that is robust across a wide range of varying lighting conditions and also somewhat robust to slight localization errors and changes in perspective (roughly up to 40°).

2.1.2 RootSIFT

RootSIFT (Arandjelović and Zisserman 2012b) is a relatively new variant of the SIFT descriptor with a minor change that has major impact: The SIFT descriptors are computed as normal, but the descriptors are normalized in a different way before the square root of each element is taken. The steps to obtain a RootSIFT descriptor $\mathbf{x}'$ are as follows:

1. The regular SIFT descriptor $\mathbf{x}$ is computed and L_1-normalized afterwards.[2]
2. For all descriptor elements the square root is taken: $\mathbf{x}'_i = \sqrt{\mathbf{x}_i}$.

[1]The whole process can also be seen as a *trilinear* interpolation of gradient samples into a 3-D histogram holding both spatial bins and orientation bins.

[2]Usually SIFT descriptors are L_2-normalized, truncated to damp down peaks in the gradient histograms and L_2-normalized again. In case of RootSIFT the last L_2-normalization is replaced by an L_1-normalization.

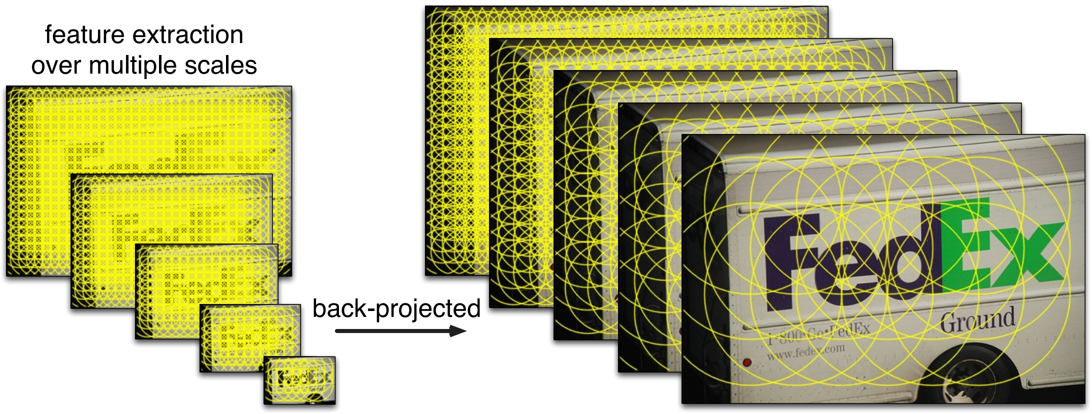


Figure 2.4: Dense SIFT: Multi-scale computation of local features sampled from a regular grid.

Intuitively, the square-rooting of the descriptor elements reflects a down-weighting of large gradients. The differences between large gradient magnitudes carry less information than differences between small gradient magnitudes. Consequently large gradient magnitudes are down-weighted. More general, this scheme is also known as power-law normalization with $\mathbf{x}'_i = \mathbf{x}_i^\alpha$. In slightly different settings it has been explored earlier with even stronger down-weighting i.e., $\alpha = \frac{1}{3}$ by Lejsek et al. (2006) or recently with $\alpha = 0.2$ by Delhumeau et al. (2013).

This minor change improves the visual description significantly, yielding higher performance for both retrieval and classification as shown by Arandjelović and Zisserman (2012b). We experimentally confirm this in Chapter 7.

2.1.3 Dense SIFT

For most methods that rely on image descriptions representing the image content by occurrences of local features, the total number of sampled local features is critical. Indeed, even random sampling of patches from an image increases performance over determining these patches by interest point detectors as long as the number of randomly sampled patches is high (Nowak et al. 2006). Consequently, it turns out to be beneficial that the image content is not described by local features sampled from sparse interest points but from a regular grid (Hörster et al. 2008a). The "dense" sampling of overlapping visual features in regular intervals increases the number of features dramatically compared to features computed from sparse interest points. This increases the chance that certain object parts are properly captured by multiple overlapping visual features and it further encodes regions where interest point detectors do not fire e.g., due to little contrast. Sparse interest point detectors discard such homogeneous regions by design. However, even homogeneous or low-contrast-regions carry visual information about the context of the object or scene. For instance, these may encode sky, a wall or simply homogeneous regions within the object's contour. The downsides are increased memory requirements and computational costs. The former makes a subsequent compression[1] step mandatory,

[1]Compression in the sense of data reduction not necessarily lossless compression.

the latter can be reduced drastically by exploiting the regularity and overlap of dense visual features (Uijlings et al. 2009, Vedaldi and Fulkerson 2010).

In the spirit of a multi-scale sliding window search the SIFT features are computed on multiple scales to achieve pseudo scale-invariance. Starting with the highest image resolution, i.e., the lowest scale, which is commonly down-sampled from the original resolution to save computations, the image is consecutively down-scaled. The down-scaling (usually by a scale factor of $2^{\frac{1}{2}}$ or $2^{\frac{1}{4}}$) is performed multiple times down to a minimum size, or equivalently up to a maximum scale of the resulting features. Unlike the Gaussian smoothing in scale-space it might be beneficial to slightly sharpen the image after down-scaling[1]. For each scale level features are computed from a dense regular grid and stored with their scale and location information. The extraction scheme of dense local features over multiple scales is illustrated in Figure 2.4.

The sheer amount of local features per image prohibits techniques like direct feature matching or visual word-based lookup in an index. However, in Chapter 4 we show a technique that is able to compress the visual information into small signatures.

2.1.4 Histograms of Oriented Gradients

Originally Dalal and Triggs (2005) have developed the *Histograms of Oriented Gradients (HOG)* features for human detection. Later these have been extended by Felzenszwalb et al. (2010) in the context of object recognition. We used the latter improved variant in this thesis.

HOG features are conceptually similar to SIFT as they also describe image gradients by a histogram of gradient orientations. However, the encoding is fundamentally different: As for dense SIFT the HOG features are computed over multiple scales by downscaling the image and repeatedly computing the features. For a single scale the corresponding image is divided into a grid of square cells. The gradients in each cell are described by a single HOG descriptor whereby image gradients are binned into spatially adjacent cells with bilinear interpolation. In contrast to SIFT there are usually much more orientation bins (i.e., 18 directed and 9 undirected orientations in the variant of Felzenszwalb et al. (2010)) to allow fine-grained distinction between object contours at classification time.

For object detection the HOG descriptors are usually arranged in templates: the descriptors of multiple cells are concatenated into a single descriptor for a certain template of e.g., 6×6 cells. However, for retrieval where data reduction as with bag-of-words is necessary, we aggregate the HOG descriptors of 2×2 neighboring cells into a single descriptor which is then quantized to a single visual word (Xiao et al. 2010). This aggregation encodes spatial information of adjacent cells but retains the property that this descriptor only weakly describes the spatial layout of image gradients, making it somewhat complementary to dense SIFT.

[1]There a few indicators: Dalal and Triggs (2005) showed that the omission of smoothing improves classification based on templates of gradient histograms. The sharpening after resizing itself is a standard operation in many image viewers justified by visual inspection. Vedaldi and Fulkerson (2010) further suggest in the documentation of the VLFeat library that it might be beneficial to “undersmooth” the image.

2.2 Visual Vocabularies

When indexing images, the corresponding visual descriptions should require little memory to allow large databases and fast searching. In practice, storing high-dimensional visual descriptors is in most cases infeasible beyond a few thousands images. Sivic and Zisserman (2003) first proposed to build *visual vocabularies* or *visual dictionaries* by clustering descriptors. The clustering groups similar descriptors into the same cluster by minimizing the intra-cluster distances. A descriptor is then represented by the identifier of the cluster it belongs to – a single number.

This approach has several benefits: (1) The representation of the descriptor by a single number requires only a fraction of memory. If further speeds up subsequent processes and allows to determine similar visual features by comparing their identifiers. (2) Each cluster may be seen as the conceptual counterpart of a textual word in the visual domain and is consequently termed *visual word*. In contrast to the real-valued high-dimensional descriptor vectors, clusters are represented by a finite number of discrete identifiers. This allows to apply techniques from the textual domain such as natural language processing to visual features.

To sum it up, due to the compact representation of visual descriptors as single integers, databases of millions of images can be efficiently searched for similar visual descriptions. We describe the use of visual words in the context of the *bag-of-words* model in Section 2.3 in detail. In this section we present three different clustering techniques to derive visual words. Each of these techniques groups high-dimensional descriptor vectors into clusters such that the corresponding cluster can be used as representative.

Clustering is an important building block for many applications and research areas. Consequently, both quantity and variety of those techniques is huge. However, for the task of visual vocabulary construction most clustering techniques are ruled out by the large number of desired clusters which lead in turn to huge amounts of training data. In practice k-means and its variants are widely used as these are simple and easy to parallelize. In this thesis we use several variants of k-means for clustering visual features and to derive a discrete set of visual words.

2.2.1 k-means

One of the oldest and most well-known clustering techniques is k-means (Macqueen 1967, Forgy 1965, Lloyd 1982). Despite its simplicity it is widely used and its principle serves as foundation for several more advanced methods. The basic idea is to group data samples by their similarity into clusters by maximizing the intra-cluster similarity between those members. The similarity between data samples is hereby measured by a distance function. k-means then produces clusters that minimizes the intra-cluster distance between those data samples belonging to a certain cluster.

The outcome of this clustering method are k cluster centers or "means" – hence the name – obtained by averaging those vectors belonging to the cluster. The cluster mean – also termed “centroid” or “center” – can then be used as a representative for those vectors within the particular cluster. k-means requires the number of clusters k to be specified by the user. There are methods to determine k automatically from data but mostly it is chosen manually. For visual vocabularies, we found that the number of clusters is application-dependent and critical for good retrieval results. Thus, we set the number of clusters ad hoc and do a parameter sweep over the number of clusters if necessary.

While k-means can be generalized to other metrics, the Euclidean distance is probably the most important distance in practice. As the similarity between visual feature descriptors is usually measured by the Euclidean distance we use this metric throughout this thesis.

2.2.1.1 The *k*-means Algorithm

A cluster k is represented by its centroid $\boldsymbol{\mu}_k$ and has an associated member set $\mathcal{C}_k$ describing which data samples are contained in this particular cluster. The clustering objective is to partition a set of N data samples $\mathbf{x}_i$ from a D-dimensional space $\mathbb{R}^D$ into K different subsets – termed clusters – such that those vectors belonging to the same cluster are similar in the sense that their pair-wise distances are small. To achieve this, k-means minimizes the quadratic error:

$$\sum_{k=1}^{K} \sum_{\mathbf{x}_i \in \mathcal{C}_k} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2 \rightarrow min \tag{2.1}$$

In other words, k-means seeks for the cluster configuration with the smallest intra-cluster error. Unfortunately, there is no guarantee that k-means will find the global minimum when seeking for the configuration with the smallest intra-cluster error. It is known to be particularly prone to finding local minima instead. Consequently many improved variants were proposed that aim to increase the chance that k-means finds the global optimum. Many of these target the initialization of k-means. In *k-means++* (Arthur and Vassilvitskii 2007) the initial clusters are drawn from a random distribution such that these are likely “far away” from each other. Another common strategy is to perform multiple k-means runs with different initializations and to take the clustering with the smallest error. We have tried both but found negligible differences in the outcome for our application - probably due to the very densely populated features space of visual features.

At the beginning all cluster centroids $\boldsymbol{\mu}_k, k \in \{1, ..., K\}$ are initialized by selecting a random yet unique data sample $\mathbf{x}_i$. The k-means clustering is then performed iteratively by repeating the following three steps:

1. The cluster assignments are reset: $\forall k : \mathcal{C}_k \leftarrow \emptyset$.
2. Each vector $\mathbf{x}_i$ is assigned to its nearest cluster. That is, that the cluster k whose centroid $\boldsymbol{\mu}_k$ has the minimum distance to $\mathbf{x}_i$ is determined and $\mathbf{x}_i$ is added as cluster member:

$$\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \{\mathbf{x}_i\} \quad \text{with} \quad k = \underset{k'}{\operatorname{argmin}} ||\mathbf{x}_i - \boldsymbol{\mu}_{k'}|| \tag{2.2}$$

3. The cluster means are (re-)computed from their corresponding cluster members as determined in step 2:

$$\forall k : \boldsymbol{\mu}_k = \frac{1}{|\mathcal{C}_k|} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \mathbf{x}_i \tag{2.3}$$

This iterative procedure converges to a local minimum of the error as given in Equation 2.1. In practice, due to slow convergence and numerical issues the two steps are usually repeated until one of the following termination criteria is met: The iterations are stopped if the centroids do not move anymore (no centroids shifts more than ϵ during two subsequent iterations) or if a fixed number of iterations is reached.

This scheme is known as *Lloyd's algorithm* (Lloyd 1982) whereas the two steps themselves are sometimes referred to as *Voronoi iteration* as the resulting partition of the space is a Voronoi tessellation. There are several variations to these steps: In Lloyd's method the cluster means are computed once in each iteration after all data samples have been assigned to their nearest cluster. This is also sometimes referred to as "batch update". However, similar to Stochastic Gradient Descent, one may alternate step 1 and 2 while iterating over all data samples such that each sample incrementally updates the corresponding cluster mean. In our implementation we implemented the former technique as it is widely used and both steps can be parallelized easily which is important for clustering large datasets.

It can happen that a cluster loses all members from one iteration to the next. Thus, empty clusters get a randomly chosen member from another non-empty cluster re-assigned. The latter is chosen randomly e.g., among the top 25% percentile of largest clusters, such that this procedure tends to split the densest regions in feature space during the next iteration.

In contrast to the problem of finding the exact solution, which is known to be NP-hard, the complexity of k-means is $\mathcal{O}(NK)$. Thus, its complexity scales linearly with the number of samples and the number of clusters, which makes k-means suitable for clustering high-dimensional data into thousands of clusters. In addition, the clustering is relatively fast and the quantization of vectors only involves plain distance computations in contrast to more complex schemes such as simulated annealing (Selim and Alsultan 1991), agglomerative clustering schemes (Day and Edelsbrunner 1984) or Affinity Propagation (Frey and Dueck 2007).

2.2.1.2 Quantization

A quantization function is a function $q(\mathbf{x}_i)$ that maps the high-dimensional, real-valued vector $\mathbf{x}_i$ from $\mathbb{R}^D$ to a discrete number $v_i \in \mathbb{N}$. The quantization of a descriptor to a single number – its representative visual word label – is an intrinsic step in the k-means method: A visual word v_i is obtained from the corresponding descriptor $\mathbf{x}_i$ by finding the centroid being closest to $\mathbf{x}_i$ as during each clustering iteration as given in Equation 2.2 and using its ID as visual word:

$$v_i = q(\mathbf{x}_i) = \operatorname*{argmin}_k ||\mathbf{x}_i - \boldsymbol{\mu}_k|| \tag{2.4}$$

The quantization maps all cluster members to the same visual word, thus all descriptors that are quantized to the same visual words are similar in the sense that their distance to the cluster centroid was smaller than to any other cluster.

2.2.2 Hierarchical k-means

The complexity of k-means is linearly dependent on the number of clusters as well as the number of samples. Therefore, creating large vocabularies – which in turn require a large number of training samples – is prohibitive. Even worse, during quantization the distance between a query descriptor and all cluster centroids needs to be computed. While visual vocabularies are usually computed off-line once in advance, this linear exhaustive search makes such "flat" vocabularies undesirable for applications that use large vocabularies beyond a few thousand words.

Hierarchical k-means is an extension that offers more efficient clustering and more efficient quantization than standard k-means. Nistér and Stewénius (2006) proposed hierarchical k-means to build a vocabulary tree where k-means is used to obtain a hierarchy of clusters that hierarchically partition the feature space. The main goal is to reduce both the time for clustering as well as for querying the tree – when quantizing a query vector. It is realized by reusing information resulting from previous clusterings and exploiting the hierarchical relationship: The whole feature space is partitioned recursively into sub-spaces. Just like in other tree structures the number of nodes that need to be visited when looking for the nearest neighbor is logarithmic to the number of leaves that eventually form the centroids of the vocabulary. At the same time, the partitioning is performed by hierarchical clustering from coarse to fine. In contrast to conventional k-means in each step only a fraction of the full dataset needs to be kept in memory eventually leading to faster clustering.

At each tree level the data samples are clustered with conventional k-means. The resulting clusters divide the parent feature space into k subspaces. Continuing this scheme yields nested subspaces where each subspace is a child of its parent space and is further recursively divided into sub-subspaces. The tree structure is directly inferred from the relationships between these subspaces: Each tree node represents a subspace with its centroid vector and the corresponding child and parent subspaces. The number of sub-spaces per node is termed *branch factor* and is equivalent to the number of clusters k that k-means produces when sub-dividing each partition.

Figure 2.5 shows an example of an (incremental) hierarchical clustering of a vocabulary tree with a branch factor of 3 over 3 tree levels.

Quantization Once the vocabulary tree is constructed, quantizing a descriptor to a single number as its representative is straightforward. Starting at the root node the input vector is propagated down in the tree as shown in Figure 2.6. In the first level (level 0) the whole feature space is represented by the root node, which in turn is partitioned into 3 sub-spaces represented by its child nodes. The query vector is thus compared to these sub-spaces by computing the Euclidean distance between the input vector and each of the k cluster centroids (red nodes).

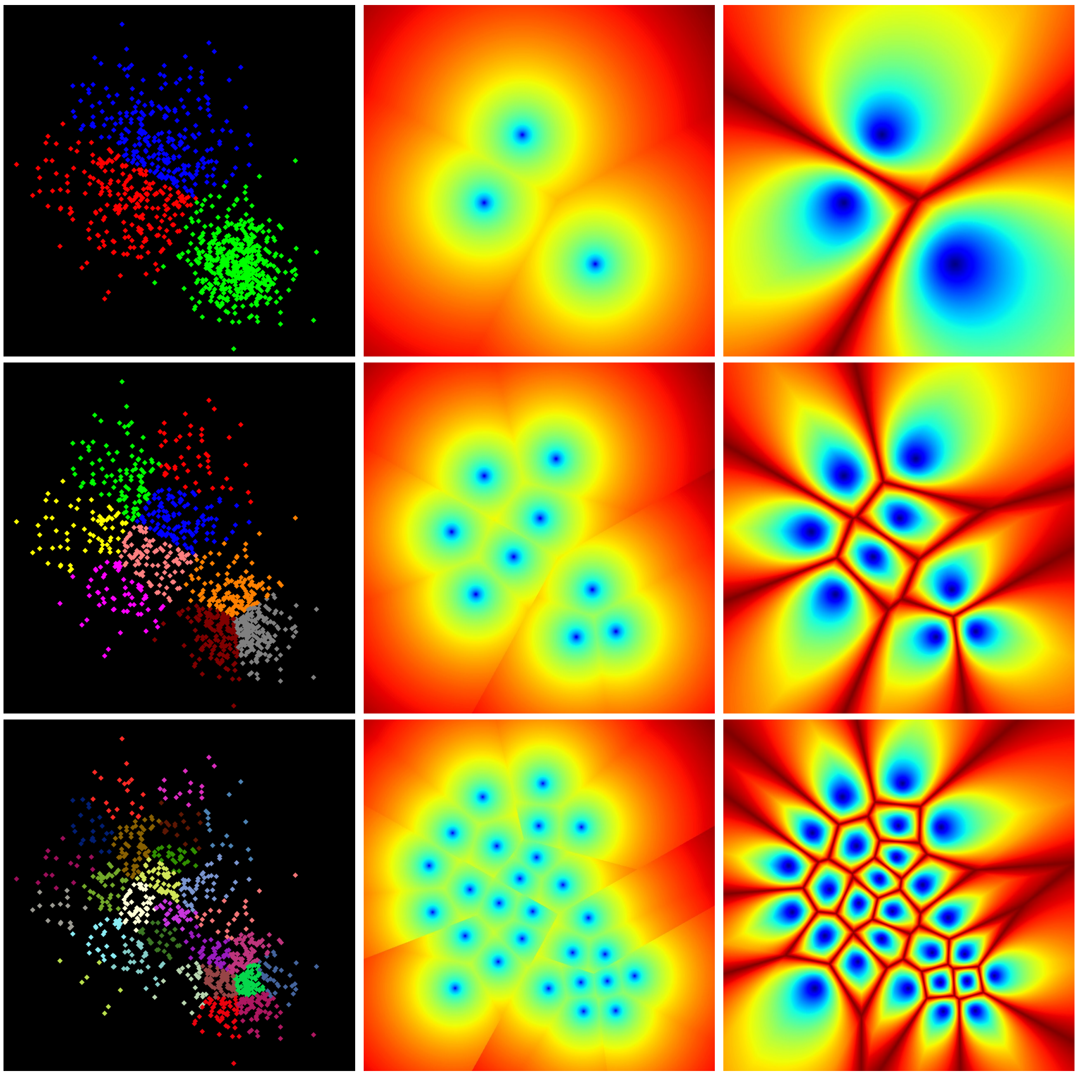


Figure 2.5: An example of hierarchical clustering with hierarchical k-means: on each level the clustering was created with k-means and $k = 3$. **From top to bottom:** Clustering at different tree levels with 3^1, 3^2 and 3^3 clusters. **Left column:** Visualization of data points assigned to each cluster. All points within a cluster have the same color – the clusters are colored differently. **Center column:** Visualization of the distance to the centroids. **Right column:** Visualization of the distance ratios. This ratio describes the distance to the closest centroid relative to the distance to the 2nd closest centroid. The visualization – as in Schindler et al. (2007) – clearly reveals the Voronoi cells.

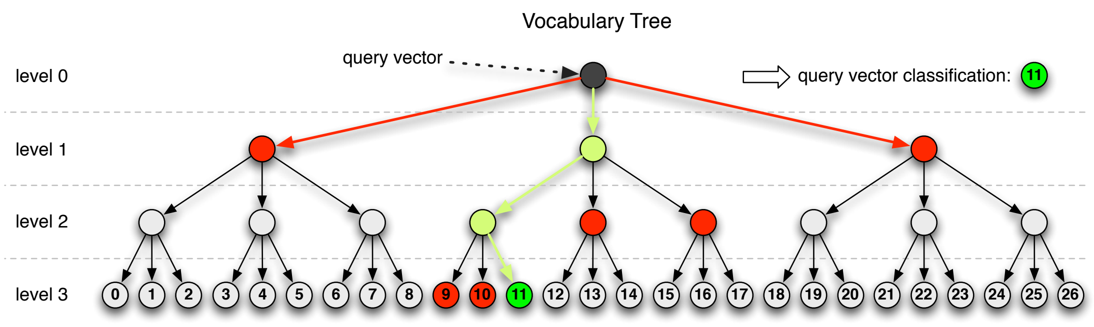


Figure 2.6: An example of querying a vocabulary tree with a branch factor of $k = 3$ and 3 tree levels (the level of the root node is not counted). Red nodes are cluster centroids that are compared to the input vector. Green nodes are cluster centroids that are compared to the input vector and turn out to be the most similar ones. Gray nodes are not involved at all.

The nearest cluster is then chosen as subspace where the query continues (green nodes). That is, the input vector is propagated down to the child node representing this nearest cluster and rooted at the corresponding sub-tree. This procedure continues recursively and terminates once a leaf node is reached. By enumerating all leaf nodes these can be used as visual word identifiers that represent the input vector. Hence, the size of the vocabulary is determined by the number of leaf nodes, which in turn is determined by both branch factor and the tree's height. The inner nodes of the tree are only needed to propagate input vectors down the tree.

The number of comparisons i.e. distance computations when propagating the query down the tree is $\mathcal{O}(K \log_K(N))$ where N denotes the number of leaf nodes and K the branch factor at each level. For instance, quantizing a vector with a vocabulary of 1 million clusters and a linear scan would require $1,000,000$ distance computations, which is infeasible in practice. A vocabulary tree with 3 levels and a branch factor of 100 reduces this to only 300 distance computations. A binary tree with $k = 2$ is optimal with respect to the number of distance computations[1] but has higher memory overhead due to the large number of inner nodes.

Hierarchical k-means has the major advantage that is able to deal with a huge amount of training samples: only the samples required for clustering a single node need to be present in the memory at the same time. For reasonable choices of the branch factor – usually between 2 and 1000 – the clustering is quite fast. Furthermore, as the tree grows, the data samples falling into each node are clustered into finer-grained sub-spaces. That is, the clustering performed at each node is able to exploit training samples for further partitioning that are *specific* for this particular region in the feature space. In contrast, conventional k-means is only able to deal with fewer training samples at the same time and thus fine-grained differences may be underestimated as the original training data usually need to be sub-sampled.

Hierarchical k-means also adapts to the training data: The training data may not provide a sufficient number of training vectors to the clustering for a certain sub-space of the original

[1] A tree with 20 levels and $1,048,576$ leaves requires 40 distance computations.

feature space. In this case, the corresponding tree branch stops growing before reaching the maximum desired height of the tree. This prevents over-partitioning of the training data into small meaningless clusters.

2.2.3 Approximate *k*-means

Another clustering technique scalable to a large number of data samples and millions of clusters is *approximate k-means (AKM)*. This method employs the same k-means iterations as standard k-means but replaces the exact distance computations by approximated ones. The speed-up comes from the sub-linear search of the nearest centroids in each k-means iteration eventually allowing both a higher number of clusters for partitioning the feature space and more data samples to be used. Once the vocabulary is built, the approximate nearest neighbor search also speeds up the quantization of high-dimensional descriptors to visual words.

While the name not further specifies which specific technique is used for the approximate nearest neighbor search, commonly a forest of randomized kd-trees is used for this purpose (Philbin et al. 2007, Muja and Lowe 2009). A predecessor – the "Filtering technique" – uses conventional kd-trees for exact but yet speeded-up distance computations (Kanungo et al. 2000; 2002). Beis and Lowe (1997) and Lowe (2004) then proposed the *Best Bin First* strategy that allows for approximate distance computations at lower computational costs. In the sense of doing multiple independent nearest neighbor searches being merged afterwards, Silpa-Anan and Hartley (2008) utilized a single priority queue across multiple trees and proposed randomized kd-trees, rotated kd-trees as well as PCA-aligned kd-trees that either have data-aligned or randomly chosen split hyperplanes. In similar manner, Jia et al. (2010) select partition axes from binary combinations of coordinate axis to improve the space partitioning.

2.2.3.1 Approximate nearest neighbor search

A classic kd-tree is a binary space-partitioning scheme for indexing and nearest neighbor search of k-dimensional vectors. The vector space is hierarchically split into partitions until each indexed data point occupies its own partition. The kd-tree is built recursively top-down: At each tree level one of the k dimensions is selected and the set of vectors is split into two disjoint partitions by comparing the vector values at that dimension with a split value. Commonly, the split value is chosen as the median value of the selected dimension among the data samples. Thus, each tree node effectively splits the data into halves. This procedure continues recursively until each partition contains only a single data point. At each tree level the split dimension is varied cyclically and the split value is computed from the corresponding dimension of the data subset in the partition that is to be split. The result is a balanced binary tree of height $\log_2 N$ where N denotes the number of data points. Each leaf node represents one data point and the corresponding vector. Each inner node splits its parent space into two sub-spaces such that the original vector space is split recursively by orthogonal hyperplanes into increasingly fine-grained partitions.

The nearest neighbor search for a query vector is performed by first finding an initial leaf node close to the query and then examining adjacent nodes to determine the true nearest neighbor. To find the initial leaf node, the tree is traversed by comparing the split value of inner nodes with the corresponding dimensions of the query vector. Depending on the comparison the search continues either on the left or right sub-tree of the corresponding inner node until a leaf node is reached. The corresponding data point is an initial nearest neighbor candidate but not necessarily the correct one. To find the true nearest neighbor the search must further continue and evaluate the distances from the query to adjacent tree cells: the true nearest neighbor may be just beyond the partition border of the initial cell but still closer than the initial candidate.

This backtracking process is performed in a branch-and-bound manner: During the search for the initial leaf node, the descent down in the tree is basically a depth-first search for the best kd-tree bin. At every inner node its split value is compared to the corresponding dimension of the query vector. The outcome of the comparison determines whether the descent follows the left or right branch of the current node, which represent the sub-spaces of the hierarchical binary space partitioning accordingly. *Branching:* The branches *not* taken are recorded for later examination. The recording may be done either implicitly by the recursion or explicitly by pushing each branch keyed by the 1-dimensional distance between split value and query vector in the corresponding dimension to a priority queue. Once the descent reaches a leaf node, the true distance between the query and the point represented by the leaf node is computed. This distance is the initial upper bound as the distance between any true nearest neighbor and the query must be closer or equal to this initial distance.

The backtracking then starts either by unwinding the recursion or by processing the not-yet-examined tree branches from the priority queue in ascending order of their distance estimate. *Bounding:* Tree branches that are further away from the query than the current upper bound on the distance are not inspected but discarded immediately. For that, it is checked whether the current search radius around the query point – given by the current upper bound on the distance – intersects the kd-tree cell of the corresponding branch. This check is done by computing the 1-dimensional distance between query vector and the corresponding split value, leading to a highly efficient bounding criterion. If there is an intersection, the search continues in that branch, otherwise the search restarts in a different branch either from recursion or by fetching the next branch from the priority queue. As both tree traversal and bounding are performed by operations on a single dimension of the underlying vector space, tree branches are explored efficiently. Once a leaf node is reached that has a smaller distance to the query than the current upper bound on the distance, the latter is updated and the search continues. An exact nearest neighbor search terminates if no more tree branches are left for inspection. An approximate nearest neighbor search may stop after inspecting a fixed number of branches.

The backtracking by recursion can be an expensive process especially in high-dimensional spaces. The nearest neighbor search with a conventional kd-tree (Bentley 1975, Friedman et al. 1977) is often reported to be inefficient in high dimensional spaces, e.g., greater than

8 dimensions in Beis and Lowe (1997). Consequently, Arya and Mount (1993) proposed a more efficient yet approximate search by using a priority queue to process the adjacent cells in increasing order of the distance to the query point. Independently, Beis and Lowe (1997) proposed the very similar *Best Bin First* technique. The main difference is that the former method uses a distance cutoff while the latter uses a constant time cutoff by only examining a fixed number of cells with the help of a priority queue.

Silpa-Anan and Hartley (2008) first proposed to use multiple randomized kd-trees in parallel to improve the accuracy of the approximate nearest neighbor search. Here, not only a single kd-tree but multiple kd-trees built from the same data are traversed with the Best Bin First search strategy. The idea is to build trees that differ in their internal structure and exploit them as if each tree was used in an independent search. Such kd-forest is built from randomized kd-trees that choose its split dimension randomly among the 5 dimensions with the highest variance such that the search is performed over multiple different orthogonal splits of the original data space. In addition, after parallel descent in each tree to find the initial leaf nodes, a single global priority queue is used to traverse the tree cells across multiple trees in the order of increasing distance to the query vector. This technique was extensively compared to other nearest neighbor-search techniques by Muja and Lowe (2009).

In this work, we adopt this technique and use a forest of 8 randomized kd-trees to index the visual word centers. This kd-forest then allows to perform approximate nearest neighbor search to find the nearest cluster for a descriptor vector both during clustering as well as when quantizing descriptor vectors to single visual words. To make our results comparable to other works in the literature we fix the number of backtracking checks to 768 (Philbin et al. 2007).

2.2.3.2 Clustering with approximate nearest neighbor search

The approximate k-means clustering that partitions the data samples $\mathbf{x}_i$ into K clusters with the corresponding centroids $\boldsymbol{\mu}_k, k \in \{1, ..., K\}$ is performed as follows:

1. First all cluster centroids $\boldsymbol{\mu}_k$ are initialized by selecting a random unique data point $\mathbf{x}_i$ or with more advanced seed generation techniques as for regular k-means (e.g., k-means++).
2. The cluster assignments are set to $\mathcal{C}_k \leftarrow \emptyset$.
3. A randomized kd-forest $\mathcal{F}_{\boldsymbol{\mu}}$ is built that indexes the centroids for nearest neighbor search.
4. Each vector $\mathbf{x}_i$ is assigned to its nearest cluster k as follows:

$$\mathcal{S} \leftarrow \mathcal{F}_{\boldsymbol{\mu}}(\mathbf{x}_i) \tag{2.5}$$

$$k = q(\mathbf{x}_i) = \underset{k'}{\operatorname{argmin}} \, ||\mathbf{x}_i - \boldsymbol{\mu}_{k'}|| \, , \; \boldsymbol{\mu}_{k'} \in \mathcal{S} \tag{2.6}$$

$$\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \{\mathbf{x}_i\}, \tag{2.7}$$

Here, the randomized kd-forest $\mathcal{F}_{\boldsymbol{\mu}}$ is used to find a set $\mathcal{S}$ of potential nearest centroids. The vector $\mathbf{x}_i$ is then added to the cluster where the corresponding centroid has the minimum distance to $\mathbf{x}_i$. Note that this formulation only sketches the outcome of the nearest neighbor search with a forest of kd-trees; it does not really reflect its internal operations. In practice, the steps of both Equations 2.6 and 2.7 are inherently linked and performed implicitly when traversing the forest. The forest $\mathcal{F}_{\boldsymbol{\mu}}$ then finds the centroid that has the minimum distance to $\mathbf{x}_i$ with high probability or a centroid with a small distance otherwise.

5. The cluster mean is computed as for conventional k-means from all cluster members:

$$\boldsymbol{\mu}_k = \frac{1}{|\mathcal{C}_k|} \sum_{\mathbf{x}_i \in \mathcal{C}_k} \mathbf{x}_i \tag{2.8}$$

6. Steps 2-5 are repeated until the clustering converges i.e. the centroid vectors move less than ϵ between subsequent iterations or a maximum number of iterations is reached. In addition, empty clusters are suppressed by replacing the centroid with a randomly drawn data sample, which is especially important during the first few iterations.

This clustering scheme is able to cope with several millions of data samples and millions of clusters. Philbin et al. (2007) claim that at least for a moderate number of clusters the difference between cluster assignments obtained by exact and approximate nearest neighbor search differs by less than 1%.

2.2.3.3 Visual words from approximate *k*-means

Once the visual vocabulary is obtained the quantization of descriptor vectors to visual words is performed in the same manner as during each approximate k-means iteration. The outcome of Equation 2.6 directly yields the cluster ID i.e. the visual word label. In addition it is straightforward to fetch not only the nearest but e.g., the 3-nearest neighbors and the corresponding distances for use with multiple assignment and soft assignment schemes (Philbin et al. 2008). As the forest maintains a priority queue with the potential nearest neighbors during each query the m-closest neighbors are simply determined by selecting the top m entries in this queue.

2.2.4 Comparison of Vocabularies and Quantization Techniques

In order to evaluate the impact of the different techniques for creating vocabularies and searching for the nearest cluster, we compare the retrieval performance on the FlickrLogos-32 dataset (8240 images) in terms of speed and search performance for visual vocabularies obtained by k-means, hierarchical k-means and approximate k-means and their corresponding quantization techniques in the following experiment.

We compute RootSIFT descriptors from Difference-of-Gaussian interest points and measure the CPU time for the feature extraction and the quantization of descriptors to visual words. The experiments are performed with a multi-threaded application on an Intel Xeon X5550 (See

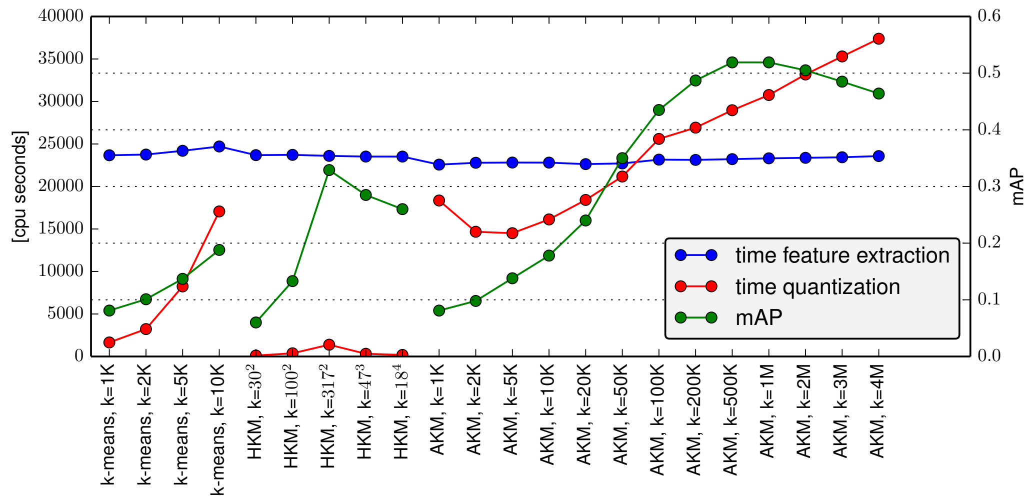


Figure 2.7: Comparison of speed and retrieval accuracy for various visual word quantization methods and various vocabulary sizes k. **Left axis**: Time for feature extraction (blue) and quantization (red). **Right axis**: mean Average Precision (mAP, see Section 3.2.3 for details) for a bag-of-words retrieval with tf-idf weighting (green). *k-means* denotes the quantization by linear nearest neighbor search using a vocabulary obtained by k-means. *HKM* denotes the quantization via a hierarchical k-means tree of depth l and a vocabulary of size k^l. *AKM* denotes the quantization with a forest of randomized kd-trees that indexes a vocabulary obtained with approximate k-means. Note that the vocabulary sizes for k-means and AKM increase from left to right roughly exponentially. Thus, quantization with a forest of randomized kd-trees roughly scales logarithmically with the number of clusters.

Appendix D). The time is accumulated per thread such that the sum is a rough estimate for the time consumption as if the program was executed sequentially.

The results of the timings are shown in Figure 2.7 (left scale). The timings for the feature extraction (blue) are approximately the same across all runs (standard deviation: 2.2%) – as expected – but the time consumption of the quantization (green) varies greatly. One can see that the vocabulary tree (HKM) is by far the fastest quantization method followed by the linear search (k-means) when used with small vocabularies. The quantization by a forest of randomized kd-trees (AKM) has significantly higher computational costs but allows to use large vocabularies with several hundreds of thousands visual words. Starting at around 100,000 (100K) visual words the quantization of descriptors to visual words takes more time than the feature extraction itself. We assume that the main issue here is the complexity of maintaining a priority queue for every query descriptor when traversing the forest of kd-trees in order to find the nearest centroid vector.

We further evaluate how the different vocabularies and quantization techniques affect the retrieval accuracy: we perform a bag-of-words retrieval with tf-idf weighting and measure the mean Average Precision (mAP). For brevity, a more extensive description of the retrieval technique is postponed until Section 2.3; the performance measure is explained in Section 3.2.3.

The results representing the retrieval quality are shown in Figure 2.7 (right scale). Clearly, when comparing the retrieval performance of vocabularies with the same size, the quantization of descriptors to visual words by a linear nearest neighbor search (k-means) – for small vocabularies – or approximate search via a forest of randomized kd-trees (AKM) – for large vocabularies – yield significantly better search results in terms of mAP than visual words obtained from a hierarchical vocabulary tree (HKM). Also vocabulary trees of greater depth perform inferior to trees with fewer levels. This is in line with Philbin et al. (2007); the hierarchical partition of the feature space may often prevent descriptors to be represented by the corresponding cluster that has truly the smallest distance in feature space. This issue may be overcome with more elaborate backtracking methods as shown by Schindler et al. (2007) and Muja and Lowe (2009). In contrast, the linear search and also the approximate nearest neighbor search via randomized kd-trees may search larger portions of the feature space, which are not separated by hard boundaries. Thus the quantization yields better visual word assignments.

2.2.5 Visual Analysis

It is good practice to visually inspect the visual vocabularies (partially) as sanity check. This can be done by inspecting those image patches that are grouped together into the same cluster and therefore are represented by the same visual word. In the following we show several examples for vocabularies of different sizes to give an intuition what the image description actually captures.

Figure 2.8 show several examples of image patches that are described by the visual words computed from RootSIFT descriptors. Each row depicts randomly sampled image patches described by the same visual word. In this case approximate k-means has been used for clustering and the corresponding forest of randomized kd-trees has been used for quantization. More specifically, Figure 2.8 (a) shows examples of visual words from a vocabulary of 10,000 words, Figure 2.8 (b) of a vocabulary with 100,000 words and Figure 2.8 (c) of a vocabulary with 1 million words.

One does observe that some of the patches are highly distinctive while others are not. Also, individual patches alone are most likely not sufficient to recognize an object from a single patch. While the few examples we can present here hardly reflect this, larger vocabularies – especially the vocabulary with 1 million words – tend to have fewer patches associated with certain distinctive words. Also the visual appearance of those patches described by the same visual word tends to be more consistent.

2.3 Visual Words and Bag-of-Words

In Section 2.2 we already described several clustering techniques to obtain visual vocabularies. In this section we discuss implications of this representation and its use for retrieval.

A turning point in scalable image retrieval and object retrieval has been the introduction of the *bag-of-words* approach (Sivic and Zisserman 2003). At its core is the quantization of local features for efficient indexing and retrieval. Here, the real-valued high-dimensional visual

(a)

(b)

(c)

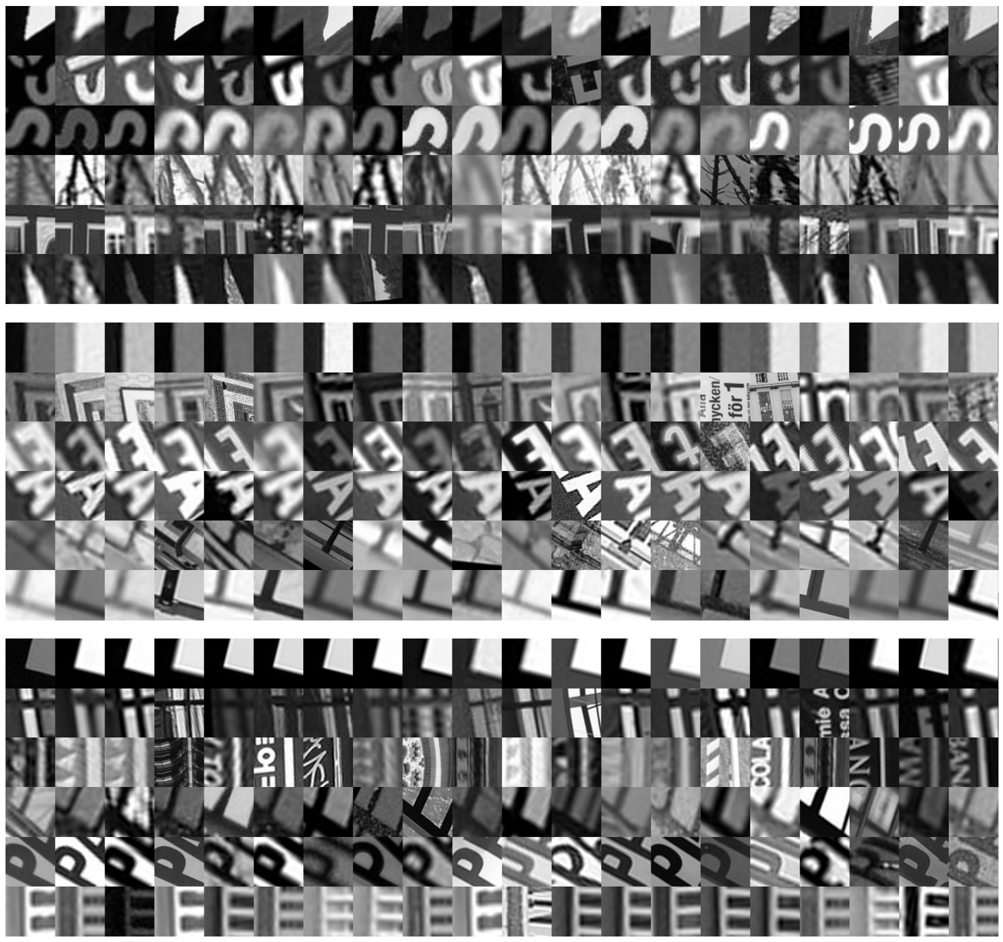


Figure 2.8: Sample patches associated with four different visual word clusters of RootSIFT features computed from DoG interest points. Each row shows patches that are described by the same visual word with a vocabulary of 10,000 (a), 100,000 (b) and 1,000,000 visual words (c). Each vocabulary contains frequent words describing mostly primitive visual structures (top row in each block) and more distinctive words (other rows).

feature descriptors are quantized to discrete visual words that are represented by a single integer. The visual words are usually obtained by clustering feature descriptors. Each cluster by definition contains highly similar descriptors and is represented by the centroid vector and its corresponding cluster label – the visual word. Given these clusters, a descriptor is represented by its most suitable visual word, i.e., the cluster label of that cluster that has the smallest distance to the descriptor vector. As a consequence, each visual word represents all the descriptors that have been grouped together by minimizing the intra-cluster distances.

Instead of matching descriptors by computing distances between the corresponding vectors, these can be matched by determining if these have identical corresponding visual word labels. Thus, once the descriptors have been quantized to visual words the matching of local feature descriptors reduces to comparing integers. This has several major implications:

- The visual word representation is an enormous data reduction. Instead of full descriptor vectors the visual description is stored as single integer number.
- The quantization reduces vectors to a discrete representation that may be counted and processed similar to textual words. This allows the usage of classic text retrieval and data mining techniques for indexing and retrieval of visual words.
- Finding correspondences between two sets of visual features $\mathcal{A}$ and $\mathcal{B}$ could be formulated as bipartite (maximum weighted) graph matching problem, which might take the distances between descriptors into account. However, such an approach is computationally expensive. By using visual words, feature matching can be performed in constant time: correspondences are simply determined by finding those descriptors that are quantized to the same visual word.
- The former concept of feature matching can easily be expanded to databases. By using the visual word label as key, sub-linear search within a database of local features can be performed. Such index structure that maps a visual words to those images that contain it is known as *inverted index* and allows sub-linear search of corresponding visual word matches between a query image and images in the database (Sivic and Zisserman 2003). Consequently the inverted index has become the most important data structure for scalable image retrieval during the last years.

The ability to treat visual features as a discrete entity (e.g., counting and matching) quickly led to the adoption of numerous text retrieval techniques in the visual domain. However, it is well-known that visual words are not as expressive as textual words. Visual words mostly only encode visual primitives (see Section 2.2.5 for a visualization of these) while a text word may have a high-level meaning (e.g., “car”), describe an abstract concept (e.g., “vehicle”) or an action (e.g., “driving”). This main difference between visual and textual words is known as *semantic gap* (Smeulders et al. 2000, Datta et al. 2008). In Chapter 4 we show how topic models can be used to obtain an abstraction layer for visual words that is able to partially deal with the semantic gap.

2.3.1 Image Representation

The most prominent way to model images is the *bag-of-words* model. Hereby, an image is represented by the visual words it contains and the similarity between images depends on the number of visual words that occur in both images. More specifically, the l local feature descriptors $\mathbf{d}_i$ extracted from an image I are quantized to visual words $v_i = q(\mathbf{d}_i), (i = 1, ..., l)$ by a quantizer function q as introduced in Section 2.2.1.2. Thus, each descriptor $\mathbf{d}_i$ is replaced by a discrete visual word label $v_i, v_i \in \{1, ..., V\}$ where V denotes the number of visual words.

The image is then represented by an unordered collection of visual words encoded into a histogram of word occurrence frequencies $\mathbf{x} = (t_1, ..., t_j, ..., t_V)$. Here, t_j denotes the occurrence frequency of visual word j given as $t_j = \sum_{i=1}^{l} \delta(v_i, j)$ where $\delta(v_i, j) = 1$ if $v_i = j$ and 0 otherwise. The collection of words $\mathbf{x}$ is termed *bag*-of-words as it is an unordered collection in the sense that neither the relationship between visual words in feature space nor in the spatial domain are preserved or encoded. Indeed, it only captures the appearance of visual features (by their visual word labels) and their occurrence frequencies.

In other words, an image is modeled by the occurrences of discrete observations without incorporating their spatial distribution. This is a major simplification, but in turn allows using simple similarity measures for classification and search and further provides robustness against rotation, scale change and other image perturbations. In addition it also provides a fixed-size representation for images that may contain a different number of local features. The simplicity and robustness of the bag-of-words model eventually led to its popularity for image retrieval, classification and other use cases.

2.3.2 Vector Space Model

The most popular model for bag-of-words retrieval in information retrieval is the *vector space model* (Salton et al. 1975). It represents documents as vectors where each vector dimension encodes the weighted occurrence frequency of a particular (visual) word. These *document vectors* are usually L_2-normalized in order to give them unit length. In the corresponding vector space the set of all possible documents thus forms a hypersphere[1].

Inherited from the bag-of-words model is the underlying word independence assumption: No relationship between words is modeled and it is further assumed that the individual words occur independently from each other. This assumption is certainly violated in practice, as visual features by design do overlap or are in close proximity and partially describe the same image content. However, it makes resulting image models simpler and computationally feasible.

In the vector-space model the similarity between images is modeled roughly by the distance between the corresponding bag-of-words histograms. While there are various similarity measures the most important and also a well-performing similarity measure in practice is the Cosine similarity from information retrieval. The Cosine similarity of two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as the scalar product of the histograms $\mathbf{x}$ and $\mathbf{y}$ where both histogram vectors are (post-)normalized to unit length:

$$sim_{cos}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{K} \mathbf{x}_i \mathbf{y}_i}{||\mathbf{x}||\,||\mathbf{y}||} \tag{2.9}$$

Here, $||\mathbf{x}||$ and $||\mathbf{y}||$ denote the L_2-norm. The Cosine similarity thus by definition measures the angle between the two unit-normalized vectors in the space of all possible documents. Taking the cosine of this angle yields a score in the interval $[0, 1]$, which is in turn used to rank the retrieval results. See Figure 2.9 for an illustration of the Cosine similarity. For L_2-normalized vectors the cosine similarity equals the dot product: $sim_{cos}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{K} \mathbf{x}_i \mathbf{y}_i$.

[1] If the weighted occurrence frequencies are all positive, these live on a quarter-hypersphere.

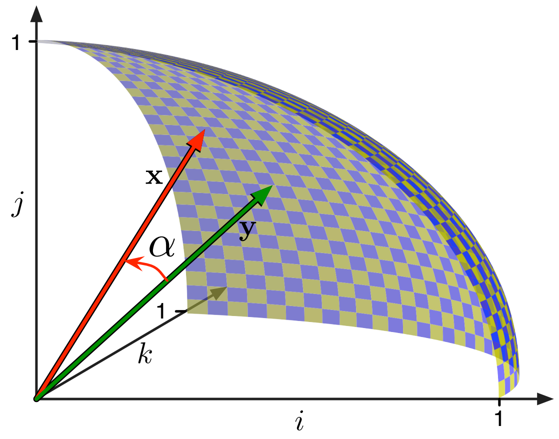


Figure 2.9: The vector space model for vectors in $\mathbb{R}^3$ and its relationship to the cosine similarity: The similarity between two document vectors $\mathbf{x}$ and $\mathbf{y}$ that are normalized to unit length is measured by $\cos\alpha$ of the angle α between these. Here $\mathbf{x}$ and $\mathbf{y}$ have three dimensions i, j and k whereby each dimension has positive values only, e.g., as for occurrences frequencies.

Note, that in this case the cosine *similarity* induces the same ranking as the L_2-*distance* as $\sum_i(\mathbf{x}_i - \mathbf{y}_i)^2 = \sum_i \mathbf{x}_i^2 - 2\sum_i \mathbf{x}_i\mathbf{y}_i + \sum_i \mathbf{y}_i^2 = 2 - 2sim_{cos}(\mathbf{x}, \mathbf{y})$.

However, if an image description is used that does not directly build on visual words, other similarity measures may also be used. In fact, while all methods in this thesis build on visual words and the bag-of-words scheme, quite different similarity measures are used: In Chapter 4 we use the L_1-distance to determine the most similar images with respect to their topic distributions. In Chapter 5 we demonstrate a method that detects the appearance of objects in images by counting the occurrence of distinguished visual feature triples. And finally, in Chapter 7 we present a min-hash based similarity search approximating the Jaccard similarity between the corresponding set of visual words. The final ranking is then obtained by the Cosine similarity.

2.3.3 Weighting

The entries in the bag-of-word histogram are usually weighted to improve the retrieval: A *local* weight $w_{i,j}^{local}$ describes the importance of a particular visual word i for a particular image I_j. A *global* weight w_i^{global} reflects the overall importance and distinctiveness of the visual word i measured across a training set or the whole database. Finally a normalization factor n_j brings the bag-of-words vector to unit length. The weights for the visual word i in the bag-of-words vector $\mathbf{x}$ are combined as:

$$\mathbf{x}_i = w_{i,j}^{local} w_i^{global} n_j \tag{2.10}$$

The most prominent example is the use of the tf-idf weighting scheme

$$\mathbf{x}_i = tf_{ij} idf_i \ , \tag{2.11}$$

where tf_{ij} describes the *term-frequency* i.e. the L_1-normalized occurrence frequency of the visual word i in image I_j. The *inverse document frequency* idf_i reflects the global importance of the visual word i as the inverse of its appearance across multiple documents measure by $log(\frac{N}{n_i})$. Here, N denotes the total number of images in the database while n_i denotes the number of images that contain the visual word i. Note, that the actual basis of the logarithm does not matter as logarithms to different bases only differ by a constant factor that is canceled out by the normalization. The normalization factor is given by the inverse of the L_2-norm and implicitly contained in the cosine similarity measure itself (see Equation 2.9).

Numerous different weighting schemes have been proposed that incorporate the informativeness of visual words (Chisholm and Kolda 1999, Sivic and Zisserman 2003, Jégou et al. 2009a, Cour et al. 2011, Zheng et al. 2013). Recently, several approaches improve retrieval by (down)-weighting repeated visual words resulting from repetitive structures in images such as building facades or fences (Schindler et al. 2007, Jégou et al. 2009b, Torii et al. 2013). A comprehensive evaluation of weighting and normalization schemes can be found in Tirilly et al. (2009).

"In god we trust. All others bring data."

Attributed to William Edwards Deming (1900-1993)

3 Datasets and Evaluation Criteria

3.1 Datasets

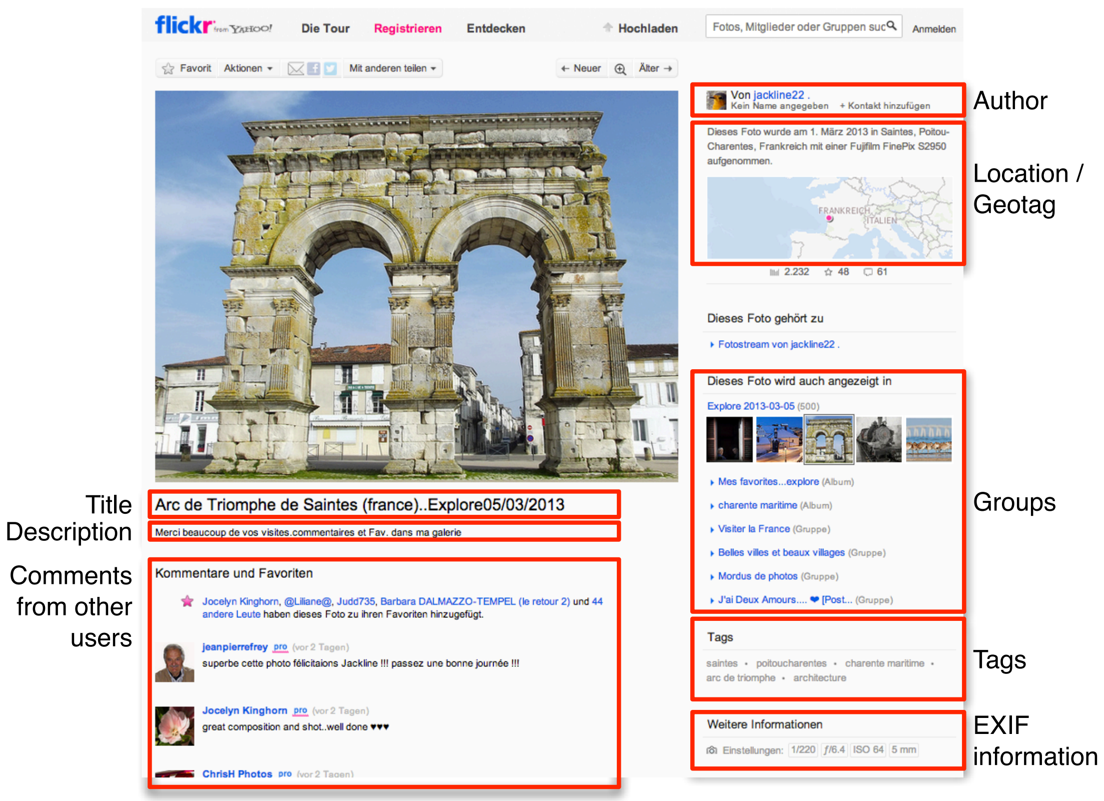
Throughout this work all methods were evaluated on publicly available datasets. Several different datasets were used and while each dataset is meant for a distinct purpose most of them have been gathered by downloading images from Flickr. All of these datasets feature real-world images taken in an uncontrolled environment with a variety of different camera models.

3.1.1 Flickr

Beginning roughly in 2005 highly interactive websites appeared - the so-called Web 2.0 - which quickly led to large community-driven websites and social networks. Soon users began to upload their private photos and videos in order to share them with friends, family and other users. Researchers started investigating in large-scale learning and data mining problems exploiting the community-generated content. This trend continues – nowadays large-scale and web-scale problems are often vaguely summarized with the buzzword "big data".

Flickr[1] was once the world's largest public image sharing website (with 6 billion images in August 2011[2]). While nowadays Facebook[3] far exceeds Flickr's size (with 220 billion images in October 2012[4]) Flickr is still a popular source for crowd-sourced image datasets.

[1] http://www.flickr.com
[2] http://blog.flickr.net/en/2011/08/04/6000000000/
[3] http://www.facebook.com
[4] http://gigaom.com/2012/10/17/facebook-has-220-billion-of-your-photos-to-put-on-ice/



Von jackline22 .
Kein Name angegeben + Kontakt hinzufügen

Dieses Foto wurde am 1. März 2013 in Saintes, Poitou-Charentes, Frankreich mit einer Fujifilm FinePix S2950 aufgenommen.

Dieses Foto gehört zu
- Fotostream von jackline22 .

Dieses Foto wird auch angezeigt in
Explore 2013-03-05 (500)
- Mes favorites...explore (Album)
- charente maritime (Album)
- Visiter la France (Gruppe)
- Belles villes et beaux villages (Gruppe)
- Mordus de photos (Gruppe)
- J'ai Deux Amours.... ❤ [Post... (Gruppe)

Tags
saintes • poitoucharentes • charente maritime • arc de triomphe • architecture

Weitere Informationen
Einstellungen: 1/220 ƒ/6.4 ISO 64 5 mm

Arc de Triomphe de Saintes (france)..Explore05/03/2013

Merci beaucoup de vos visites.commentaires et Fav. dans ma galerie

Kommentare und Favoriten

Jocelyn Kinghorn, @Liliane@, Judd735, Barbara DALMAZZO-TEMPEL (le retour 2) und 44 andere Leute haben dieses Foto zu ihren Favoriten hinzugefügt.

jeanpierrefrey pro (vor 2 Tagen)
superbe cette photo félicitaions Jackline !!! passez une bonne journée !!!

Jocelyn Kinghorn pro (vor 2 Tagen)
great composition and shot..well done ♥♥♥

ChrisH Photos pro (vor 2 Tagen)

Figure 3.1: A screenshot of the Flickr user interface as of March 2013. Areas that display user-generated metadata or social relationships are highlighted.

For the first time these social platforms provided the users a convenient interface to manage their private photos. Consequently, millions of users created photo collections, added descriptions and tags to their photos and associated them with geographic locations. Figure 3.1 shows the user interface of Flickr's platform available to end-users. The ease of use and the large number of active users are probably the main reason why community websites such as Flickr became popular as image source.

Researchers exploited that Flickr not only provides access to the images themselves but also to the associated metadata. This includes image descriptions, titles, relationships to other users and user groups, EXIF information and much more. Especially the tags that users assign to images in order to briefly describe their content received much attention: Many researchers treat such images as weakly labeled images, e.g., in weakly-supervised learning approaches.

In this work two large-scale datasets were used: *Flickr-250K* and *Flickr-10M*. Starting with the first dataset Flickr-250K we subsequently continued to increase size and diversity of our datasets to demonstrate the scalability of our methods, eventually ending up with 10 million images up to now.

Landmarks	Scenes	Objects
Abu Simbel, Angel falls, Arc de Triomphe, Church of Saviour Blood, Ayers Rock, Basilica de Notre Dame, Bilbao Guggenheim Museum, Big Ben ...	Beach, Carnival, Christmas, City, Desert, Forest, Portrait, Street, Sunset, Wedding	Aircraft, Bicycle, Bird, Boat, Bottle, Building, Bus, Butterfly, Car, Cat, Chair, Cow, Dog, Fish, Flower, Horse, ...
Activities	National Parks	Stars
Aikido, Archery, Ax throwing, Badminton, Ballett, Baseball, Basketball, Belly dance, Billiards, BMX, Bowling, Boxing, ...	Abel Tasman, Acadia, Addo Elephant, Algonquin, Ayuittuq, Bandhavgarh, Banff, Bromo Tenger, Cuc Phuong, Gran Paradiso, ...	Alice Cooper, Angelina Jolie, Audrey Hepburn, Barack Obama, Bill Clinton, Bill Gates, Brad Pitt, Britney Spears, Bruce Willis, ...
Total number of images (without duplicates)		10,080,251

Table 3.1: Example categories in Flickr-10M. The full list is available online[1].

3.1.1.1 Flickr-250K

The *Flickr-250K* dataset consists of 246,348 images. Each image is geo-tagged and associated with at least a single tag. By grouping related tags into categories such as “dogs”, “flowers”, etc. each image belongs to at least one of 12 different categories. A detailed description can be found in Hörster (2009). The database has not been cleaned or post-processed.

3.1.1.2 Flickr-10M

Continuing the idea of sub-sampling a real image database by a smaller stochastically equivalent image dataset we have further created another publicly available dataset called “Flickr-10M”[1] to evaluate the proposed retrieval methodology on a large real-world image database. This dataset consists of 10 million images downloaded from Flickr. This size of the dataset is beyond most datasets targeting a specific domain like scenes (e.g., SUN database (Xiao et al. 2010)), objects (e.g., PASCAL VOC (Everingham et al. 2009)), or landmarks (e.g., Oxford (Philbin et al. 2007)). As of 2013 its size is only comparable to Imagenet (Deng et al. 2009) and TinyImages (Torralba et al. 2008) and orders of magnitudes larger than most other datasets used for image retrieval evaluations.

We aimed to make this dataset as diverse as possible to allow the evaluation of greatly varying retrieval approaches. Therefore we collected images that were annotated with specific tags, which indicate a variety of landmarks, scenes, cities, stars as well as objects. Geotags were explicitly not used to download images for two reasons: In most cases the number of images that actually have been geo-tagged is very small even for popular landmarks. Furthermore many landmarks are photographed from the far distance. In that case the geo-tagged location may be far from the position of the landmark itself. Also, for many categories like cities or national parks geotags are relatively meaningless despite narrowing down the number of available images. Therefore we focused on tags and image descriptions. In cases a certain category did not yield

[1]The dataset is available at http://www.multimedia-computing.de/wiki/Flickr-10M

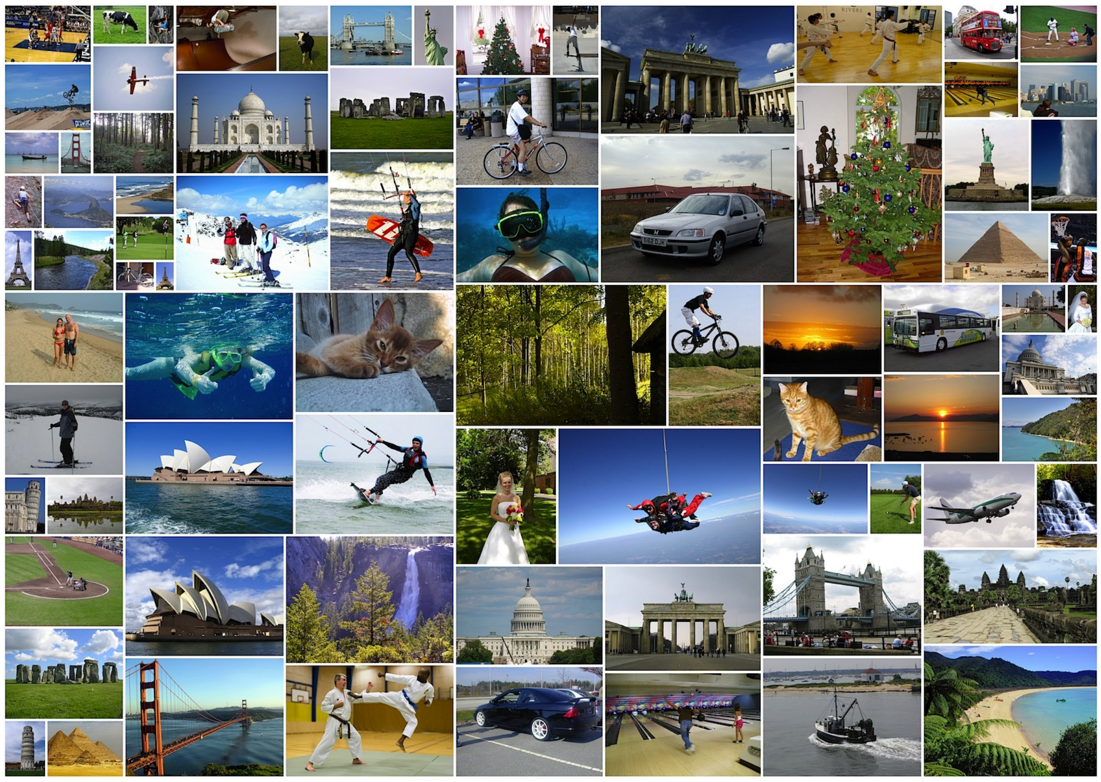


Figure 3.2: The 80 queries of the Flickr-10M dataset. The object or concept of interest are clearly depicted. The queries of the Flickr-250K dataset were selected in similar manner.

a sufficient number of images (e.g., several thousands) we performed a full-text search for the query term in the image description to select the downloaded images.

This dataset consists of JPEG images with their associated metadata. This includes tags, titles, descriptions, and other user-generated content as well as other information stored with the photos (e.g., EXIF data if available). There are 852,697 different Flickr users that contribute at least one photo to our dataset. In total there are more than 300 different categories yielding a total of 10,080,251 images. The 80 query images we used for evaluation on this dataset are show in Figure 3.2. These evaluate the retrieval with different objects, scenes or concepts. As for previously mentioned datasets, this datasets has not been cleaned or post-processed. Thus, it includes all kinds of content, e.g., from high-quality to low-quality photographs with and without annotations in all kinds of languages. In short, we believe this database is a representative sample of the real data that is uploaded and shared on community websites and social networks on a daily basis.

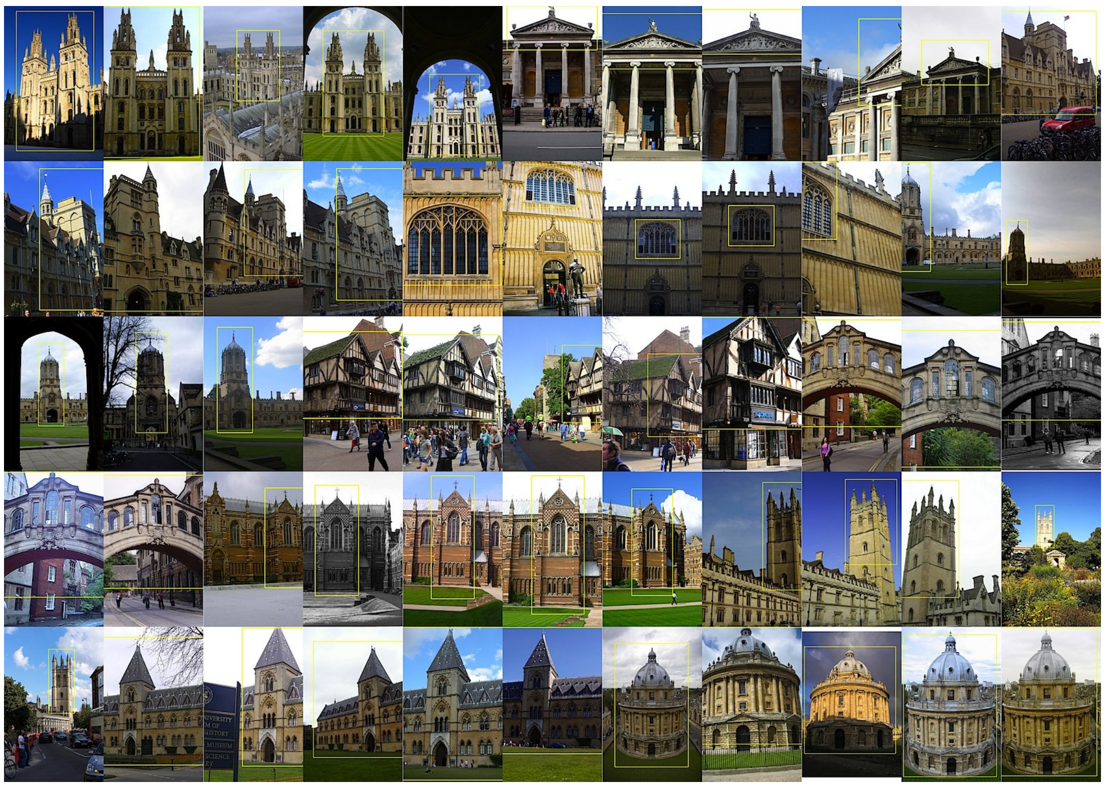


Figure 3.3: The 55 query images of the Oxford5K dataset. The yellow rectangle denotes the region of interest used to query an image database.

3.1.2 Oxford Buildings

Where appropriate we tested our approaches on the Oxford buildings dataset (Philbin et al. 2007). This dataset contains 5063 images of 11 buildings from Oxford as well as various distractor images. It is known for its difficulty to discriminate very similar building facades from each other and is one of the most well-known datasets for image retrieval.

The original dataset is often termed “Oxford5K”. To demonstrate the scalability of a certain retrieval system a common practice is to increase the database size by adding unrelated images – often called “distractor images” – downloaded from Flickr, which are unlikely to show the same object. The enriched database is often then termed according to its size “Oxford105K” and “Oxford1M”, meaning that 100,000 and 1,000,000 Flickr images have been added.

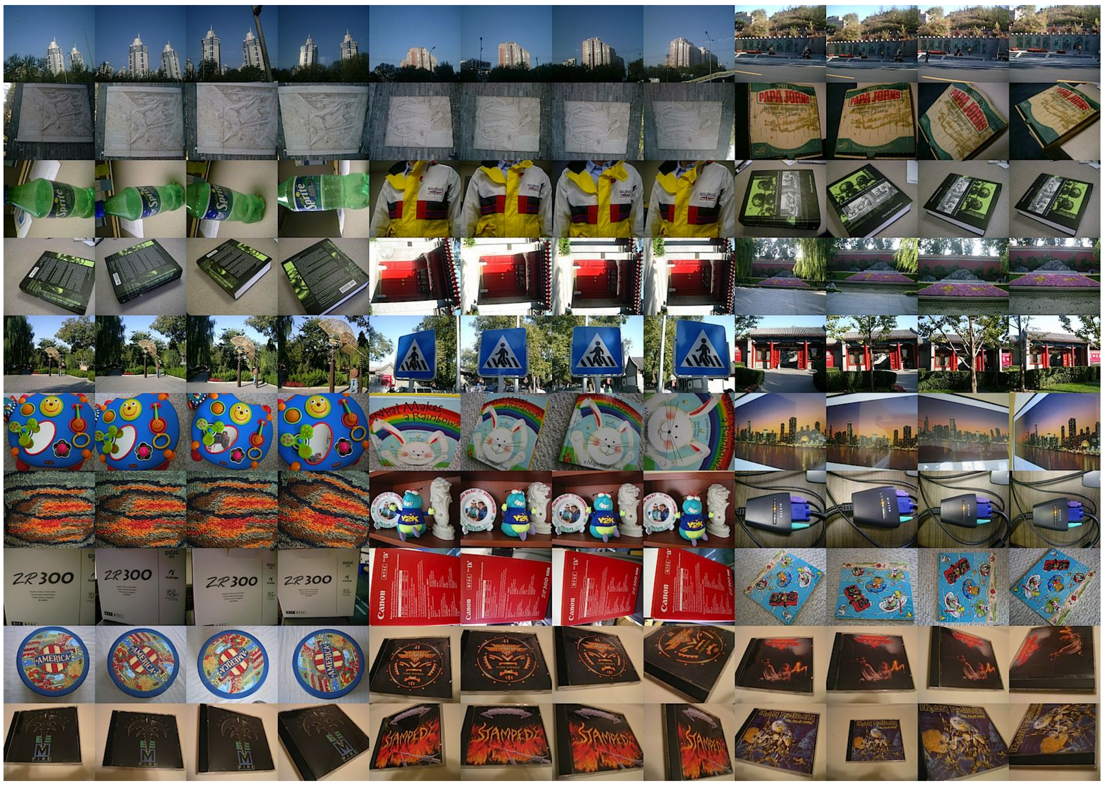


Figure 3.4: Examples of the images in the UkBench dataset. The datasets consists of groups of four near-duplicate images showing both scenes, objects and CD covers.

3.1.3 UkBench

Nister et al. published a dataset of 10,200 images called UkBench targeting specifically the evaluation of near-duplicate image detection and retrieval systems (Nistér and Stewénius 2006). The images are arranged in groups of four images with a resolution of 640x480 pixels. The images depict scenes, objects and a variety of CD covers whereby the objects are roughly centered in the image. For evaluation all images are indexed and each image is used as query once. As there are four images to be retrieved the performance is measured by the average top 4 score (see Section 3.2.5) counting the number of correctly retrieved images of each group among the top 4 retrieved results.

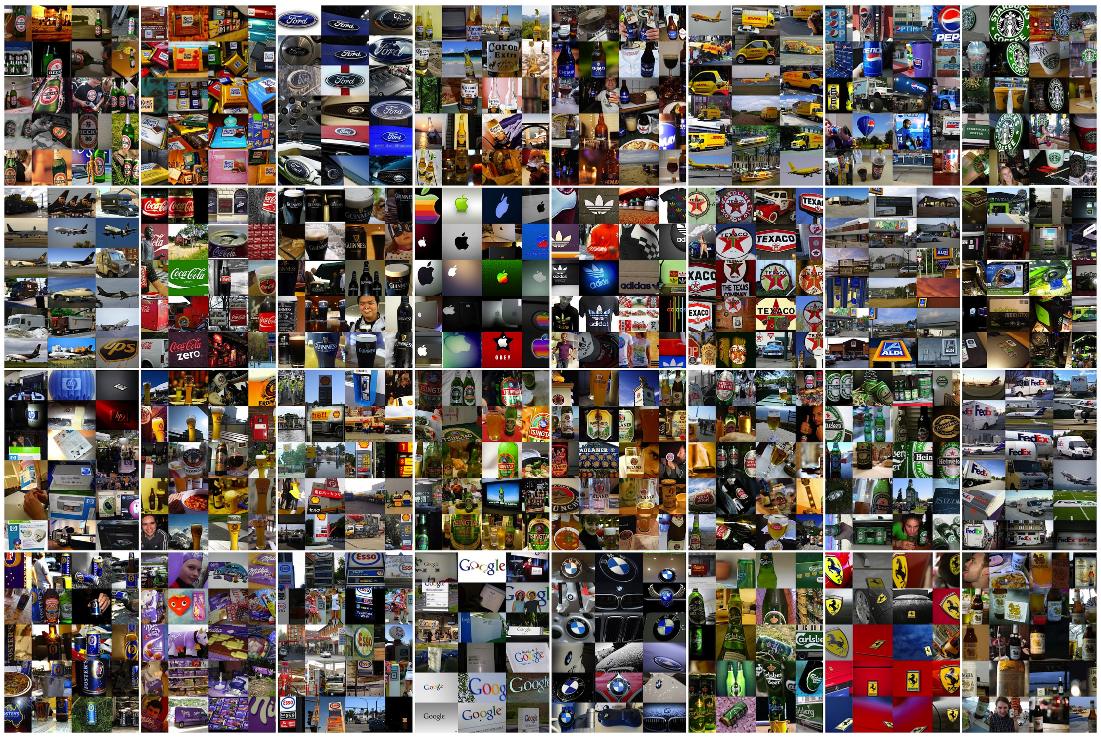


Figure 3.5: Example images of the test set (all 32 brand classes) of the FlickrLogos-32 dataset.

3.1.4 FlickrLogos-32

For the evaluation of methods targeting object retrieval and in particular logo retrieval or logo detection we built a new dataset called *FlickrLogos-32* containing photos depicting logos (Romberg et al. 2011).[1] We collected logos of 32 different brands by downloading them from Flickr. The specific brands were roughly those where we could retrieve images showing the correct logo by querying the Flickr web service with appropriate queries. We only included logos that have an approximately (piece-wise) planar surface.

There are 32 logo classes: Adidas, Aldi, Apple, Becks, BMW, Carlsberg, Chimay, Coca-Cola, Corona, DHL, Esso, Erdinger, Fedex, Ferrari, Ford, Foster's, Google, Guiness, Heineken, HP, Milka, Nvidia, Paulaner, Pepsi, Ritter Sport, Shell, Singha, Starbucks, Stella Artois, Texaco, Tsingtao and UPS. We manually inspected the retrieved images to ensure that the specific logo is actually shown. The whole dataset is split into three disjoint subsets P_1, P_2, and P_3. Each subset contains images of all 32 brands. The first partition P_1 consists of 10 images of each brand that were manually selected such that these consistently show a single logo from various views with little background clutter. The other two partitions P_2 and P_3 contain 30 images per brand. Unlike P_1 these images may contain more than one instance of a logo. In total both P_2

[1]The dataset and supplementary material is available at http://www.multimedia-computing.de/flickrlogos

Subset	*Description*	*Images*	*Sum*
P_1	Hand-picked images, single logo, clean background	10 per class	320
P_2	Images showing at least a single logo under various views Logo-free images	30 per class 3000	3960
P_3	Images showing at least a single logo under various views Logo-free images	30 per class 3000	3960

Table 3.2: Disjoint subsets of our dataset

and P_3 have 960 images showing logos. The subset P_3 for all 32 classes is shown in Figure 3.5.

For high-precision classifiers the evaluation of their sensitivity on non-logo images is very important. Therefore both partitions P_2 and P_3 include another 3000 images downloaded from Flickr with the queries “building", “nature", “people" and “friends". These images are unlikely to contain one of the 32 brand logos and are considered as negatives i.e., “logo-free” images. A brief summary of the data subsets is shown in Table 3.2.

This dataset targets both retrieval and classification methods. It features logos, which can be considered as rigid objects with a (roughly) planar surface visible from a single viewpoint only. The difficulty arises from the great variance of object sizes, from tiny logos in the background to image-filling views. Other challenges are perspective tilt and for classification eventually the task of multi-class recognition.

Compared to other well-known datasets suited for image retrieval, e.g., Oxford buildings, images of a similar class in FlickrLogos-32 share much smaller visually similar regions. For instance, the average object size of the 55 query images (annotated in the ground truth) of the Oxford dataset is 38% of the total area of the image (median: 28%) while the average object size in the test set of the FlickrLogos dataset is only 9% (median: 5%) of the whole image. As the retrieval of the Oxford buildings is sometimes coined “object retrieval”, the retrieval task on the FlickrLogos dataset can be considered as “small object retrieval”.

3.2 Performance measures

This section discusses the performance measures relevant in the context of this thesis.

3.2.1 Precision & Recall

The *precision (P)* measures how precise a retrieval system is, i.e., how many of the returned images actually show the object being searched for. In contrast, the *recall (R)* describes how many of the relevant objects in the database are actually retrieved when querying the retrieval system with the appropriate query. Precision and recall are defined as follows:

$$P = \frac{TP}{TP + FP} \tag{3.1}$$

$$R = \frac{TP}{TP + FN} \tag{3.2}$$

Here, *TP* denotes the *true positives* i.e., the number of correctly retrieved images that belong to the same class as the query image. *FP* denotes the *false positives* denoting the number of returned images that do not show the object of interest; i.e., these do not belong to the same class as the query image. Finally, *FN* denotes the *false negatives* – the total number of images of the same class as the query that were missed by the retrieval.

Both precision and recall require a ground truth of the database. In cases where it is not possible to define a ground truth, e.g., for large image collections, these scores are not applicable. However, the precision may be evaluated for the top N images (sometimes termed *Precision@N*) with manual judgments. This is described in more detail in Section 3.3.

3.2.2 Mean Precision & Mean Recall

In order to report a single score describing the performance of a retrieval system measured across multiple queries often the *mean precision (mP)* and *mean recall (mR)* are used. Note that these scores *cannot* be obtained by averaging precision and recall over multiple queries.

To illustrate this, consider the following example. For half of N queries a retrieval system returns all the corresponding relevant images in the database and no other images. For the remaining half of the queries the retrieval system returns no images at all. Computing the mean precision naively by averaging over the individual precision P_i of each query i as $\frac{1}{N}\sum_i^N P_i$ yields an incorrect mean precision of 0.5.

Instead, the mean precision must be computed similar to Equation 3.1 whereby the true and false positives are summed over all N queries:

$$mP = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} TP_i + \sum_{i=1}^{N} FP_i} \tag{3.3}$$

$$mR = \frac{\sum\limits_{i=1}^{N} TP_i}{\sum\limits_{i=1}^{N} TP_i + \sum\limits_{i=1}^{N} FN_i} \tag{3.4}$$

Similar, the mean recall is computed analogously as in Equation 3.2 by summing over the true positives and false negatives of all queries. In the example above the correct mean precision is 1 as the retrieval system only returned correct images and its mean recall is 0.5.

3.2.3 Average Precision (AP)

The trade-off between precision and recall is well-known, i.e., often high-precision implies low recall and vice versa. One drawback of those measures is that neither precision nor recall do take the ranking of a retrieval system into account. For instance, if two different result lists contain the same items, precision and recall of these are identical disregarding the actual ranking and position of each individual item in the result list.

Therefore the retrieval performance is often measured by a score that incorporates both precision and recall as well as the actual position of retrieval results in the ranking list. Examples of such scores that (partly) address this issue are the *F-measure*, *Discounted cumulative gain (DCG)* and *normalized DCG (NDCG)*. However, for image retrieval, the *average precision (AP)* is the most well-known and most frequently used measure.

The AP is defined as the area under the precision recall curve. It characterizes both precision and recall; a system will only gain high AP scores if both are high. Unfortunately, there are multiple definitions of this measure used in the literature. Depending on the definition and its implementation the AP varies greatly.

Its continuous form that integrates the precision at recall level $P(r)$ over all recall levels r

$$AP = \int_0^1 P(r) \cdot dr \tag{3.5}$$

must be transformed into its discrete formulation to cope with discrete rankings. The discrete formulation is given as

$$AP = \sum_{i=1}^{N} P_i \cdot \Delta R_i = \sum_{i=1}^{N} P_i \cdot (R_i - R_{i-1}), \; with \; R_0 = 0 \tag{3.6}$$

where P_i is the precision for the top i results (P@i) and ΔR_i describes the increase in recall from position $i-1$ to i. This solution considers all recall levels by summing from $i = 1...N$. In fact this is more elaborate than summing over a limited number of fixed recall levels, e.g., by selecting the 11 recall levels $0, 0.1, 0.2, ..., 1.0$.

However Equation 3.6 is still an approximation and even worse, it consistently overestimates the AP. This can be shown with a toy example as given by Table 3.3. A list of retrieval

	Position									
Position	1	2	3	4	5	6	7	8	9	10
Class	⊕	⊖	⊖	⊕	⊖	⊖	⊖	⊕	⊖	⊖

Table 3.3: Toy example of a retrieval result. The results were sorted after an arbitrary similarity score (not shown) yielding a position per item. ⊕ denotes items that are true positives with respect to the query, ⊖ denotes false positives.

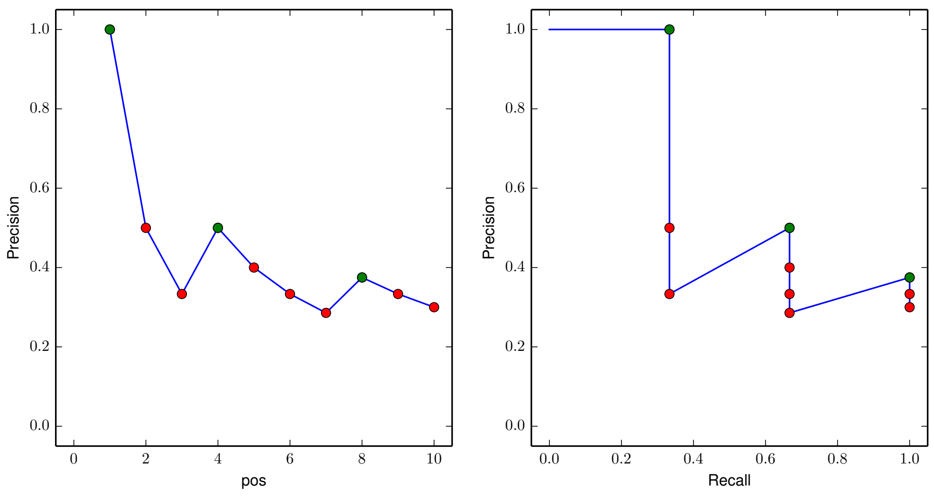


Figure 3.6: A toy example. **Left:** The precision of a fictive result set with positive examples (green dots) and negative examples (red dots) at the respective position *pos* within the result list. **Right:** The corresponding precision-recall curve.

results contains true positives (⊕) and false positives (⊖). For simplicity we assume the whole database contains a total of 3 true positives and all of them were retrieved. The corresponding precision-position (*pos*) and precision-recall curves for this example are shown in Figure 3.6.

To compute the AP this result list is processed starting from position 1. Following Equation 3.6 the AP is computed by summing over the precision at those points in the result list that are positive results, because otherwise $\Delta R_i = 0$. This results in an overestimation of the area-under-curve as shown in Figure 3.7 on the left.

To overcome this problem and to obtain the exact AP, not only the current precision P_i but also the precision at the previous position P_{i-1} must be considered, when processing the list of items at each position i.

$$AP = \sum_{i=1}^{N} \frac{1}{2}(P_i + P_{i-1}) \cdot (R_i - R_{i-1}),\ with\ R_0 = 0,\ P_0 = 1 \tag{3.7}$$

Again, ΔR_i describes the increase in recall from position $i-1$ to i. Analogously, ΔP_i describes the increase/decrease in precision from position $i-1$ to i. Equation 3.7 represents the numerical integration of Equation 3.5 with the trapezoidal rule. It leads to the correct AP

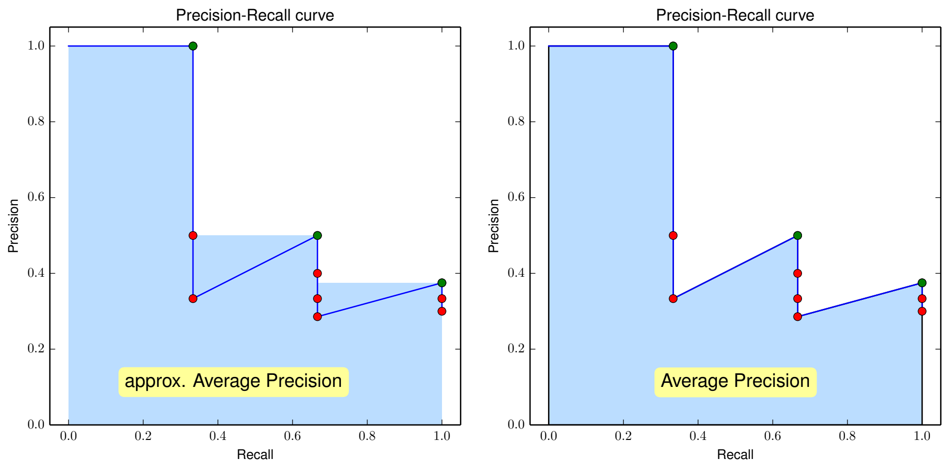


Figure 3.7: Left: Overestimation of the AP as computed by Equation 3.6. **Right:** Correct computation of the AP as by Equation 3.7.

as it exactly describes the area-under-curve as shown in Figure 3.7 on the right. Throughout this work we computed the AP always by this definition.

3.2.4 mean Average Precision (mAP)

The AP measures the area-under-curve for a single list of retrieval or classification results. To obtain a single score representing the performance of the whole system the AP is usually averaged over multiple queries. Note that unlike in the case of mean precision, the averaging of the AP is valid. It represents the mean performance of a system (in terms of AP) whereby every query has the same impact on final score.

3.2.5 Average Top 4 score (Top4)

Nistér and Stewénius (2006) introduced the *average top* 4 score for the evaluation with their dataset UkBench as it consists of groups of four near-duplicate images (see Section 3.1.3). The retrieval is measured as the average number of correctly retrieved images among the top 4 results. Hereby, the query image itself is counted as true positive if retrieved among the top 4. A perfect retrieval would retrieve 4 correct top-ranked images and therefore yield a score of 4.0.

3.2.6 Response Ratio (RR)

The *response ratio (RR)* is used to measure the efficiency of the retrieval. It describes the number of retrieved images in relation to the database size. The higher the response ratio the more images are contained in the result set, which is usually post-processed or verified by computationally expensive methods. A low response ratio will therefore increase the overall efficiency of the search.

For two retrieval systems with similar mAP, the one with lower response ratio is preferable. The response ratio is related to mAP, precision and recall in the sense that a system with a low response ratio must have high precision to retrieve the correct images in order to also have high recall and therefore a high mAP.

3.3 Evaluation on Large-scale Datasets with User Studies

All of the scores discussed so far require the presence of a ground truth. In cases where it is not possible to define ground truth for a certain dataset these scores are not applicable. For instance, for large-scale datasets with millions of images downloaded from community websites it is usually not feasible to create an annotation for every single image because of the required time and manpower. At the same time, semi-automatic annotation e.g., with the help of computer vision techniques is also not desirable because of the danger that the resulting dataset is heavily biased towards certain techniques and might contain mostly "easy" images.

In this particular case we used the following methodology: A number of predefined queries are issued to the retrieval system returning the top k results. These top results are then presented to human users which in turn label the results as "correct" or "incorrect". This allows us to compute a precision up to the k-th result often denoted as *Precision@k* (P@k) which is then averaged over multiple queries and multiple users.

More precisely, the participants of a user study were asked to rate the top 19 results of each query image. The following scoring then yields a quantitative performance measure: An image considered being similar gets 1 point, an image considered as "somewhat" similar gets 0.5 points. All other images get 0 points. The mean Precision@19 is calculated for each user and the mean over all users' means yields the final score of the system being evaluated.

For this we have developed a graphical tool that allows to perform user studies where human non-experts can label and rate the retrieval results. A screenshot of this tool is shown in Figure 3.8. We collect and process the obtained ratings in order to compute the final score.

We have strived to offer the participants an easy and convenient tool but nevertheless these user studies are tedious. There are various reasons; the most obvious is the plain number of experiments and the number of query images. In order to get results with reasonable efforts of the participants that are meaningful and significant we had to make the following design choices for the setup of all experiments:

1. The query images are not chosen randomly but carefully selected. This is done to avoid "junk" query images where no object or any other visual concept is depicted clearly. Judgments for retrieval results of those queries would hardly be consistent and useful. Examples of such query images are shown in Figure 3.2.

2. The different queries are chosen such that different visual concepts are depicted, especially those that were used to build the dataset. As there is no ground truth we do not

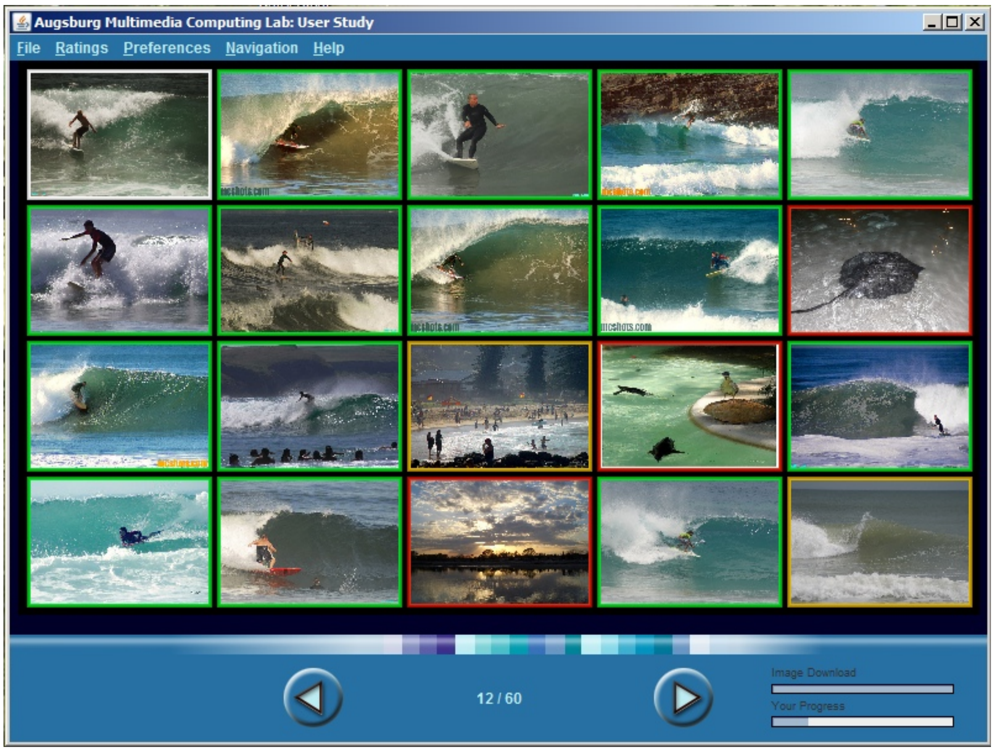


Figure 3.8: Graphical user interface used for evaluation of the retrieval results in user studies. The pre-defined query image is shown at the top-left (white frame). The other images show the top 19 retrieval results. The user may label each retrieved image as “correct” (green frame), “somewhat correct” (yellow frame) or “incorrect” (red frame). This process is repeated for multiple queries.

know exactly which concepts are covered by the images present in the dataset. Thus we manually select query images based on our knowledge how we built the dataset, where it is highly likely that there are conceptually similar images contained in the database. For instance, the dataset was created by downloading images having certain tags such as “car”, “cars”, etc. Thus, an image showing a car is a reasonable choice as query.

3. As we have a limited number of participants in our user studies only, the number of results is not sufficient to cancel out a potential bias introduced by the particular selection of query images. Therefore, the same query images are used for each experiment. Also, each user rates the results for all of these query images. In short, every user rates the same retrieval results for the same query images.

This set of rules leads to the score that is used as final performance measure for our retrieval systems. It is important to note that the actual value of the score is not meaningful, as it will vary once query images or the dataset are changed. But the relative difference between scores of different retrieval systems is a good indicator, which has the better overall performance.

Part II

Multimodal Image Retrieval

4
Multilayer Multimodal pLSA

Recently, community websites and social networks have become increasingly popular. These networks store a huge number of images and associated metadata. Due to the sheer amount of data, data management becomes non-trivial and the demand for effective yet memory-conservative retrieval methods is high.

In this chapter we present a retrieval technique suited for the context of large-scale community databases. In particular, we follow the query-by-example scenario: A user initiates the retrieval by selecting an image from the image database as query image – *without* supplying further information such as keywords. The retrieval engine can only exploit the information, which is already stored within the database – such as the image itself but also its associated metadata.

For that reason, we especially focus on fusing information from different domains such as visual features and textual annotations into a single compact and effective representation. We study our approach on two large databases of 250,000 and 10 million images.

More specifically, in the following the standard single-layer probabilistic Latent Semantic Analysis (pLSA) approach (Hofmann 1999; 2001) is extended to multiple layers. We name this approach *multilayer multimodal probabilistic Latent Semantic Analysis (mm-pLSA).* It captures the information from multiple modalities – such as visual features or textual annotations – within a single high-level description. As for unimodal topic models, each modality is modeled by the corresponding topic distribution. By aggregating the topic distributions of multiple modalities into a higher-level distribution we build a hierarchy of abstractions. Eventually, multiple modalities are described by a single description in a compact representation.

Parts of the work presented in this chapter have been described in Romberg et al. (2009), Lienhart et al. (2009) and Romberg et al. (2012). The remainder of this chapter is organized as follows. Section 4.2 surveys related work. In Section 4.3 we introduce the standard pLSA model. Then, Section 4.4 presents the multilayer multimodal pLSA. A heuristic for fast initialization of the mm-pLSA model is described in Section 4.5. In Section 4.6 we present the experiments, their setup and pre-processing as well as the results on the Flickr-250K dataset. In the same manner, the experiments regarding the Flickr-10M dataset, their setup, their results and a detailed discussion are given in Section 4.7. Finally, Section 4.8 concludes the chapter.

4.1 Introduction

Many image retrieval systems rely only on visual representation of the image content. However, nowadays images are often stored in and retrieved from large-scale community databases such as Flickr or Facebook. In many of those community databases the images are associated with different kinds of metadata, such as titles, descriptions, tags, comments and more. The aim is to exploit this additional information in order to improve the accuracy of image retrieval.

In this chapter we describe an approach suited particularly for community databases. Such databases allow photographers to label their images with keywords commonly termed tags. These tags reflect the user's personal view of an image and may express its content and context. As such tags are widely used even by non-professional photographers, we focus especially on exploiting these tags. The downside is that a large fraction of tags is usually either misleading, ambiguous or missing. Thus, using tags directly for retrieval is difficult, making more sophisticated models necessary.

We base our approach on *probabilistic Latent Semantic Analysis (pLSA)* models of Hofmann (2001), a popular topic model with moderate complexity and thus fast learning and inference. Topic models, originating from text retrieval, are one attempt to deal with data such as noisy and ambiguous words. Such topic models represent the actual words occurring in a document by a mixture of topics that generated these. Implicitly, topic models are thus able to deal with ambiguous words such as synonyms or sememes.

With that in mind, pLSA was adopted for the visual domain where textual words are replaced by their visual counterparts, the visual words. However, in most works the pLSA is applied to unimodal data only. One reason is that the early fusion of feature vectors in a multimodal setting often does not yield the desired improvement.

We propose a multilayer multimodal pLSA model (referred to as *mm-pLSA*) that can handle different modalities effectively and efficiently. The data of the different modalities is modeled by a hierarchy of topics, naturally representing the hierarchy of the underlying concepts. While our approach is applicable to multiple modalities and also hierarchies across multiple layers, we constrain ourselves to using the smallest possible non-degenerated mm-pLSA model: Two modalities are represented by two separate sets of (leaf-)topics. A set of top-level topics then merges the knowledge of the two sets of leaf-topics.

This approach somewhat resembles the computation of two independent pLSA models for each data modality separately. Their modality-specific topics are in turn merged by a single top-level pLSA model, and thus lends the proposed approach its name. In fact, we borrow this approach for a fast and strictly stepwise forward procedure to initialize the mm-pLSA model bottom-up. Following that scheme, we demonstrate that we obtain much better results for the mm-pLSA algorithm compared to random initialization.

4.2 Related Work

Topic models originate from text-based document retrieval. Here, documents are represented by a mixture of topics. This representation is much more compact than representing documents by the words themselves and also deals with synonyms and ambiguous words. Numerous models have been proposed, the pLSA of Hofmann (2001) and the Latent Dirichlet Allocation (Blei et al. 2003) are probably the most popular ones. In this work we focus on the former as it was widely used in the context of our work and both training and inference can be done efficiently.

The pLSA model was originally proposed for document retrieval (Hofmann 1999) and was later used for analyzing document corpora (Hofmann 2001). The pLSA but also other topic models were soon ported to the visual domain by replacing the textual words with visual words. Barnard et al. (2003) and Blei and Jordan (2003) as well as Monay and Gatica-Perez (2004) exploit topic models to model annotated image databases. Here, the learned models are used to infer the annotations given only the visual description.

Bosch et al. (2006) use the pLSA for scene classification where the topic distribution is used to decide to which category an image belongs. Fergus et al. (2005) learn pLSA models from training images retrieved by a text-based image search engine. The learned models are then used for object recognition in new unseen images. Inspired by their approach Li et al. (2006) use radial templates and topic models for object recognition. Lienhart and Slaney (2007) create a compact image representation by using the topic distributions as visual signatures for image search. Similarly, Hörster et al. (2007) apply Latent Dirichlet Allocation (LDA) while Greif et al. (2008) adopt the Correlated Topic Model (CTM) (Blei and Lafferty 2005) for the same application. Hörster et al. (2008a) study the impact of various local features and their parameters on image retrieval by topic models. Rodner and Denzler (2009) further propose to use an ensemble of pLSA models to diminish the impact of overfitting. In Ries et al. (2010) we show that the vocabulary size has less impact when used with topic models rather than when used directly as bag-of-words. A comprehensive review of different topic models in the context of image retrieval is given by Hörster (2009).

Our approach uses a hierarchical model with more than one topic layer. In a similar manner, Sivic et al. (2008) adapt the Hierarchical Latent Dirichlet Allocation model to the visual domain,

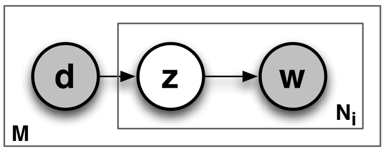


Figure 4.1: The graphical representation of the standard pLSA-Model in plate notation. The number of documents is denotes as M, while N_i denotes the number of words within document i. Observable variables are shaded in gray. The respective distribution $P(z|d)$ for the unobservable latent variable z is eventually used to represent images by a mixture of topics.

originally developed for unsupervised discovery of topic hierarchies in text. It is then used for object classification and image segmentation.

However, most previous works build their image representation solely from a single modality. In contrast, we aim specifically at incorporating information from two different sources or modalities into a single model. By exploiting the relationships, correlation and redundancy of two different modalities we aim to improve image representation. Similar, Zhang et al. (2011) propose a multi-feature pLSA. However, their approach uses a single topic layer that models the co-occurrence of visual features of two different types at once and is thus somewhat restricted to different kinds of visual features alone. Our approach is explicitly able to fuse totally different domains such as visual words and textual words into a combined representation. This chapter describes its direct usage for retrieval, but the representation of images by topic distributions has also been used for a multimodal similarity measure in the context of graph algorithms. In Richter et al. (2010; 2012) we build a graph of images linked by their multimodal similarities and apply the PageRank algorithm (Page et al. 1999) in order to obtain an estimate of the importance of each image.

4.3 Probabilistic Latent Semantic Analysis (pLSA)

The *probabilistic Latent Semantic Analysis (pLSA)* was originally devised by Hofmann (1999; 2001) in the context of text document retrieval, where words constitute the elementary parts of documents. Quite similar, in the visual domain images are represented by the visual words they contain. In the following we use the terms *document* and *image* as well as the terms *word* and *visual word* or *term* interchangeably.

4.3.1 Model

The key concept of the pLSA model is to map the high-dimensional word distribution vector of a document to a lower dimensional topic vector (or aspect vector). Therefore pLSA introduces a latent, that is, unobservable topic layer between the documents and the observed words. The model assumes that each document consists of a mixture of topics and that the actual occurrences of words are a result of the topic mixture. This generative model is expressed by

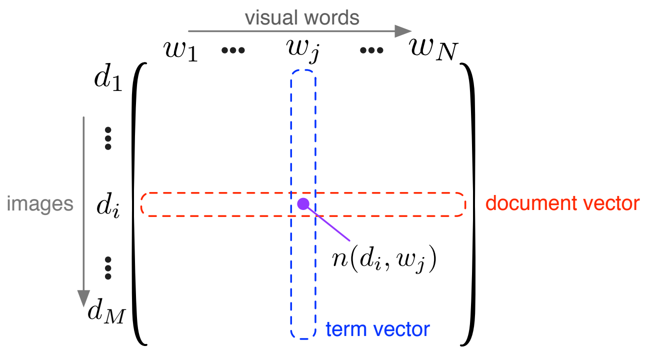


Figure 4.2: Term-document matrix. Each entry $n(d_i, w_j)$ denotes the number of occurrences of the word j in document i. Each row is equivalent to the bag-of-words histogram of document d_i.

the following probabilistic model:

$$P(d_i, w_j) = P(d_i) \sum_K P(z_k|d_i) P(w_j|z_k) \tag{4.1}$$

Here, $P(d_i)$ denotes the probability of a document d_i of the database to be picked, $P(z_k|d_i)$ the probability of a topic z_k given the current document, and $P(w_j|z_k)$ the probability of a visual word w_j given a topic. The graphical representation of this model is shown in Figure 4.1. Here, given M documents, N_i denotes the number of words of which document d_i consists.

The topic mixture $P(z_k|d_i)$ is used to represent each document d_i by a mixture of K topics. Naturally, the number of concepts or topics in the model is chosen to be much smaller than the vocabulary sizes. In other words, the representation by a distribution of topics instead of a distribution of words acts as data reduction yielding a compact representation. Furthermore, the topic mixture is a high-level representation as it decouples documents from the words that it contains. Documents are represented by the topics that are likely to generate the observed words. This representation deals with synonyms as it learns which words co-occur with each other and incorporates the statistical dependencies into the distribution $P(w_j|z_k)$. It also handles ambiguous words as a certain word may be generated by multiple topics.

The topic vector can be used directly for retrieval. For that we measure the document similarity by computing the distance between the topic vectors of the corresponding documents. An evaluation of different distance measures in this context was performed by Hörster (2009). It was shown that the L_1-distance – though not being the best metric – works reasonably well while being computationally efficient.

4.3.2 Training and Inference

Computing a *term-document matrix* of the training corpus is a prerequisite for deriving a pLSA model (see Figure 4.2). Each entry in row i and column j of the term-document matrix $[n(d_i, w_j)]_{i,j}$ specifies the absolute count with which word w_j (also called a term) occurs in document d_i. The terms are taken from a predefined dictionary consisting of N terms. The number of documents is M. Note that by normalizing each document vector to 1 using the L1-norm, the document vector $(n(d_i, w_1), ..., n(d_i, w_N))$ of d_i becomes the estimated mass probability distribution $P(w_j|d_i)$.

We learn the unobservable probability distributions $P(z_k|d_i)$ and $P(w_j|z_k)$ from the observable data $P(w_j|d_i)$ and $P(d_i)$ using the *Expectation-Maximization* algorithm (EM-Algorithm) (Dempster et al. 1977, Hofmann 2001). The update equations are as follows:

Expectation-Step:

$$P(z_k|d_i, w_j) = \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_{l=1}^{K} P(w_j|z_l)P(z_l|d_i)} \tag{4.2}$$

Maximization-Step:

$$P(w_j|z_k) = \frac{\sum_{i=1}^{M} n(d_i, w_j)P(z_k|d_i, w_j)}{\sum_{j=1}^{N}\sum_{i=1}^{M} n(d_i, w_j)P(z_k|d_i, w_j)} \tag{4.3}$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)}{n(d_i)} \tag{4.4}$$

Given a new test image d_{test}, we estimate the topic probabilities $P(z_k|d_{test})$ from the observed words. The sole difference between inference and learning is that the K learned conditional word distributions $P(w_j|z_k)$ are never updated during inference. Thus, only Equation (4.2) and Equation (4.4) are iteratively updated during inference.

4.4 Multilayer Multimodal pLSA

4.4.1 Motivation and Model

In recent years pLSA as well as other topic models have been applied successfully to unimodal data such as text (Hofmann 2001), image tags (Monay and Gatica-Perez 2004), or visual words (Hörster et al. 2008b) for different applications.

However, combining two modalities such as visual words and tags is challenging. The straightforward combination by concatenating the input vectors is one choice, but may not perform well. For instance, commonly there are very few tags per image but several hundred to a few thousand visual words. Also, the vocabulary sizes for visual and textual words may differ greatly. As a consequence, the resulting word histograms are heavily imbalanced and require at least normalization or other treatment. The direct combination of input vectors can be seen as an early fusion scheme. In contrast, our multimodal multilayer pLSA effectively acts as a late fusion method that deals with class imbalances by learning the appropriate distributions.

In the following we describe our multi-layered pLSA model for two leaf models and one top-level model fusing the former two lower-level models. In the following description, we assume that the first modality of our image representation are visual words and the second textual

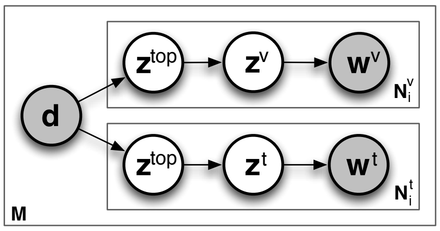


Figure 4.3: Plate notation of the multilayer multimodal pLSA mode combining two different modalities. Only the visual words w^v and the textual words w^t can be observed. The distribution of visual and textual words is represented in the lower layer by the separate topic distributions $P(w^v|z^v)$ and $P(w^t|z^t)$. The top-level topics z^{top} model the distribution over both lower level distributions – representing the distributions $P(z^v|z^{top})$ and $P(z^t|z^{top})$, merged as $\sum P(z^v|z^{top}) + \sum P(z^t|z^{top}) = 1$. The image representation is then given by $P(z^{top}|d)$.

word or tags. However, our model is not constrained to these modalities in any way and may be further extended to more than two modalities and also more than two layers.

The smallest possible multilayer multimodal pLSA model is depicted in Figure 4.3. It consists of two modes, the observed words of that mode and the corresponding distributions. Every word of mode x (here: $x \in \{v, t\}$ with v standing for *visual* and t for *text*) occurring in document d_i is generated by an unobservable document model:

- Pick a document d_i with prior probability $P(d_i)$
- For each visual word in the document:
  - Select a latent top-level concept z_l^{top} with probability $P(z_l^{top}|d_i)$
  - Select a visual topic z_k^v with probability $P(z_k^v|z_l^{top})$
  - Generate a visual word w_m^v with probability $P(w_m^v|z_k^v)$
- For each tag associated with the document:
  - Select a latent top-level concept z_l^{top} with probability $P(z_l^{top}|d_i)$
  - Select a tag topic z_p^t with probability$P(z_p^t|z_l^{top})$
  - Generate a tag w_n^t with probability $P(w_n^t|z_p^t)$

Thus, the probability of observing a visual word w_m^v in document d_i is

$$P(d_i, w_m^v) = \sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_l^{top}|d_i)P(z_k^v|z_l^{top})P(w_m^v|z_k^v). \tag{4.5}$$

Similar, the probability of observing the tag w_n^t in document d_i is given by

$$P(d_i, w_n^t) = \sum_{l=1}^{L}\sum_{p=1}^{P} P(d_i)P(z_l^{top}|d_i)P(z_p^t|z_l^{top})P(w_n^t|z_p^t). \tag{4.6}$$

To summarize, a mixture of topics in each mode describes an image or more specifically the occurrences of its visual and textual words. The topic distributions represent the distribution of the observed words over the topics. These low-level topics are in turn represented by the top-level topic distribution that describes the distribution of lower-level topics given the top-level topics. Representing an image by a mixture of topics seems natural, since images often contain multiple objects or a scene may be visually decomposed into different parts.

4.4.2 Training and Inference

Given our word generation model (see Figure 4.3) with its implicit independence assumption between generated words, the likelihood L of observing our database consisting of the observed pairs (d_i, w_m^v) and (d_i, w_n^t) from both modes is given by

$$L = \prod_{i=1}^{M} \left[\prod_{m=1}^{N^v} P(d_i, w_m^v)^{n(d_i, w_m^v)} \prod_{n=1}^{N^t} P(d_i, w_n^t)^{n(d_i, w_n^t)} \right] . \tag{4.7}$$

Taking the log to determine the log-likelihood l of the database

$$l = \sum_{i=1}^{M} \left[\sum_{m=1}^{N^v} n(d_i, w_m^v) \log P(d_i, w_m^v) + \sum_{n=1}^{N^t} n(d_i, w_n^t) \log P(d_i, w_n^t) \right] \tag{4.8}$$

and plugging Equation (4.5) and (4.6) into Equation (4.8), it becomes apparent that there is a double sum inside of both *log*s making direct maximization with respect to the unknown probability distributions difficult. Therefore, we learn the unobservable probabilities distribution $P(z_l^{top}|d_i)$, $P(z_k^v|z_l^{top})$, $P(z_p^t|z_l^{top})$, $P(w_m^v|z_k^v)$ and $P(w_n^t|z_p^t)$ by using the Expectation-Maximization (EM)-Algorithm (Dempster et al. 1977).
Introducing the indicator variables

$$\triangle c_{lk} = \begin{cases} 1 & \text{if the pair } (d_i, w_m^v) \text{ was generated by } z_l^{top} \text{ and } z_k^v \\ 0 & \text{otherwise} \end{cases}$$

$$\triangle d_{lp} = \begin{cases} 1 & \text{if the pair } (d_i, w_p^t) \text{ was generated by } z_l^{top} \text{ and } z_p^t \\ 0 & \text{otherwise} \end{cases}$$

the complete data likelihood L_c, that is the data likelihood assuming that d_i, w_n^z, w_m^v, $\triangle c_{lk}$, and $\triangle d_{lp}$ are observable, is given by

$$L_c = \prod_{i=1}^{M} [\prod_{m=1}^{N^v} P(d_i, w_m^v, \triangle c)^{n(d_i, w_m^v)} \prod_{n=1}^{N^t} P(d_i, w_n^t, \triangle d)^{n(d_i, w_n^t)}] \tag{4.9}$$

with

$$\triangle c = (\triangle c_{11}, ..., \triangle c_{1K}, ..., \triangle c_{LK}) \tag{4.10}$$

$$\triangle d = (\triangle d_{11}, ..., \triangle d_{1K}, ..., \triangle d_{LP}) \tag{4.11}$$

$$P(d_i, w_m^v, \triangle c) = \prod_{l=1}^{L}\prod_{k=1}^{K} P(d_i)P(z_l^{top}|d_i)P(z_k^v|z_l^{top})P(w_m^v|z_k^v)^{\triangle c_{lk}} \tag{4.12}$$

$$P(d_i, w_n^t, \triangle d) = \prod_{l=1}^{L}\prod_{p=1}^{P} P(d_i)P(z_l^{top}|d_i)P(z_p^t|z_l^{top})P(w_n^t|z_p^t)^{\triangle d_{lp}} \tag{4.13}$$

Unlike in Equation (4.8), we now only have product terms in the complete likelihood L_c, thus its log-likelihood can easily be determined and maximized. A complete derivation of the EM-update equations for this mm-pLSA model is given by Hörster (2009). The resulting expectation (E-step) and maximization (M-step) steps are given as follows:

Expectation Step: We estimate the unknown indicator variables $\triangle c_{lk}$ conditioned on the observable variables d_i and w_m^v by computing their expected value:

$$\begin{aligned} c_{lk}^{im} &:= E(\triangle c_{lk}|d_i, w_m^v) \\ &= P(\triangle c_{lk} = 1|d_i, w_m^v)\cdot 1 + P(\triangle c_{lk} = 0|d_i, w_m^v)\cdot 0 \\ &= P(\triangle c_{lk} = 1|d_i, w_m^v)\cdot 1 \\ &= \frac{P(d_i, w_m^v, \triangle c_{lk} = 1)}{P(d_i, w_m^v)} \\ &= \frac{P(d_i)P(z_l^{top}|d_i)P(z_k^v|z_l^{top})P(w_m^v|z_k^v)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_l^{top}|d_i)P(z_k^v|z_l^{top})P(w_m^v|z_k^v)}. \end{aligned} \tag{4.14}$$

Analogously we estimate the unknown indicator variables $\triangle d_{lp}$ conditioned on the observable variables d_i and w_n^t by computing their expected value:

$$\begin{aligned} d_{lp}^{in} &:= E(\triangle d_{lp}|d_i, w_n^t) \\ &= \frac{P(d_i)P(z_l^{top}|d_i)P(z_p^t|z_l^{top})P(w_n^t|z_p^t)}{\sum_{l=1}^{L}\sum_{k=1}^{K} P(d_i)P(z_l^{top}|d_i)P(z_p^t|z_l^{top})P(w_n^t|z_p^t)} \end{aligned} \tag{4.15}$$

Maximization Step: For legibility of the M-step estimates, we set

$$\gamma_{lk}^{im} := n(d_i, w_m^v)c_{lk}^{im} \tag{4.16}$$

$$\delta_{lp}^{in} := n(d_i, w_n^t)d_{lp}^{in}, \tag{4.17}$$

which is the expected probability of observing a pair (d_i, w_m^v) multiplied with the actual number of occurrences. Thus, we get:

$$P(d_i)^{new} = \frac{\sum_{m=1}^{N^v} n(d_i, w_m^v) + \sum_{n=1}^{N^t} n(d_i, w_n^t)}{\sum_{i=1}^{M}\left(\sum_{m=1}^{N^v} n(d_i, w_m^v) + \sum_{n=1}^{N^t} n(d_i, w_n^t)\right)} \tag{4.18}$$

$$P(z_l^{top}|d_i)^{new} = \frac{\sum_{m=1}^{N^v}\sum_{k=1}^{K}\gamma_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P}\delta_{lp}^{in}}{\sum_{l=1}^{L}\left(\sum_{m=1}^{N^v}\sum_{k=1}^{K}\gamma_{lk}^{im} + \sum_{n=1}^{N^t}\sum_{p=1}^{P}\delta_{lp}^{in}\right)} \tag{4.19}$$

$$P(z_k^v|z_l^{top})^{new} = \frac{\sum_{i=1}^{M}\sum_{m=1}^{N^v}\gamma_{lk}^{im}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v}\gamma_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t}\delta_{lp}^{in}} \tag{4.20}$$

$$P(z_p^t|z_l^{top})^{new} = \frac{\sum_{i=1}^{M}\sum_{n=1}^{N^t}\delta_{lp}^{in}}{\sum_{k=1}^{K}\sum_{i=1}^{M}\sum_{m=1}^{N^v}\gamma_{lk}^{im} + \sum_{p=1}^{P}\sum_{i=1}^{M}\sum_{n=1}^{N^t}\delta_{lp}^{in}} \tag{4.21}$$

$$P(w_m^v|z_k^v)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L}\gamma_{lk}^{im}}{\sum_{m=1}^{N^v}\sum_{i=1}^{M}\sum_{l=1}^{L}\gamma_{lk}^{im}} \tag{4.22}$$

$$P(w_n^t|z_p^t)^{new} = \frac{\sum_{i=1}^{M}\sum_{l=1}^{L}\delta_{lp}^{in}}{\sum_{n=1}^{N^t}\sum_{i=1}^{M}\sum_{l=1}^{L}\delta_{lp}^{in}} \tag{4.23}$$

Note that Equation (4.18) is constant across all iterations and needs not be recomputed.

Given a new test image d_{test}, we estimate the top-level aspect probabilities $P(z_l^{top}|d_{test})$ with the same E-step equations as for learning and Equation (4.19) for $P(z_l^{top}|d_{test})$ as the M-step. The probabilities of $P(z_k^v|z_l^{top})$, $P(z_p^t|z_l^{top})$, $P(w_m^v|z_k^v)$ and $P(w_n^t|z_p^t)$ have been learned from the corpus and are kept constant during inference.

4.4.3 Implementation Details

Normalization Before starting the mm-pLSA the document vectors of different modalities, i.e., the entries $n(d_i, w_m^v)$ and $n(d_i, w_n^t)$ should be normalized to equal scale. This is especially important if one modality has much higher occurrence frequencies of the respective words than the other modality. For instance, compare the highly populated histograms of visual features to the extremely sparse tag histograms. In such setting, the mm-pLSA on unnormalized histograms is dominated by the visual domain.

The simplest way to achieve a balancing of modalities, is to convert the absolute occurrence counts to relative frequencies:

$$n'(d_i, w_m^v) = \frac{n(d_i, w_m^v)}{\sum_m n(d_i, w_m^v)} \qquad n'(d_i, w_n^t) = \frac{n(d_i, w_n^t)}{\sum_n n(d_i, w_n^t)} \tag{4.24}$$

The sums over each modality are then of equal scale as $\sum_m n'(d_i, w_m^v) = \sum_n n'(d_i, w_n^t) = 1$.

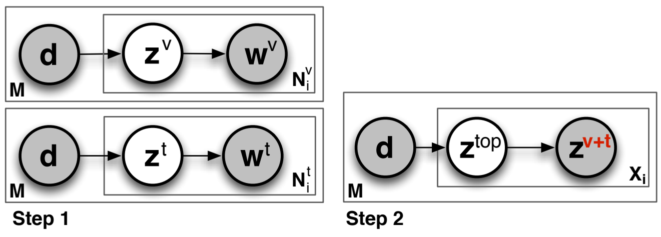


Figure 4.4: The fast initialization of the mm-pLSA model is computed in two separate steps. In step 1 two pLSA models are learned on each modality separately. In step 2 a third model is learned on top; the topic distribution of the former two serves as input for the third model.

Note that this normalization does *not* mean that the modalities have the same weight within the mm-pLSA as the constraint for the conditional probabilities of the subtopics given the supertopics is given by

$$\sum_{k=1}^{K} P(z_k^v|z_l^{top}) + \sum_{p=1}^{P} P(z_p^t|z_l^{top}) = 1. \tag{4.25}$$

In fact we noticed that the mm-pLSA on SIFT features and tags determines a higher weight for the textual domain. See the discussion in Section 4.7.5 for further details.

Training The training itself must only consider documents that have non-zero document vectors for both domains. With missing co-occurrences across the modalities the model training is useless. However, the inference is still able to derive a topic distribution even if one modality is not available for an image. This is important, as in practice images often lack annotations.

Furthermore the training procedure should sample training documents such that basically all visual and textual aspects that appear in the database are also present in the training set. However the number of images for a certain class or category may vary. Therefore we pseudo-randomly pick training samples by selecting documents at certain intervals from the whole list of documents starting at a random offset. This guarantees that the whole database is used when drawing samples disregarding the actual layout and order. Training documents of a certain category are drawn with a probability corresponding to its size.

4.5 Fast Initialization by a Step-wise Forward Procedure

More complicated probabilistic models always come with an explosion in required training time. This issue is becoming more severe, the more layers and the more individual models are aggregated into higher-level models. Therefore, we use the outcome of an approximate multimodal multilayer pLSA model produced by the stepwise forward procedures as input for our global optimization. It serves as a fast initialization for the estimates of the conditional probabilities.

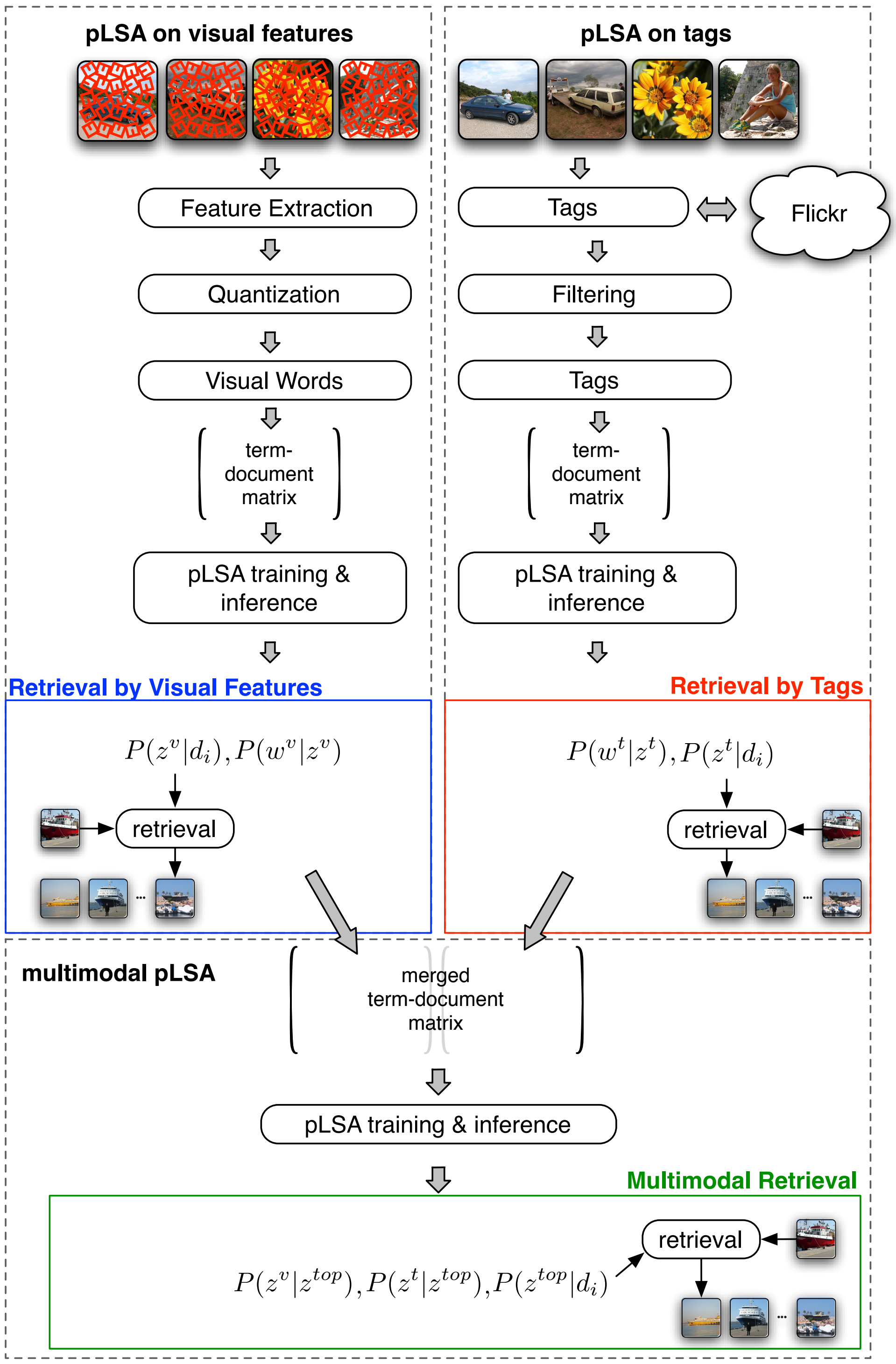


Figure 4.5: Schematic overview of the retrieval system based on our fast initialization strategy. The subsequent mm-pLSA then globally optimizes all three topic distributions jointly across all layers. Alternatively, the fast initialization scheme may be directly used for retrieval.

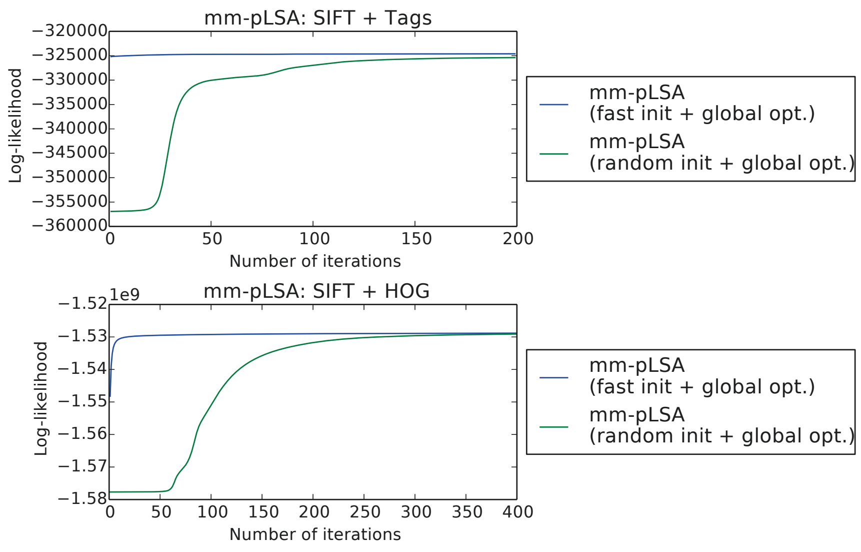


Figure 4.6: Log-likelihood over training data when learning the mm-pLSA model. The mm-pLSA initialized by the strictly stepwise forward multimodal pLSA converges much faster than the model starting from a random initialization. The upper image shows the log-likelihood when the mm-pLSA is applied for SIFT features and tags, the lower image shows the log-likelihood for SIFT and HOG block features.

The basic idea is to apply pLSA in a first step to each mode separately yielding mode-specific topic distributions. In the second step another pLSA model is learned on top. That is, the top-level pLSA uses the respective topic distributions of the former two base models as input, as if these were weighted occurrence frequencies.

For a hierarchy of two levels and two leaf modalities – such as visual features and text – this procedure first computes separate pLSA models for each modality. The topic distributions of these models are only implicitly linked as they originate from the same image (see Step 1 in Figure 4.4). Next the computed topics of all modes are taken as the observed words at the top level (see Step 2 in Figure 4.4). The final image representation is then given by the top-level topic distribution. It describes each image as a “topic distribution over topic distributions” and thereby fuses the visual pLSA model and the tag pLSA model. A full schematic overview of an image retrieval system based on this fast initialization strategy is shown in Figure 4.5.

This procedure can also be applied in a strictly forward procedure to multiple modalities and multiple layers. We expect, that with increasing complexity of the global model the fast initialization may become more important due to its computational efficiency.

As we will show in the experimental results, this fast initialization procedure already produces a decent model. It can be further improved by applying the globally optimizing EM-algorithm as stated in Section 4.4.2 to the complete model after initializing it with the strictly forward computed solution. Thus, as we initialize our model with constrained solutions – i.e., multiple partial solutions, each limited to a single modality – we expect (1) that the global optimization further improves the solution and (2) this procedure yields faster convergence than learning a mm-pLSA model starting from a random initialization.

To demonstrate this, Figure 4.6 shows the development of the complete data log-likelihood over the number of iterations when learning mm-pLSA models on the Flickr-10M dataset. Indeed, one can observe that the global optimization of the mm-pLSA model converges much faster when initialized with our fast initialization strategy (termed “fast init”) instead of random initialization (“random init”).

4.6 Evaluation on the Flickr-250K dataset

4.6.1 Visual pLSA-Model

We build a bag-of-words representation of images by extracting SIFT features from images. In a previous work Hörster et al. (2008a) showed that SIFT features computed from a dense regular grid outperform sparse SIFT features in the context of image retrieval via topic models. Thus, we extract dense SIFT features at points on a regular grid with a vertical and horizontal step size of 10 pixels. The image is incrementally down-scaled by a scale factor of $2^{\frac{1}{4}}$ and features are extracted at each scale of the resulting image pyramid. The image regions around keypoints are described by SIFT descriptors computed over a region of 41×41 pixels. Finally, the 128-dimensional feature vectors are quantized into discrete visual words. Hierarchical k-means clustering was used to create a visual vocabulary of 10,000 visual words. Efficient quantization was then performed by the corresponding vocabulary tree (see Section 2.2.2).

4.6.2 Tag-based pLSA-Model

The second modality we consider are tags. Tags are free-text annotations provided by the image authors or image owners. In the context of the Flickr community website, a tag can be a single word as well as a phrase or a sentence. While Flickr stores the original form of an annotation such as “Golden Gate Bridge” in three separate words, it also provides a generated raw tag like “goldengatebridge” that encodes the whole word combination. In this work we treat each of these generated raw tags as a single word, disregarding if it is a natural or an artificially generated word. In the following the term *tag* denotes a single word derived from the set of all raw tags and is used interchangeably with "word" and "term".

As we use Flickr images to evaluate our mm-pLSA model, it is important to note that these tags reflect the photographer’s personal view of the uploaded image. In previous work carefully annotated image databases were used for learning combined image and tag models (Barnard et al. 2003). In contrast, the image tags from Flickr are often subjective, ambiguous, and do not necessarily describe the image content shown (Kennedy et al. 2007, Lienhart and Slaney 2007). This makes it difficult to use the tags directly for retrieval purposes and thus some preprocessing

breakfast	eat food meal
dog	animal beast canine creature fauna mammal move object travel
flying	action air event motion move movement travel travelling
house	construction home object structure
love	emotion state
yellow	color colour

Table 4.1: Examples for hypernyms (right) found in Wordnet for the words left.

is required. Even worse, some images do not have tags at all. In fact about 13% of all Flickr images lack annotations[1]. In this case textual information is not available for retrieval and a fallback strategy is needed. This underlines the importance of using a multimodal approach when exploiting user-generated content for image retrieval.

In order to prepare the input for our topic model, we first need to reduce the virtually infinite number of different tags to a finite vocabulary of frequently occurring tags. For that we keep all tags that have been used more than T_{minOcc} times and by at least $T_{minUsers}$ different users. This heuristic discards all rarely used tags. Note that a tag is also neglected if only a few users have used it. We further filter the list by discarding all tags that contain numbers and by splitting tags at underscores into separate words. In a last filtering step all words within the vocabulary are checked whether they are known by Wordnet (Fellbaum 1998), a lexical database of English. Only words that exist according to Wordnet are gathered in the final vocabulary.

Once the tag vocabulary is defined, the term-document table is built by counting the tag occurrences for each image. To augment the existing annotations we expand the list of tags associated with an image by using Wordnet. For that, we query Wordnet for the semantic parents of the tags specified by the author. Semantic parents for a word can easily be extracted from Wordnet as each word in Wordnet is associated with hypernyms. Here, Y is a hypernym of X if every X is a (kind of) Y. A word's hypernyms denote its parents that express the specific concept of the tag more generally. Table 4.1 shows hypernyms (right) for several example tags (left). As these hypernyms build a hierarchy and form a tree structure, we add the hypernyms up to three levels above in the hierarchy of the tag itself to the tag list of the corresponding image. Thus, while counting tag occurrences for each image in order to build the document vector, these parents are included in our model by counting them as if they were present in the list of tags. In case the vocabulary does not contain a tag or the associated hypernyms of an annotation, the word is simply ignored. This procedure emphasizes semantic features rather than just plain word counts.

In our experiments we set the parameters for the tag vocabulary to $T_{minOcc} = 18$ and $T_{minUsers} = 10$ resulting in a vocabulary size of 2421 words. Most images in our database have between 5 and 15 tags (including hypernyms) as an analysis reveals (see Figure 4.7). The number of tags for some images is however unreasonably large as some users also tend to label

[1]Analysis was done on the Flickr-10M dataset.

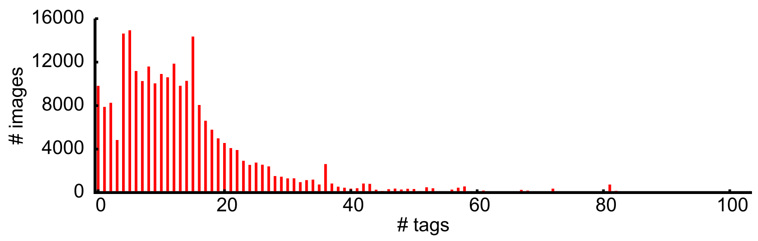


Figure 4.7: Histogram of the number of tags per image (including hypernyms) within the Flickr-250K database.

images with whole sentences and phrases.

4.6.3 Setup

We evaluate our mm-pLSA model by comparing it to various standard pLSA models on the Flickr-250K dataset (Section 3.1.1.1), which consists of 246,347 geo-tagged Flickr images associated with at least a single tag and belonging to one of 12 different categories.

We learned two 50-topic pLSA models that serve as baseline: one for tag features and one for visual features. All models were trained with 5000 images except the tag model that was learned from 10,000 images. The training corpus for tags had been widened to cover more of the "tag space" as the tag occurrence histograms were very sparse (see Figure 4.7).

The fast initialization strategy of the mm-pLSA mapped the two 50-dimensional image representations computed by the two base models (based on visual features and tags) to a multimodal topic distribution over 50 "super" topics. The global mm-pLSA models directly computed models with 50 topics. The number of iterations used during training and inference varied. All models were learned from 500 EM-iterations, except the mm-pLSA with the fast initialization method, which was computed using 50 iterations, since we already started from a decent initialization.

The only probability distribution computed during inference is the probability distribution $P(z_l^{top}|d_i)$ of the top-level topics given the document. Therefore the EM-algorithm converges faster than during training and we could reduce the number of iterations. For the inference of these topic distributions we used 200 iterations with the visual-based pLSA, tag-based pLSA, concatenated topic-based pLSA, and the fast initialization of the mm-pLSA. 75 iterations were used for the mm-pLSA with random as well as fast initialization.

We evaluated all the systems in a query-by-example task and evaluated the results by a user study as described in Section 3.3. 60 query images were selected and the L_1 distance was used to find their most similar images. The users were asked to rate the 19 closest results to each of our query images. We used the following scoring to get a quantitative performance measure: An image considered being similar received 1 point, an image considered somewhat similar received 0.5 points. All other images got 0 points. A mean score was calculated for each user; the mean over all users' means yielded the final score of the system being evaluated.

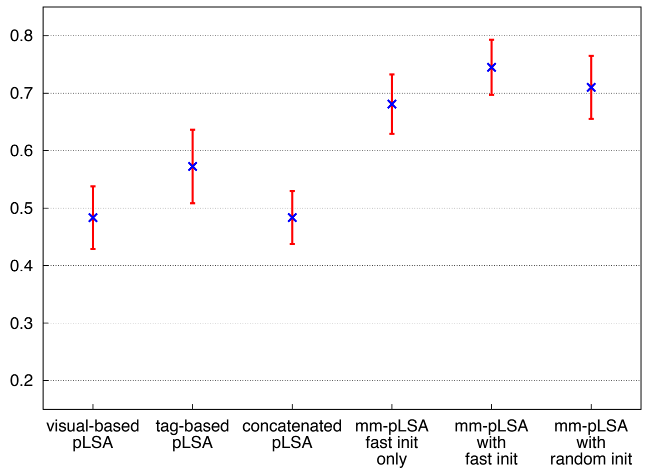


Figure 4.8: Scores for our different retrieval systems. Vertical bars mark the standard deviation of the individual mean scores of all users.

It is important to note that during the user study we presented the images to the participant *without showing their associated tags*. This is due to our application scenario, which follows the query-by-example paradigm: We assume that within a community-website image search is initiated by a user with selecting a query image. The associated tags are just used in the background by the retrieval engine to improve the search; the user does not enter his own keywords for querying.

4.6.4 Results

The results of our experiment are shown in Figure 4.8. The first two experiments denote the performance of the systems based solely on visual features or tags. "Concatenated pLSA" denotes the pLSA model computed from merging the words from the visual domain as well as the tag domain into one vocabulary. The fast initialization scheme that applies a third pLSA model on top of the two base models is termed "mm-pLSA (fast init only)". The mm-pLSA being initialized randomly or with the outcome of the fast initialization strategy is denoted as "mm-pLSA (random init)" or "mm-pLSA (fast init)", respectively.

It can be seen that the systems relying solely on tags perform better than the system relying solely on visual features. In contrast, the system aiming to fuse the modalities by concatenating the occurrence counts performs worse than the system based on tags only. Expectedly its performance is the same as of the system based on visual features only, as the few items added to the co-occurrence matrix by concatenating the tag occurrences to the visual feature occurrences are unlikely to have a major impact on the learned pLSA model training. Both mm-pLSA models with fast initialization only and with optimizing the already good initialization outperformed these systems by up to 24% which confirms the expected superior performance of multimodal multilevel models.

The randomly initialized mm-pLSA performs better than the pure fast initialization strategy (“mm-pLSA fast init only)”, but not quite as good as the mm-pLSA initialized with the outcome of the latter and further global optimization (“mm-pLSA with fast init)". This is in line with our expectations: we expect the randomly initialized model to perform inferior to its well-initialized counterpart. It should be noted that as the EM-algorithm already starts from a relatively good solution, the number of training iterations can be small. Therefore the mm-pLSA model with fast initialization and a few iterations of the global EM equations is a fast and effective way to obtain good results.

4.7 Evaluation on the Flickr-10M dataset

4.7.1 Models for Visual Features

SIFT As for the previous dataset we extract dense SIFT features from a regular grid over multiple scales. Again, we use a scale factor of $2^{\frac{1}{4}}$ for the spacing between scale level and a step size of 10 pixels. The descriptors were computed over image patches of 41×41 pixels.

Quantization of the features into visual words is performed by using a flat vocabulary of 10,000 visual words derived by k-means clustering as described in Section 2.2.1. In contrast to the previous setting we use a flat vocabulary rather than a vocabulary tree. Thus, we quantize the descriptors to visual words by an exact linear nearest neighbor search in order to minimize the (potential) loss in performance due to the quantization step early in our pipeline.

HOG We further extract Histogram of Oriented Gradient features using the improved 31-dimensional HOG features of Felzenszwalb et al. (2010) (see Section 2.1.4). Each individual HOG cell describes an image patch of 8×8 pixels. Similar to dense SIFT, these cell features are densely computed across several scales with a scale factor of $\sqrt{2}$. We concatenate the feature vectors of adjacent cells into multiple overlapping block features, each describing a 2×2 neighborhood. The resulting 124-dimensional vectors are then quantized into visual words using a flat vocabulary of 10,000 visual words created with k-means clustering.

4.7.2 Tag-based pLSA-Model

We build a tag vocabulary holding the most frequent and meaningful tags by applying a series of filtering steps: As for the previous dataset we filter tags by requiring that these occur more than T_{minOcc} times and are used by at least $T_{minUsers}$ different users. We further discard all tags that have less than three characters, contain numbers or are stop words. Table 4.2 shows the vocabulary sizes before and after filtering the available tags step by step.

Once the tag vocabulary is defined, the term-document matrix is built by counting the tag occurrences for each image. On average, the number of tags per images in our database is 7.7

Number of images	10,080,251
Number of images with tags	(90.4%) 9,109,593
Number of Flickr users	852,697
Vocabulary size after each filtering step	
Number of all tags (unfiltered)	1,691,336
Removal of tags with less than 3 characters	1,690,029
Removal of tags that occur in less than T_{minOcc} images	6,681
Removal of tags that contain numbers	6,500
Removal of stop words	6,467
Removal of tags used by less than $T_{minUsers}$ different Flickr users	3,158
Final vocabulary size	3,158
#Images with tags within vocabulary	(87.3%) 8,803,834
Vocabulary words present in Wordnet	(78.6%) 2,483

Table 4.2: The vocabulary size before and after each filtering step. T_{minOcc} has been set to $1,000$ occurrences and $T_{minUsers}$ has been set to 500 users.

– not taking tag-free images into account.

In our previous experiment (Section 4.6.2) we used Wordnet to expand the available image annotations. However, Wordnet is limited to English, and more than 20% of the words in our final vocabulary are not part of Wordnet (see Table 4.2). This may be caused by the use of different languages, slang words and abbreviations for annotations as well as the generation of raw tags that describe a specific location or scene. We assume that these annotations may carry specific and meaningful information for correct retrieval. We further expect, that the automatic expansion of tags to their hypernyms may also introduce additional noise to the annotations. Therefore, we do not use Wordnet for the following experiments and focus on the plain annotations provided by the users who uploaded the image themselves.

In our experiments we set the thresholds for the minimum number of occurrences $T_{minOcc} = 1000$ and for the minimum number of distinct users $T_{minUsers} = 500$ resulting in a vocabulary size of 3158 words. A larger tag vocabulary would be beneficial for a retrieval that is based solely on tags or other textual information. However, the training of the pLSA model is performed by sampling a subset of the whole database as training set (in this work 10,000 images). Thus, tags that do not occur within the set of training documents are not used for learning the pLSA model. In other words, tags that should be handled by the topic model need to be sufficiently frequent across all images in order to be included when (randomly) sampling the training set. This is the reason for choosing this relatively small vocabulary for tags.

4.7.3 Setup

For all models – low-level and high-level – we chose 50 topics. The number of iterations used during training and inference varied. All models were computed using 500 iterations except the mm-pLSA with the fast initialization method. In this case the model was computed using 50 iterations, since we already had a good starting point. All models were trained with 10,000 images, disregarding whether a unimodal pLSA model or a mm-pLSA model was used.

For the inference of these topic distributions we used 200 EM iterations for the visual- and tag-based pLSA, as well as for the concatenated topic-based pLSA and the fast initialization strategy. 50 iterations were used for the inference of the mm-pLSA models when randomly initialized or initialized by the fast initialization technique.

Following the same evaluation procedure as for the smaller Flickr-250K dataset we evaluated all the systems in a query-by-example task and evaluated the results by a user study with 9 users. 80 query images were selected and the L_1 distance was used to find their most similar images. The participants were asked to rate the 19 closest results to each of our query images. Again, we always showed the images *without* their associated tags as we study a query-by-example system.

As we also evaluate one system that is based solely on tags, it happens that there are several hundred up to thousands of images that have the same distance to the query image. This is due to the fact that images annotated with the same words will yield the same topic distribution disregarding the image content. For an unbiased evaluation the images in the result list need to be sorted by ascending distance (as usual) with an additional randomization step for images with equal distances. That is, images with equal distance to the query are randomized in their order while the ascending order of distances is still maintained for the whole list. This procedure eliminates any bias introduced by the order, in which similar images are found when scanning through the database.

We further impose two additional constraints:

- Images from the Flickr user who uploaded the query image will be ignored during retrieval.
- Any Flickr user may only contribute a single image to the result set. The image being selected is the one with the smallest distance; other images of that user will be ignored.

These restrictions basically prevent that the result set is dominated by image series uploaded by a single user and thus emphasizes both recall and diversity of the search.

4.7.4 Results

First we evaluate the fusion of the visual domain (represented by SIFT features) with the image annotations. The results of this experiment are shown in Figure 4.9. The first two experiments measure the performance of the systems based solely on visual features or tags and are labeled "pLSA on SIFT" and "pLSA on tags", respectively. "Concatenated pLSA" denotes the model computed from merging the words from the visual domain as well as the tag domain into a single feature vector. The straight-forward approach of applying a third pLSA model on top of the two base models is termed "mm-pLSA (fast init only)", while the mm-pLSA that is initialized randomly or with the outcome of the fast initialization is denoted as "mm-pLSA (random init)" or "mm-pLSA (fast init)", respectively.

It can be seen that the system relying solely on tags performs worse than the system relying solely on visual features. This is somewhat unexpected as in previous work tags were shown

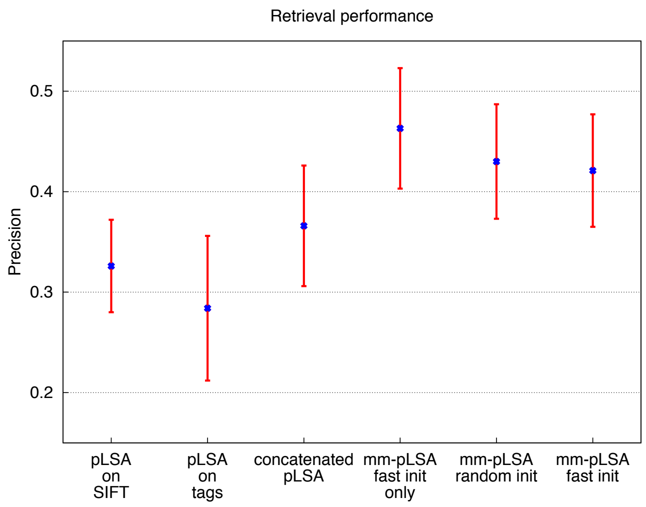


Figure 4.9: Scores for our different retrieval systems based on SIFT features and tags. Vertical bars mark the standard deviation between the users' means.

to outperform the visual features alone (Lienhart et al. 2009). Clearly, more work is needed to understand this effect. The third system, aiming to fuse the modalities by simply concatenating the occurrence counts – which were normalized to equal scale, in contrast to the corresponding experiment in Section 4.6.4 – already performs better than the unimodal systems but worse than than any mm-pLSA model.

All mm-pLSA models – with either using the fast initialization strategy or optimization of the already good initialization – outperform the unimodal models, which confirms the expected superior performance of multimodal models. However, the mm-pLSA models with global optimization (starting either from a random or fast initialization) perform slightly worse than the model that only performs the fast initialization. This is somewhat contradictory to previous works (Lienhart et al. 2009). We suspect that the global optimization drifts too much towards the textual domain. Given the poor performance of tags alone the overall performance then suffers. Another possible reason is that the global optimization is unable to optimize the solution from the fast initialization strategy any further but deteriorates it due to the less constrained optimization problem. Figure 4.6 shows that the log-likelihood of that model does hardly increase. This may be caused by too much noise on image annotations or a too small number of training documents.

The randomly initialized mm-pLSA model performs worse than the mm-pLSA with fast initialization strategy. This is in line with our expectations: we expected a random initialized model to perform inferior to its well-initialized counterpart.

In a second series of experiments we evaluate how the mm-pLSA can be used to fuse multiple features into a combined representation. In these experiments the two modalities that are evaluated are SIFT and HOG features. The results of the corresponding user studies are

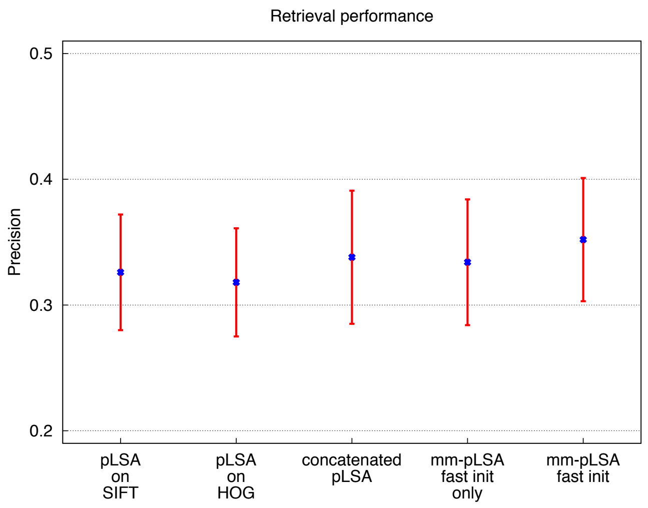


Figure 4.10: Scores for our different retrieval systems based on SIFT and HOG features. Vertical bars mark the standard deviation between the users' means.

shown in Figure 4.10. Similar to the previous experiments, the pLSA on the concatenated feature histograms does hardly improve over the better of the two modalities. This observation underlines the importance of hierarchical models even for assumed easy tasks such as multi-feature combination. Despite the close relation of these gradient-based features one can see that a stepwise combination of three pLSA models (termed "mm-pLSA fast init only") further improves the retrieval, but is slightly outperformed by the mm-pLSA model that performs a global optimization.

Finally, for visualization purposes two example queries and the topmost retrieved images of various variants are shown in Figure 4.11.

4.7.5 Discussion

It remains subject of future research why the mm-pLSA model with fast initialization strategy and global optimization performs worse than expected on this data set but outperformed all other variants in previous work in the case where SIFT features and tags are combined. A probably related issue is the inferior performance of the tag-based model. One possible solution may be to upscale the tag vocabulary in order to describe such a large dataset more accurately. However, one needs to address that only a few thousand tags occur frequently and most are used only by a relatively small number of images. Here, stemming could be of help. Another potential solution may be to also include the provided textual image description of Flickr images rather than tags alone.

For further insights we visualize the conditional probabilities of the modality-specific low-level topics ("*subtopics*") given the top-level topics $P(z_k^v|z_l^{top})$ and $P(z_p^t|z_l^{top})$ ("*supertopics*") of the mm-pLSA training. We chose the mm-pLSA with fast initialization strategy and plot these

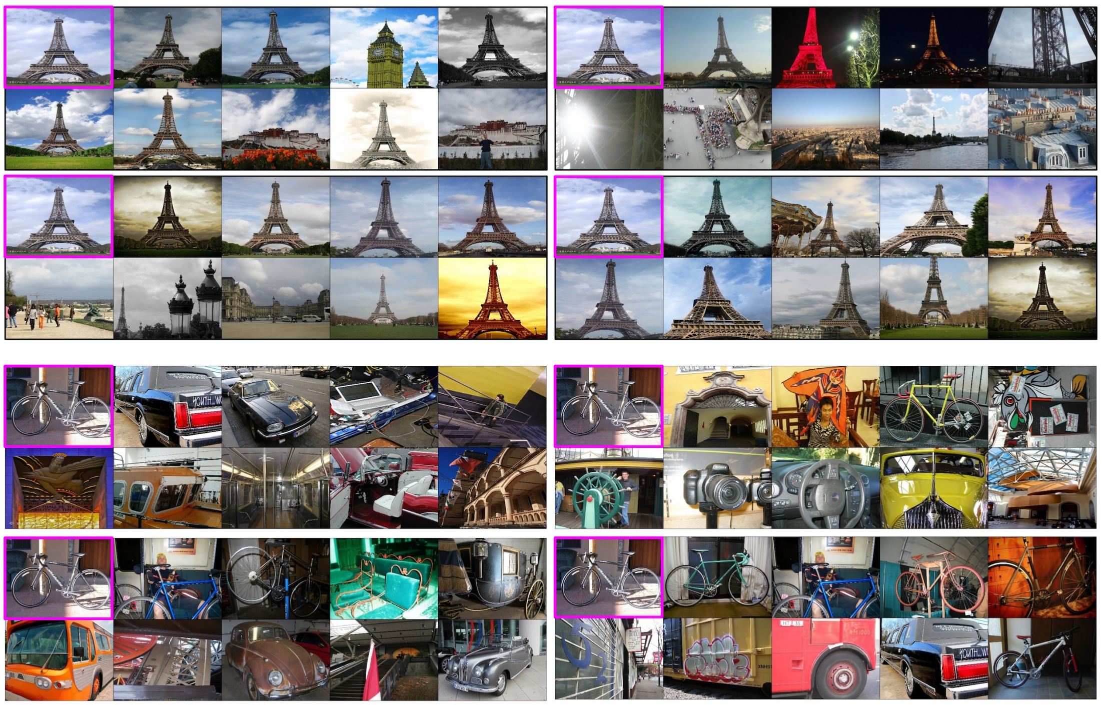


Figure 4.11: Examples of retrieval results on the Flickr-10M dataset for the different approaches and two different queries. The query image is shown at the top left corner (pink frame) followed by the retrieved images. **Query: "Eiffel Tower"**: Upper left: pLSA on SIFT features. Upper right: pLSA on tags. Lower left: mm-pLSA (the fast initialization only) on both SIFT and tags. Lower right: mm-pLSA with fast initialization and global optimization on both SIFT and tags. **Query: "bike"**: Upper left: pLSA on SIFT features. Upper right: pLSA on HOG features. Lower left: mm-pLSA (the fast initialization only) on both SIFT and HOG features. Lower right: mm-pLSA with fast init and global optimization on both visual feature types.

probabilities as a matrix, where the actual probability value is mapped to a color ranging from dark black for 0 to bright white for 1. Each row l of such a matrix represents $P(z_k^v|z_l^{top})$ on the left half (split by the red line) and $P(z_p^t|z_l^{top})$ on the right half. The columns then enumerate the subtopics k and p correspondingly. Note that each row sums up to 1. Therefore one can visually identify the mixture of the topics by looking at each row.

The conditional probabilities for SIFT features and tags are shown in Figure 4.12. It can be seen that most entries with high probability value are present for tags only (right half of Figure 4.12). The visual part (left half) has no peaks but is apparently less sparse. One can further observe that the entries in each row with a significant probability (the non-black entries) are either on the visual or on the textual side, not on both. There is no direct correspondence between visual topics and textual topics. Apparently, the mm-pLSA basically acts as a kind of auto-selection mechanism, selecting either a visual or a textual sub-topic distribution for each top-level concept. The mixture of visual and textual description is thereby achieved by representing each individual image by a mixture of such supertopics. These are in turn mutually

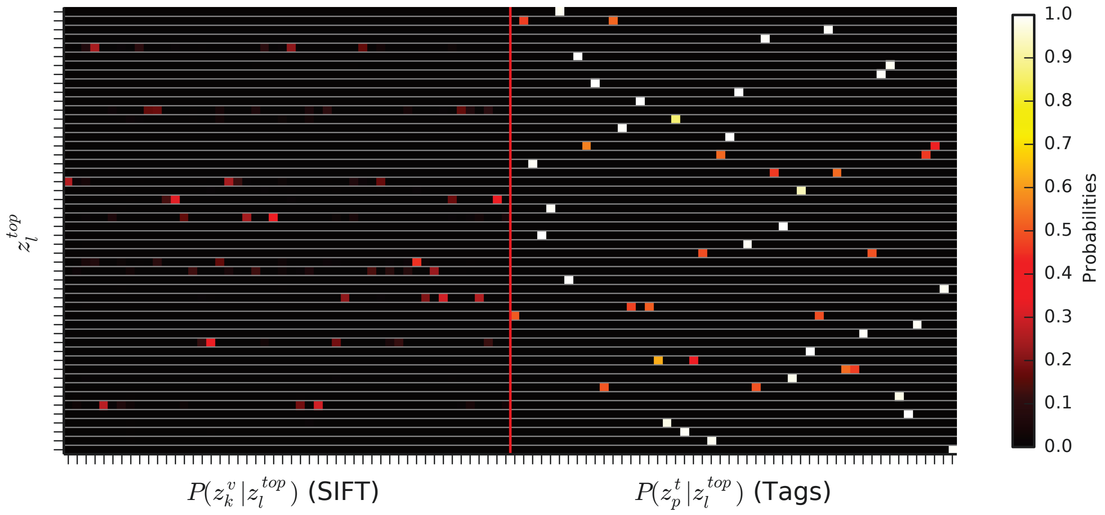


Figure 4.12: Visualization of the matrix $P(subtopic|supertopic)$ for the mm-pLSA on SIFT features and tags. One row in this matrix denotes the conditional probabilities $P(z_k|supertopic)$ (left half) and $P(z_p|supertopic)$ (right half), which sum to 1. The subtopics for the SIFT features are shown on the left half, the subtopics describing tags on the right half.

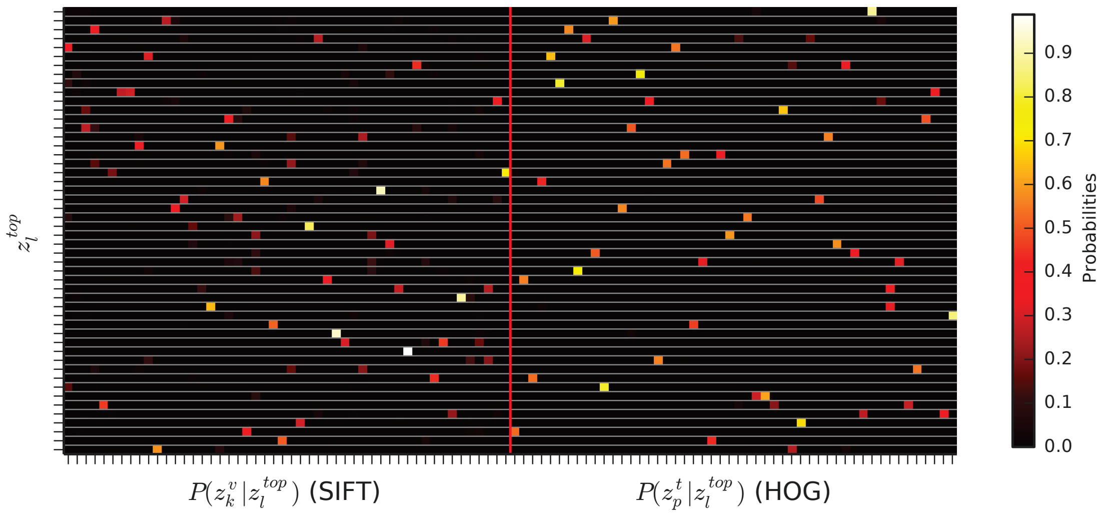


Figure 4.13: Visualization of $P(subtopic|supertopic)$ for the mm-pLSA on SIFT and HOG features. One row in this matrix denotes the conditional probabilities $P(z_k|supertopic)$ (left half) and $P(z_p|supertopic)$ (right half), which sum to 1. The subtopics for the SIFT features are shown on the left half, the subtopics describing HOG features on the right half.

exclusive on their subtopic representation, but the mixture of these describes both modalities.

This is different for the multi-feature model combining SIFT and HOG features. In Figure 4.13 one can see that the supertopics represent a real mixture of subtopics from different modalities. Seemingly, the close relation of the two types of visual features is directly reflected in the top-level concept.

4.8 Summary

In this chapter we presented a general scheme, the multilayer multimodal probabilistic Latent Semantic Analysis. It extends the single-layer pLSA to the concept of layered or hierarchical topics – a natural way to describe an image composition. It also allows to grasp concepts from different modalities such as visual features and text and describe them with a single compact representation by the distribution of top-level concepts.

We described the structure of our mm-pLSA model as well as training and inference and further proposed a fast initialization technique making the mm-pLSA efficiently computable in a step-wise forward manner. The overall approach was evaluated on two large-scale datasets of 250,000 and 10 million images originating from a real-world database. The result consistently show that the mm-pLSA outperforms unimodal pLSA significantly and can be applied to fuse information from very different domains.

Part III

Feature Bundling for Object Retrieval

5 Feature Triples

In collaboration with Yahoo! Inc. – the company running Flickr – we developed a retrieval-based technique to detect brand and product logos in images. The main motivation is to boost click-through rates and thus advertising revenues on large community-driven websites such as Flickr, YouTube, Facebook or Google+ by improving the user experience of advertisements. This may include the selection of ads based on the user's interests and his profile but also the selection of ads based on the current visual content presented to the user such as images or videos. For instance, if a user visits a sport news site and his favorite team wears clothes of a particular brand, it might be a good idea to present him advertisements and offerings of exactly this brand and not of those brands supporting the opponents. For such applications logo recognition is the back-end service necessary to extract the needed semantics from images that are then exploited for better user experience and at the end for more profitable services.

We entirely focus on the logo recognition step itself and consider it a subtask of object recognition as most logos can be considered as objects with a planar surface having only a moderate variance in their appearance. A pleasing property is that logos are usually visually distinctive and designed to catch someone's attention. Also, many logos contain text, which on the one hand is often a prominent description of the brand. On the other hand, text-like structures are particularly difficult to deal with as they have repeated visual primitives that can occur across a wide range of images. In the end, the recognition of logos in natural images is challenging due to major changes in lighting, scale and perspective, making this setting significantly different to pure near-duplicate retrieval.

There are many powerful object recognition schemes, the most popular at the moment probably being deformable part-based models (Felzenszwalb et al. 2010). However, such methods require to learn and apply one or more models per class. From an industrial point of view classification-based approaches that follow the 1-vs-all classification paradigm are rather undesirable: these implicitly require that given an unknown test images the models of *all* classes must be evaluated on this single image in order to determine if objects of the corresponding classes are present. There are multi-class classification schemes, but these do not scale well with hundreds and thousands of classes.

In contrast, our long-term goal is to perform logo recognition on thousands of images per minute, whereby the number of different logo classes is huge. In that case the usage of one-vs-all classifiers is most likely too expensive and thus infeasible. In general a logo recognition system should be able to determine quickly if an unknown image contains a logo of a certain class. While it is certainly doable to test an incoming test image for *every* logo class present in the database, this is not an option due to speed issues, the expected high false alarm rate and especially because of the likely disagreement among the multiple classifiers. Typically, a logo database contains multiple samples of thousands of brands. While the commercial interest to detect logos in images is huge, any recognition system needs to address the scalability constraints in terms of images processed per minute and numbers of classes supported. At the same time a very high rate of recognition accuracy must be maintained.

In this chapter we describe a system heavily tuned towards efficient multi-class object recognition with high precision at the cost of recall. Our approach may be combined with classification-based approaches and acts as a filtering before more costly classifiers are applied.

At the basis of our system we use local features, which have been proven effective for object retrieval in general. In that setting, the bag-of-word model is probably the most common technique for image retrieval. However, experience shows that individual visual words lack distinctiveness, which leads to the retrieval of many incorrect yet high-scored images. This is often addressed by employing post-retrieval geometric verification steps to discard images that were retrieved, but do not share geometrically consistent visual features with the query.

Consequently, one goal is to incorporate geometric information into the visual signature that is being used for searching similar images in order to suppress false positives. The key challenge is to derive a search and index scheme that is capable of robust similarity search beyond pure near-duplicates. In other words, we want to find similar images while maintaining translation, rotation and scale invariance of the underlying visual description. Solutions for visual similarity search that satisfy these requirements are rather tricky. Most methods we are aware of are either targeting near-duplicate search using a global signature or do not use the signature for search but rather for post-processing.

In this work we describe a system that encodes both the appearance as well as the spatial configuration of three local features (in the following termed *feature triples*) into the visual signature. The signature is translation, scale- and rotation invariant and due to the geometric information highly distinctive. Logo classes are represented by distinguished spatial configurations of feature triples being learned on a training set. By exploiting that feature triples themselves consist of feature pairs, a hierarchical index can be built that allows for efficient search of feature pairs and triples. Given a query image and its feature triples, the logo detection is performed by searching for matching spatial configurations within the database of feature triples from multiple classes.

The remainder of this chapter is organized as follows. Section 5.1 surveys related work relevant to our approach. In Section 5.2 we describe how to determine spatially consistent feature pairs and triples from training images. The usage of this representation in combination with our index for efficient logo recognition is discussed in Section 5.3. Finally, the results of our experiments are presented in Section 5.4, followed by the conclusions in Section 5.5.

5.1 Related Work

There are several publications that address the retrieval of printed logos, that is, for efficient search in logo databases of design patents. However, logo recognition in real-world images has not gained as much attention yet. Bagdanov et al. (2007) determine correspondences between SIFT descriptors of video frames and reference images in order to detect whether a logo is present. The logo is then further localized by estimating the center of all the matches. Joly and Buisson (2009) propose a new query expansion strategy for querying a database with SIFT descriptors followed by a geometric consistency check. Both approaches do not scale well with growing database size.

A turning point in scalable object retrieval, which also provides the basis for our approach, has been the introduction of the bag-of-words approach (Sivic and Zisserman 2003). At its core is the quantization of local features to visual words allowing efficient indexing and search.

Fu et al. (2010) combine three types of local features within the bag-of-words framework to capture gradient distribution, shape and patch appearance and adaptively weight their combination during retrieval. Jiang et al. (2011) use a regular grid to bundle local features that reside in the same grid cell. Similar to Partition Min-Hashing (Lee et al. 2010) each grid cell is described by a bag-of-words but the lookup for matching grid-cells is done by bag-of-words retrieval followed by branch-and-bound object localization. Letessier et al. (2011) perform feature selection to determine consistent visual words. This allows to reduce the number of visual words used to query the inverted index with no or little loss of accuracy.

Many approaches building on the inverted index as indexing data structure employ geometric verification steps to filter the high number of false-positives. By embedding the spatial information in the index one can significantly reduce the number of false positives. However most approaches use plain visual words to query the index and store additional geometric information in the payload associated with each word for immediate yet post-retrieval verification of the matches between the query word and retrieved candidates. Due to memory constraints, only compressed or quantized geometric information is stored and weak-geometric consistency is checked on query time (Wu et al. 2009, Jégou et al. 2008; 2009a, Perdoch et al. 2009).

In contrast, we explicitly focus on the encoding of the geometry *into* the query itself such that (1) the index neither has to store the additional information nor (2) the result set of each individual visual word needs to be verified at query time. To achieve this, the spatial structure is encoded with a hash function. In short, similar regions in two images are determined by indexing the geometric layout of local features in hash tables. Our work in this area is in the spirit of geometric hashing (Rigoutsos et al. 1997), however we do not have a single model image but create a model for each logo-class out of several training images. The result of the training is a set of triples of interest points, which consistently appear across different training images.

Similar to our approach Zitnick et al. (2007) use feature triples to describe product logos. In contrast to our work they focus on the post-retrieval verification by exploiting the geometry of feature triples and use an inverted index holding all possible n^3 feature combinations out of n features of a reference image. An exhaustive search for all combinations of feature triples in the query image is then used for retrieval. This basically renders this method impracticable for real-time applications. We address this issue by a specialized index structure that can be utilized for efficient search of feature triples.

Cranston and Samet (2007) review different ways to encode the geometry of triangles and pay special attention to degenerated cases that we explicitly discard in advance. Poullot et al. (2009) use bucketing techniques to build signatures from local feature triples. In contrast to our approach they build a signature for the whole image and their approach is more suited to near-duplicate detection. Avrithis et al. (2010) incorporate global geometry in the index by means of hashing the global layout of all features. Rather than building a global geometric representation, we index spatial structures that describe a small region of the image, making our approach more suitable for the detection of small logo regions. Kalantidis et al. (2011) use feature triples and distinctive signatures are derived from a multi-scale Delaunay triangulation of interest points. However, this approach heavily relies on the tessellation and requires very stable interest point detections.

In this work we use SIFT (Lowe 2004) descriptors derived from hessian-affine interest points (Mikolajczyk et al. 2005) to describe images, as these are more robust to image tilt and perspective transformations than other features. However, our approach may be used with any other local feature as well.

5.2 Training

In the following we describe a method that automatically discovers local feature pairs and triples with consistent spatial layout from training images of a certain class. These feature pairs and triples are described by a representation that not only encodes their visual appearance but also their spatial layout. The mined feature configurations are used to represent the logo classes and indexed into a hash table that is used for logo detection described in the subsequent Section 5.3.

In fact, we use the underlying representation of feature triples as triangles twice: During training it is used to find feature triples that have a similar spatial layout across training images. Once indexed the triangle representation is further used for detection: logos are discovered by testing if feature triples of an unknown image are present in the index.

We would like to stress the fact that the training we employ can be replaced or combined with other approaches for finding local feature correspondences between image pairs.

5.2.1 Modeling Logo Classes

We model our logo classes by the appearance of certain distinctive configurations of local features. More specifically, we look for *spatially consistent* local feature configurations that consistently appear at prominent locations of the logo when seen from multiple different views. In order to obtain such configurations we search for those that appear across pairwise combination of training images. Given n training images per class, our training procedure processes $\frac{n(n-1)}{2}$ image pairs and collects those feature triples that appear across multiple training images.

This implicitly requires that the training images show the object of interest with varying background, otherwise spatially consistent feature triples might also be mined from background regions. In practice, the training images we used have a clutter-free background and due to our distinctive visual description feature triples are mined only from the logos themselves.

For several reasons the matching procedure sometimes is not able to find feature triples for certain training image pairs. For instance, the matching of individual features fails for image pairs where the logo's colors are inverted or where images differ greatly in their perspective viewpoint. One may see the pairwise matching as the creation of a graph where images represent vertices that are linked by feature triples with consistent spatial layout. The benefit of the pairwise matching is the implicit discovery of multiple connected components in this graph.

Eventually, all feature triples determined from pairwise combinations of training images form the model of a logo class covering those views of the logos that we present in the training set. In the following we describe the discovery of spatially consistent feature pairs and feature triples from pairs of training images.

Throughout this chapter, we represent an image by the set of its local features $\mathcal{I}$. Each individual feature $i \in \mathcal{I}$ is described by its position, scale, orientation. Its visual appearance is captured by its visual word label $v(i)$. A *correspondence* $i \leftrightarrow i'$ between two local features i and i' means that these have been considered in some sense as similar and are declared as *matching*. Pairs of features are naturally represented as an *edge* (i, j). An *edge correspondence* $(i, j) \leftrightarrow (i', j')$ denotes that two pairs of local features are considered as similar.

5.2.2 Discovery of Spatially Consistent Feature Pairs

We start the automatic discovery of feature pairs that match across a pair of training images $\mathcal{I}_A$ and $\mathcal{I}_B$ by mining visual feature correspondences. The correspondences are obtained in

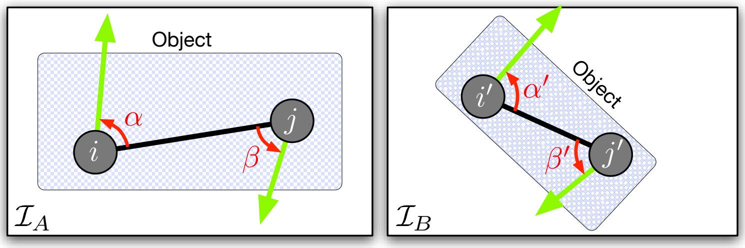


Figure 5.1: Relative position of feature pairs in two images. Green arrows indicate the orientation of the corresponding SIFT feature.

constant time by determining those local features that have been assigned to the same visual word. Given the visual word matches $\mathcal{V} = \{i \leftrightarrow i' | i \in \mathcal{I}_A, i' \in \mathcal{I}_B, v(i) = v(i')\}$ we further compute the Euclidean distance between the corresponding descriptors. The matches are then sorted in ascending order by the distance between descriptors. For each image pair we keep up to N_{feat} feature correspondences with the smallest distances for further processing.

Given the correspondences of individual features, we proceed by searching for corresponding pairs of them, i.e., edge correspondences. This is done by determining pairs of correspondences $\{(i,j) \leftrightarrow (i',j') \mid i \leftrightarrow i' \in \mathcal{V}, j \leftrightarrow j' \in \mathcal{V}, v(i) < v(j)\}$ from the set of individual correspondences. To remove ambiguities we require that edges must consist of different visual words that are ordered by their label. In particular, we are only interested in those edge correspondences where the participating local features have a consistent spatial layout in both images. For that, the absolute distance between points cannot be taken into account, as it would break the desired scale-invariance of our representation. Thus, we explore the *relative orientation* of a feature i to a feature j by determining the relative angle α between the dominant orientation of i (as given by the interest point detection itself) and the spatial position of j. Vice versa, the reversed relation is captured by the angle β between the orientation of j and the position of i.

If a feature pair has a spatially consistent counterpart in the other image, the relative orientations α and α', as well as β and β' are similar to each other (see Figure 5.1 for an illustration). Naturally, we can represent such relationship as the correspondence between two edges e and e' enriched with the corresponding relative orientations: $e \leftrightarrow e' \equiv (i,j,\alpha,\beta) \leftrightarrow (i',j',\alpha',\beta')$. In the remainder of this chapter the term edge always refers to two points plus their relative orientation angles. In the following we construct a filtering function to determine edge correspondences with similar spatial layout. For any two pairs of points we compute the relative orientation of these two correspondences *across* both images by computing the difference of the relative angle:

$$\Delta\alpha = \mathit{angle\_diff}(\alpha, \alpha') \quad \text{and} \quad \Delta\beta = \mathit{angle\_diff}(\beta, \beta') \tag{5.1}$$

Here, $\mathit{angle\_diff}(\alpha, \alpha')$ denotes the smallest sign-preserving enclosing angle between α and α'

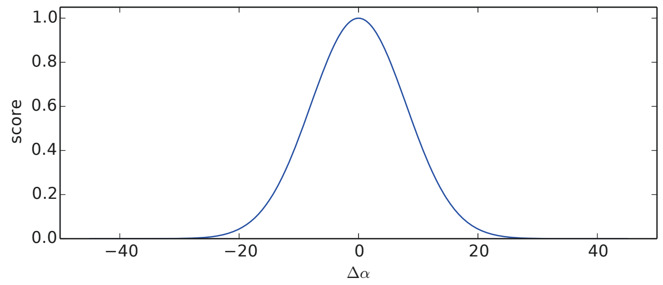


Figure 5.2: Score function for scoring the difference of angles as in Equation 5.2.

taking into account the circular wrapping at 0° and 360° (see Appendix B for details). The similarity score $s(\Delta\alpha)$ is based on the difference of the angles. We compute a normalized score for this difference of the two angles:

$$s(\Delta\alpha) = \eta \; e^{-\frac{(\Delta\alpha)^2}{2\sigma^2}}. \tag{5.2}$$

This score yields a value that indicates the similarity of the angles. We empirically adjusted $\sigma = 8$ and normalize the score by η such that the function has its maximum of 1.0 at 0° difference. With increasing $\Delta\alpha$ the score quickly and smoothly decreases towards zero for differences higher than 25° (see Figure 5.2). While $s(\Delta\alpha)$ only considers a score for the relative position of j seen from point i, we further use $s(\Delta\beta)$ to describe the spatial consistency from the view of point j relative to position of i. The final symmetric score is then defined as

$$sim_{edge}(e, e') = s(\Delta\alpha)s(\Delta\beta). \tag{5.3}$$

Note that the score will quickly drop to 0 if one of the angles is not consistent across the image pair. While the comparison of such a difference of angles could yield a binary result the use of a continuous score function allows to sort the matching edges according to their match quality.

We compute the score sim_{edge} describing the consistency of the spatial layout for all possible edge correspondences between the pair of images. The top N_{edge} edge correspondences that have a score above a threshold T_{sim} form the set E_{match} for further processing. Eventually, this filtering discards many potential combinations of features that are not spatially consistent. The remaining edge correspondences serve as a starting point for deriving feature triples.

5.2.3 Triangle Representation

In previous qualitative experiments we experienced that even pairs of local features are often not as discriminative as desired for logo recognition. Also, as we want to capture the spatial structure of objects we chose to describe them by feature triples of local features. Such feature triples naturally form a triangle (if the point configuration is non-degenerated). As for the edges we capture both the visual appearance of the individual local features as well as their

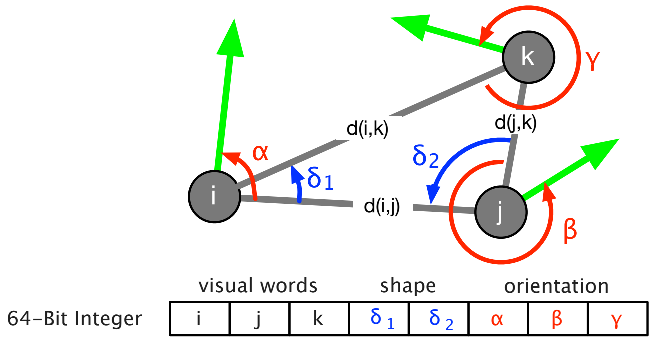


Figure 5.3: Representation of a triangle

relative position in a highly distinctive signature. Each triangle is described by an 8-tuple:

- The three visual words of each of the points i, j, k. To remove ambiguities the points are ordered by their visual word labels such that $v(i) < v(j) < v(k)$.
- The angle δ_1 between edges (i, j) and (i, k).
- The angle δ_2 between edges (j, k) and (j, i).
- The relative orientations α, β and γ of the three points.

Figure 5.3 depicts the information extracted from each triangle to create a signature.

There are many equivalent ways to describe the shape of a triangle. The proportion of the edge lengths may be one choice, but we chose to capture the shape by the two angles δ_1 and δ_2 as we assume that the ratio of angles is less affected by out-of-plane rotations than a ratio of edge lengths. By observation we then found that taking the orientations of the features themselves into account leads to better performance on letter-like logos. In this case the orientations of the features are also important to describe the layout. Therefore we include the *relative* orientations α, β and γ (see Figure 5.3) of the three points in the signature. These orientations depend on the orientations of the SIFT features but are relative to each other. Therefore both in-plane rotation invariance and scale invariance of the triangle representation is maintained.

In all operations we discard degenerated triangles that carry little spatial information and are not descriptive by imposing the following constraints:

1. Each of the three points must be a different visual word. This constraint addresses that logos are often framed with some kind of border. Along that border many identical visual words may be detected. Feature triples consisting entirely of such words do not carry sufficient discriminative information.
2. Points must not be in a degenerated position. That is, no two points must coincide. Even stricter, the spatial distance between all of the points i, j, k has to be above 5 pixel and the minimum angle of the triangle has to be $15°$. This discards degeneracies where at least two points are located very close to each other and the resulting triangle would hardly

describe spatial structure. The minimum distance further deals with the fact that the interest point detector tends to over-detect and returns many interest points that only differ slightly by their location and scale.

3. Finally, once two features on the actual logo were detected we observed cases where a bad detected triangle was caused by the detection of a correct edge and a false positive third point outside the actual logo. Thus we further constrain the eccentricities $\frac{d(i,j)}{d(i,k)}$ and $\frac{d(j,k)}{d(i,j)}$ such that these are in $[\frac{1}{3}, 3]$. This constraint discards triangles which carry little information of spatial structure and also improves the locality of the detection.

5.2.4 Discovery of Spatially Consistent Feature Triples

Given the set of detected edges correspondences E_{match}, we can construct a triangle (i, j, k) by selecting an edge (i, j) from E_{match} and a third arbitrary point k from V_{match} where $V_{match} = \{i, j | (i, j) \in E_{match}\}$. While the search for edge correspondences is of quadratic nature – as it requires to evaluate the score function for every feature correspondence between a pair of training images – it is still feasible as there are usually only a few hundreds of those per image pair. Moreover, the score function can be efficiently evaluated. However, moving from pair-wise to triple-wise feature combinations renders exhaustive search techniques impractical.

To determine feature triples that have a geometrically consistent spatial layout across the two training images $\mathcal{I}_A$ and $\mathcal{I}_B$ in sub-linear time, we make use of the same technique we later use for indexing. We store the triangle representations of all triangles within image $\mathcal{I}_A$ in a hash table by element-wise quantizing the triangle representations into discrete hash keys. For that the angles capturing the triangle shape δ_1 and δ_2 as well as the relative orientations α, β, γ are quantized into regular bins. The final triangle signature consists of these 5 discrete numbers and the three visual word labels. For space-efficiency each signature is packed into a 64-bit integer value (see Figure 5.3) used as hash key to access the hash table when indexing or searching.

The "search" for triangles that are spatially consistent across both images then proceeds as follows: All possible triangles and their quantized representation are computed for image $\mathcal{I}_B$. By performing look-ups in the hash table one can immediately determine whether these discrete signatures have a matching counterpart. In other words, once triangle representations have been quantized and indexed, matching triangles can be found in constant time by querying the hash table. Thus, spatial consistent triangles are determined by finding identical signatures.

Quantizing the angles and proportions using hard boundaries introduces errors and potential loss of matches that eventually decreases recall. We have extensively experimented with the parameter settings to optimize the recall, while maintaining the specificity of the triangle signatures. We observed the best performance when multiple signatures for each triangle were stored. Similar in spirit to the multiple assignment strategy where local features are quantized to the k-closest cluster centers instead of a single one, the five different angles are quantized to both the best bin and the second best bin. Furthermore, the circular wrapping of angles is taken into account. We explicitly omit the multiple assignment step for the visual words

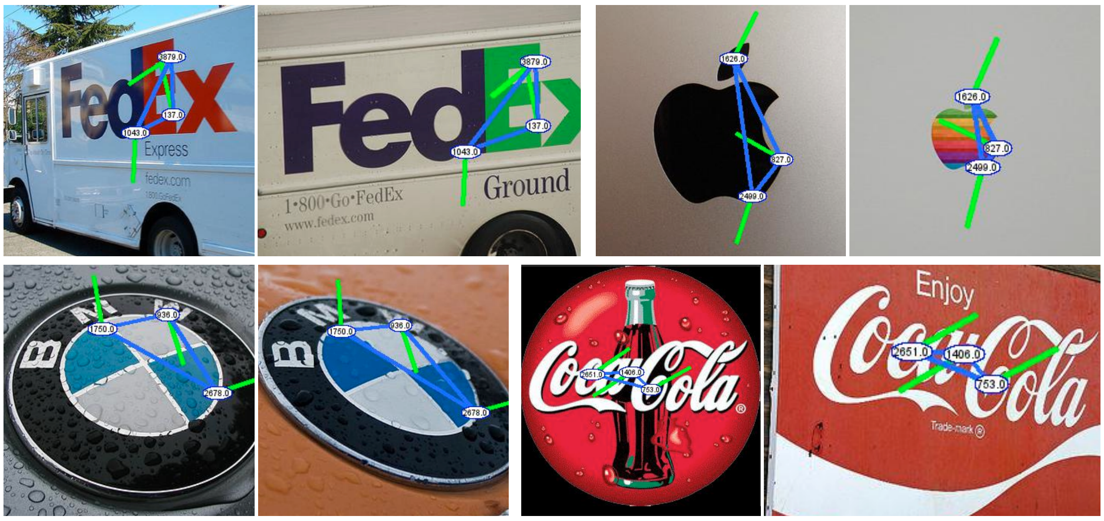


Figure 5.4: Examples of spatially consistent triangles across pairs of training images from the FlickrLogos-32 dataset. Blue lines denote the edges, green lines the feature orientations. The numbers denote the visual word label.

themselves as we already use small vocabularies of a few thousand visual words. Therefore, when constructing the signature for each triangle, multiple signature variants are generated such that we store 32 ($= 2^5$) different signatures for each triangle. The optimal bin size for the quantization is evaluated in Section 5.4.2. Note that for every triangle in $\mathcal{I}_B$ we only generate a single discrete representation and only query the hash table of $\mathcal{I}_A$ once.

To summarize, our training procedure produces a set of spatially consistent triangles per class by aggregating those found across individual pairs of training images. Due to memory constraints we control the memory consumption with the following parameters: The number of feature correspondences N_{feat} kept for each training image pair – ranked by the Euclidean distance between descriptors – was set to 300. Out of these point correspondences we determine up to $N_{edge} = 2000$ spatially consistent edge correspondences per image pair. Finally, for efficiency during logo detection, we sort the triangles of each logo class by the scales of the participating features and keep only the 500,000 triangles with the largest scales. Figure 5.4 shows several examples of spatially consistent triangles determined by our learning procedure.

5.3 Recognition

5.3.1 The Cascaded Index

Given an unknown test image we have no prior knowledge regarding which logo it contains - if any at all - and at what locations, scales and sizes. As we exploit a higher-order visual description that aggregates three distinct local features into a single description, testing an image for the presence of a logo by evaluating each combination of features is infeasible.

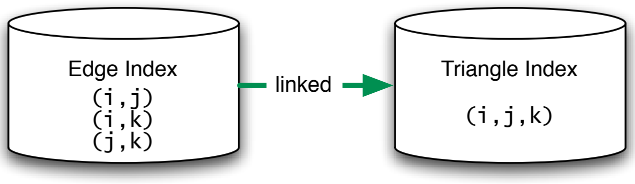


Figure 5.5: Schema of the cascaded index. i, j and k denote the local features that have been indexed. (i, j) denotes a pair and (i, j, k) denotes a triple of local features.

For instance, given a set of points $\mathcal{V}$, the space of all possible combinations of points into 2-tuples or edges is given by the Cartesian product of the set $\mathcal{V}$ with itself as $E = \mathcal{V} \times \mathcal{V}$. Similar, the space of all possible 3-tuples or triangles is spanned by $\mathcal{T} = \mathcal{V} \times \mathcal{V} \times \mathcal{V} = \mathcal{V}^3$. An exhaustive search for a certain combination of three points (i, j, k) would thus require to fully search the discrete space $\mathcal{V}^3$, which is intractable.

However, by exploiting the structure of the data we index, we can scan the sparsely populated high-dimensional feature space of feature triples by scanning the lower-dimensional subspace of feature pairs part of it first. Therefore we exploit a *cascaded index*. The cascaded index holds both lower- and higher-dimensional representations in terms of their combinatorial complexity. These are linked such that each lower-dimensional representation *is part of* a higher-dimensional representation.

In our case, we index local feature pairs and their relative spatial layout within an *edge index* as the lower dimensional description. The higher-dimensional *triangle index* contains feature triples including their relative layout. The two indexes are linked, such that the edge index contains only edges that are part of the triangle index. Figure 5.5 illustrates this scheme. This dependency allows us to perform efficient querying in two steps: First feature pairs are sampled and used for querying the first stage of the cascaded index, i.e., the edge index. The majority of queries will not yield any hits in this index but the ones which do serve as starting points for constructing more complex queries that are issued to the second stage of the cascaded index, i.e., the triangle index. We then accumulate the numbers of hits in this index as votes for the presence of the corresponding class.

In this work we limit ourselves to a cascaded index consisting of two linked index structures where the first stores 2nd-order features (edges derived from feature pairs) and the second stores 3rd-order features (triangle derived from feature triples). Future work might extend this to multi-layer cascaded indices for efficient search in databases of higher-order features. In our case, both indexes are implemented as hash tables but the concept is not limited to these.

5.3.2 Querying the Cascaded Index

Querying the Edge Index We employ two complementary edge-sampling methods to bootstrap the detection:

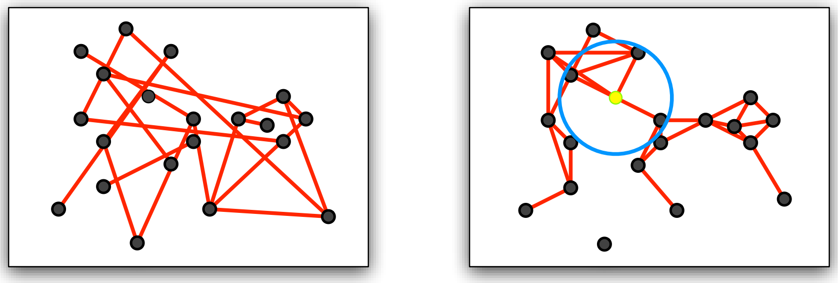


Figure 5.6: Examples of sampled edges (red) from local features (black dots) with two different sampling strategies: **Left:** Monte Carlo sampling of edges. **Right:** Proximity sampling. The neighbors within a given distance (blue circle) of a point (yellow) are used to sample edges. This process is repeated for all local features.

1. **Monte Carlo Sampling:** The edge index is queried with randomly selected feature pairs out of all visual features of the input image. As for the triangle representation we encode both the visual appearance and the relative spatial position of the feature pair into a discrete signature used as hash key. The hash table is then used to determine whether it contains an identical edge signature. This kind of sampling is shown in Figure 5.6 (left).
2. **Proximity Sampling:** As the number of interest points in logo regions roughly correlates with the size of the logos, bigger logos are more likely to be covered when random points are selected. In order to cover also even very small logos by multiple edge samples we further scan the immediate neighborhood of each local feature in addition to the Monte Carlo sampling. For that, close pairs of points are sampled by selecting the neighbors within a distance of $5px$ to $30px$ of an interest point. As the radius is limited, a relatively small number of edges will cover the whole image on very small scales. This sampling scheme is illustrated in Figure 5.6 (right).

Querying the Triangle Index The key idea is that once we discover a certain edge of the query image that is contained in the edge index it further implies that a part of a triangle within the triangle index has been detected. Thus, the more complex triangle representation is constructed by randomly sampling an edge from the set of detected edges and sampling an additional point belonging to one of these. Duplicates and degenerated triangles (see Section 5.2.2) can be discarded without actually querying the index.

Once we find matching triangles, the votes for each class are accumulated. The frequency of detections per class is used as confidence score for a per-class decision: The system decides by a class-specific threshold T_c on the detection counts whether a certain object class is present in an image. The decision thresholds are learned for *each class separately* (see Section 5.4). In general, once more than T_c feature triples of a certain class c are detected the logo is considered to be present in the image.

Figure 5.7 shows example detections of different logos. One can see that in contrast to

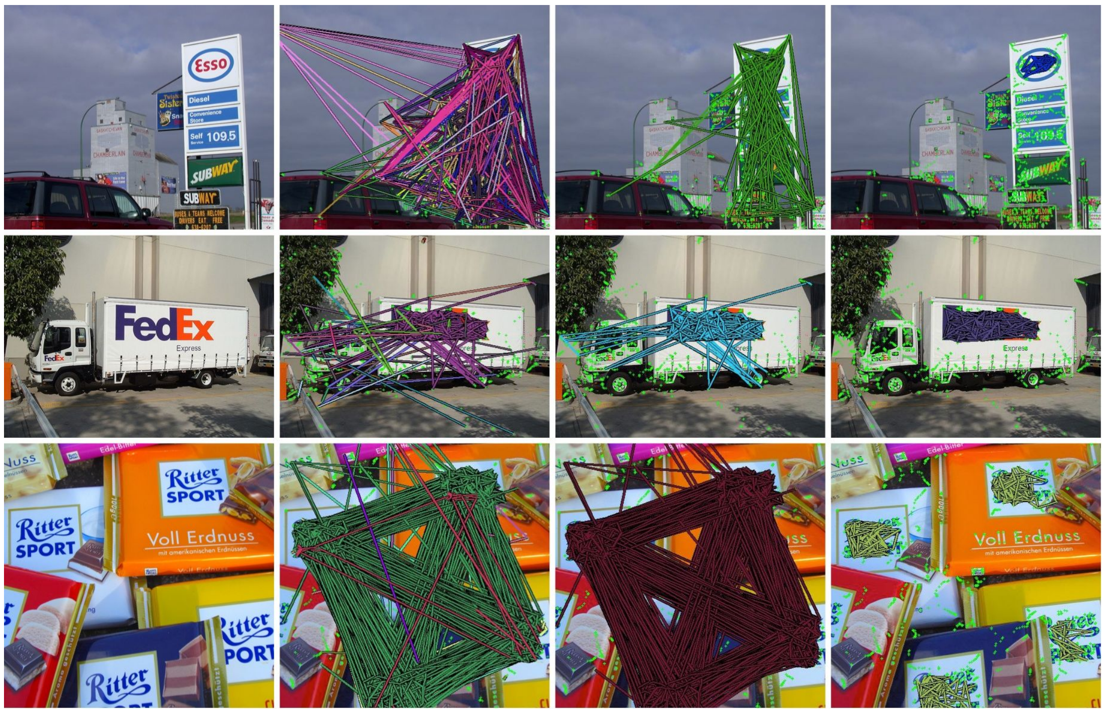


Figure 5.7: Example detections of the Esso logo (top row), the FedEx logo (middle row) and the Ritter Sport logo (bottom row). From left to right: Original image (left), all detected edges of any class (middle left), edges belonging to real class (middle right) and the final detected triangles (right). The locations of local feature are marked with green crosses. Within each image: edges of different classes are colored differently.

feature pairs the triangle representation is highly distinctive. Moreover, due to the constraints on shape and eccentricity (Section 5.2.3) the detections provide a good localization of the logo inside the image.

5.4 Evaluation

In the following we describe the setup of our experiments and their results. For all training images we performed the matching of image pairs as described in Section 5.2 to derive the triangles for each image pair. Then we index all triangles in our cascaded index and let our system perform the logo detection. We tune the parameters using a validation set (subset P_2 of the FlickrLogos-32 dataset) and report the performance obtained on the test set (subset P_3).

5.4.1 Automatic Optimization for High Precision

Our proposed system has several parameters that can be tuned for performance. Many of them directly change the number of detections of point pairs and triples. Moreover, due to the different visual appearance of logos but also due to their typical placement within images, a varying number of local features is detected on logos of different brands. This directly influences the chance to find these logos by drawing random samples of local feature triples for searching within a logo database. For instance, logos of brands like “Coca-Cola”, “Corona”, “Becks” or “Guinness” are typically placed on tables or in front of people being photographed. In contrast, logos of brands like “FedEx”, “Shell” or “DHL” tend to be not in the main focus of the image and as a result these are often depicted rather small. As the number of local features in those logo regions may vary significantly and depends on the logo class, we adaptively select our threshold T_c for the decision whether a logo of a given class c is present or not.

An individual threshold per logo class addresses the issues mentioned above by controlling the sensitivity of our detector. Thus, we first run the detection on a validation set and determine class-specific thresholds: Once we obtained the raw detection counts on the validation set, we perform a parameter sweep over the threshold T_c and recompute precision and recall for each class separately. We determine the optimal threshold for each class by fixing the precision to 0.95 and selecting the corresponding minimal threshold where the system reaches the desired precision. In case one class does not reach 0.95 precision we select the threshold for the best precision obtained. Note that this sweep is performed efficiently by running the detection once on the validation set and counting the detected feature triples for each image. As we have the detection counts for each image we can determine the impact of the decision threshold by varying it without re-running the actual detection.

The final performance is then computed on the test set using those class-specific thresholds that have been refined on the validation set. In other words, in our experiments we choose our target precision of 0.95 first and then select the decision threshold. Since we fix precision only for the validation set, we also report the precision on the test set as it may differ.

Note that while we tune parameters defining the quantization of items in the cascaded index we implicitly tune the constraints of our indexing scheme to be restrictive or tolerant. If these constraints get less restrictive false positive detections increase. However, the precision is kept constantly high by the adaptive increase of the decision threshold if more noise is present. Thus, more noise on the validation set implicitly leads to decreasing recall on the test set.

5.4.2 Evaluation of Parameters

As initial parameters we chose parameter values that have been determined empirically and seemed a good starting point. Here we used a visual vocabulary of 2000 visual words derived from gray-scale images. The quantization is done with empirically determined bin sizes of 24° for the three feature orientations and 10° for δ_1 and δ_2.

In each experiment we optimize one parameter at a time and keep all others fixed. However, during all experiments, we re-learn the adaptive threshold for all logo classes and re-build the cascaded index for all shape and orientation parameter values. This assures that the evaluation is not biased towards certain parameter configuration

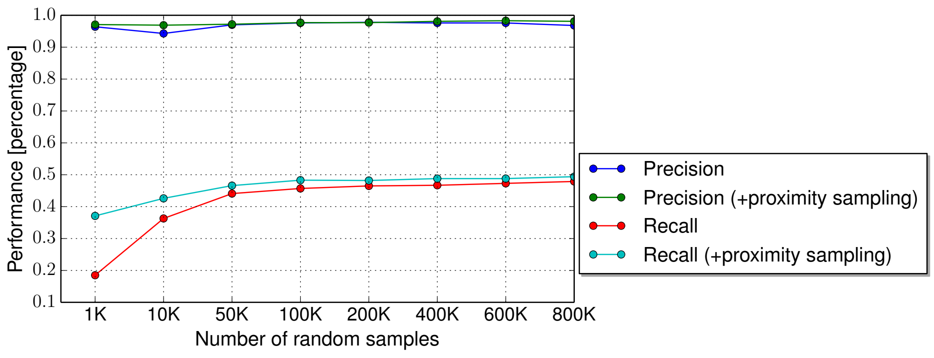


Figure 5.8: Precision and recall for varying numbers of random samples.

Monte Carlo Sampling Density As the recall of our approach is based on Monte Carlo sampling of points; one crucial parameter is the number of random samples, that is, the number of queries to the index. To increase recall, we can increase the density of the Monte Carlo sampling. Increasing the number of samples and thus queries will increase the chance to detect edges and triangles that occur both in query image and in the index. Note that the sampling can be adjusted at query time according to the application needs or time budgets.

Consequently, we evaluate the impact of varying the number of queries made to the cascaded index. As the number of queries instantly changes the number of detections we perform the parameter sweep as described in Section 5.4.1 on the validation set and compute the final performance on the test set. This assures that we compare the recall only for equivalently well performing systems (in terms of precision). In Figure 5.8 the results of two type of systems are shown. The first system only performs the Monte Carlo sampling, i.e., it only samples random edges. The second additionally performs the proximity sampling that uses the edges from spatial nearest neighbors as described in Section 5.3.2 as queries.

One can see from Figure 5.8 that recall consistently improves with an increasing number of random samples. The proximity sampling significantly improves recall when only a few thousand edges are tested by the Monte Carlo sampling. There is a major improvement of the latter when e.g., 100K samples are used instead of a few thousands but only little improvement since then. Thus, in the following we always perform both proximity sampling as well as Monte Carlo sampling of 100,000 random edges.

Visual Vocabularies An important part of the discriminative triangle representations is the distinctiveness of the visual words themselves. Therefore we compare different vocabulary sizes from 1000 to 4096 visual words, up to the maximum number of distinct labels we can pack into our 64-bit integer code. As we chose to use 12 bits to encode each visual word label, 4096 words are the largest possible vocabulary size. The remaining 28 bits are needed to encode the 5 angles of the relative orientations and the shape of the triangle. In addition to SIFT features extracted from gray-scale images we also compare their performance to color-SIFT features.

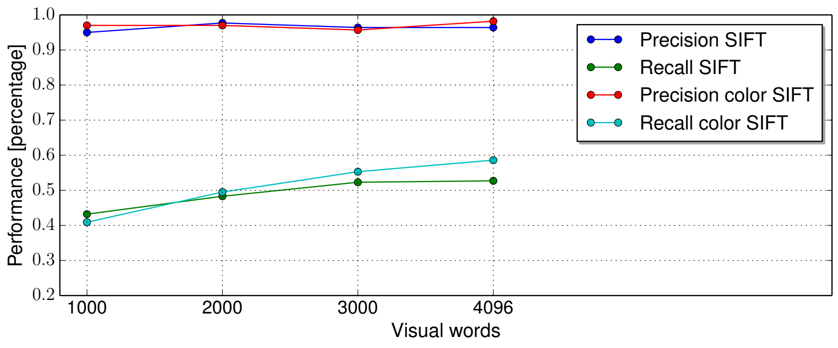


Figure 5.9: Performance for different visual vocabulary sizes of both gray-scale and color SIFT.

The clustering and the quantization of the descriptors introduce hard boundaries in feature space, thus a smaller vocabulary should be more robust to minor changes of descriptors but also possibly yield more false positives during detection. Note that in our setting an increasing amount of false positive detections on the validation set leads to smaller recall on the test set (see Section 5.4.1) as we keep the precision high by the automatic threshold selection strategy.

The result of this experiment is shown in Figure 5.9. One can see that larger vocabularies increase the performance. Most likely, this trend will further continue with increasing vocabulary sizes. However, longer signatures make efficient indexing in hash tables more difficult but may be explored in the future, though. In this work, we are constrained to 64-bit signatures. One can further observe that capturing color in descriptors further improves performance. Therefore the following experiments are performed with the largest color-SIFT vocabulary.

Quantization of Shape Representation and Feature Orientation In our representation the shape of a triangle is captured by two angles δ_1 and δ_2 that are quantized and packed into our 64-bit signature. Each of those is put into the first and second best bin during indexing to minimize loss of recall due to these quantization boundaries. We thus evaluate how the quantization of those angles affects the performance. From Figure 5.10 (left) we can see the best recall is obtained with a bin size of 15°.

The same experiment is repeated to test how the performance is affected if the quantization of the feature orientations changes. Figure 5.10 (right) shows the recall for different quantization bin sizes of feature orientations. There is no clear peak, but the best performance is achieved when the features orientations are quantized with a bin size of 40°. This in line with our experiences: While designing our system we observed that the orientations of features can change quite significantly depending on the view angle of the logo - much more than the shape of the corresponding triangles.

Larger and smaller quantization bin sizes of either shape or feature orientations lead to overly specific or coarse triangle signatures that then degrade recall. However, both graphs in Figure 5.10 show rather insignificant differences. We assume this might be caused by the

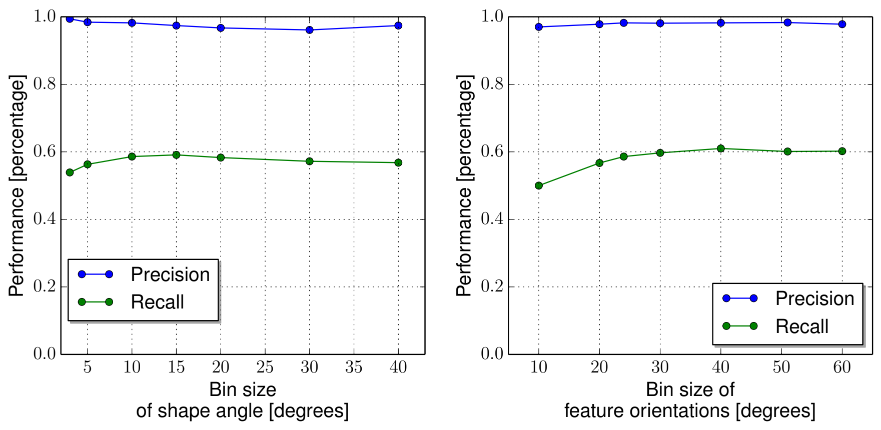


Figure 5.10: Performance for different bin sizes used for quantizing angles. **Left:** Quantization bin sizes for angles that capture the shape of the triangles. **Right:** Quantization bin sizes for relative feature orientations of the individual features.

extremely sparse triangle representation and the fact that its discriminativeness is bound by several components instead of a single component.

5.4.3 Logo Detection Results

After tuning the parameters of the system, our best system achieves a precision of 0.982 at a recall of 0.61 on the test set. The corresponding edge index holds about 613,000 keys and the triangle index holds about 11 million keys. This demonstrates that even without sophisticated post-processing the detection accuracy based on the occurrences of spatial configurations of local features can be kept high.

Finally, Figure 5.11 shows the confusion matrix for all 32 classes of this system, which further underlines the high precision of the logo recognition system. The results obtained clearly show that our system produces an extremely low number of false positives, resulting in high precision at the cost of recall. However, one can also observe that the detection of some logo classes works better than for others. This may be caused by many factors; among them the visual appearance i.e., the texture of a logo and its typical size in the photographed environment play a major role. With fewer local features and thus fewer triangles and edges in the index the chances of detecting a logo declines.

5.5 Summary

In this chapter we described a highly effective and scalable framework for recognizing logos in an image. Our approach is based on encoding and indexing the spatial configurations of local features in logo images. We use an automatic method for constructing a model for each

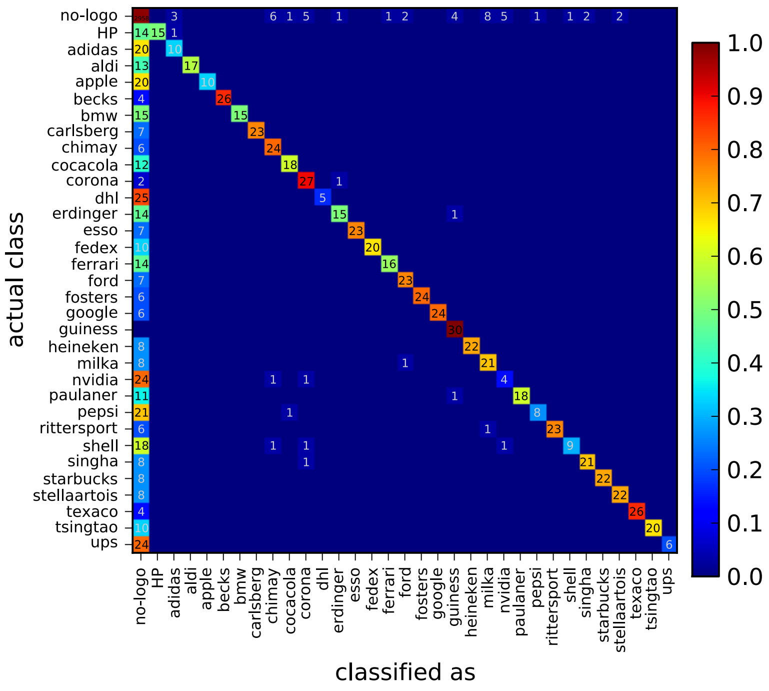


Figure 5.11: Confusion Matrix showing the true positive detections per class.

logo-class out of multiple training images, which significantly extends the ability to detect logos under varying conditions.

Our logo recognition system is inspired by the bag of visual words approach, but through embedding spatial knowledge into the index, we have shown that we effectively suppress false positive detections such that the resulting logo recognition has an extremely high precision of about 99%. We have tuned and tested our system against the novel FlickrLogos-32 benchmark, and found that we can effectively recognize the different logo classes with a high precision and good recall.

Though we demonstrated the detection of logos by counting the occurrences of highly distinctive visual feature triples, the underlying technique might be further exploited within an inverted index framework e.g., for high-precision near-duplicate search in images and videos.

6 Fast Spatial Re-ranking with 1P-WGC-RANSAC

In order to ensure that the top retrieved images correctly show the query object we employ a re-ranking step that ranks the retrieved images with respect to the spatial consistency of their local features to the query. Good practice for this purpose is to employ *Random Sample and Consensus* (RANSAC)-based methods as these cope well with false local feature correspondences.

We describe the underlying principles and discuss a RANSAC variant that uses single feature correspondences to estimate a transformation between two images (Philbin et al. 2007). The associated scales and dominant orientations of the two local features of a single correspondence is used to estimate a similarity transform. Evaluating all these correspondences makes this procedure deterministic, fast and robust to small inlier ratios.

In this chapter we present our 1-point-based WGC-constrained RANSAC variant (Romberg and Lienhart 2013b). We demonstrate that the spatial re-ranking can be considerably accelerated by omitting the projective re-estimation that is usually employed to refine the best k transformations with a fully projective re-estimation on the set of inliers. A further speed-up is obtained by imposing a weak geometric constraint. Correspondences violating this constraint are directly treated as outliers and evaluating the error function can be omitted.

In the end, our 1-point based WGC-constrained RANSAC outperforms the results in the literature. Especially for small vocabularies the re-ranking is significantly better in terms of mAP. Moreover, our approach achieves real-time performance and is even faster than the method of

Tolias and Avrithis (2011) designed specifically for fast re-ranking of retrieval results.

6.1 The Homography

A *homography* or projective transformation $\mathbf{H}$ is defined as the transformation that maps a point $\mathbf{x}$ that describes the real-world point $\mathbf{x}_\pi$ in one image to the point $\mathbf{x}'$ in another image describing the same real-world point $\mathbf{x}_\pi$ but from a different view. Both points $\mathbf{x}$ and $\mathbf{x}'$ lie on the image plane as they have no depth information (i.e., no z-coordinate) and are represented in homogeneous coordinates as $\mathbf{x} = (x, y, 1)^T$ and $\mathbf{x}' = (x', y', 1)^T$.

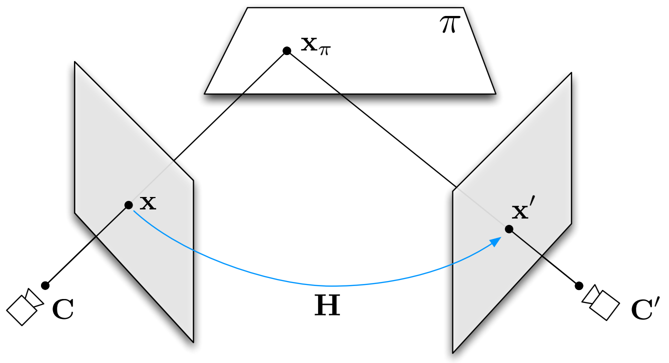


Figure 6.1: Homography model: The point $\mathbf{x}_\pi$ lying on the word plane π is mapped to the points $\mathbf{x}$ and $\mathbf{x}'$ in the image planes of cameras $\mathbf{C}$ and **C'**. The point correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$across two images is related by a homography $\mathbf{H}$. Adapted from Hartley and Zisserman (2003).

The 3-D point in world coordinates $\mathbf{x}_\pi$ is mapped to the point $\mathbf{x}$ in image 1 by a projection matrix $\mathbf{P}_1$ and a camera matrix $\mathbf{K}_1$:

$$\mathbf{x} = \mathbf{K}_1\mathbf{P}_1\mathbf{x}_\pi \tag{6.1}$$

In the same manner, the point $\mathbf{x}_\pi$ is mapped to the point $\mathbf{x}'$ in image 2 as

$$\mathbf{x}' = \mathbf{K}_2\mathbf{P}_2\mathbf{x}_\pi. \tag{6.2}$$

Thus, the two points $\mathbf{x}$ and $\mathbf{x}'$ are related by a chain of transformations as

$$\mathbf{x}' = \mathbf{K}_2\mathbf{P}_2\mathbf{P}_1^{-1}\mathbf{K}_1^{-1}\mathbf{x} = \mathbf{H}\mathbf{x}. \tag{6.3}$$

This relation between the two *measured* points in image coordinates $\mathbf{x}$ and $\mathbf{x}'$ is summarized in a single transformation matrix $\mathbf{H}$. The homography $\mathbf{H}$ describes the mapping of points in the image plane of image 1 to points in the image plane of image 2. Most important, the computation of such homography does not require the knowledge of the actual real world coordinates - it can be directly estimated from correspondences of points in the image planes.

The homography model requires that all correspondences consistent with a possible model must lie on a plane in the real world (see the plane π in Figure 6.1). Fortunately, the presented methods are robust against minor violations of this assumption, e.g., when correspondences arise from bumpy or uneven object surfaces.

In the following we describe techniques to directly solve for $\mathbf{H}$ without the explicit computation of the matrices $\mathbf{K}_2, \mathbf{P}_2, \mathbf{P}_1^{-1}$ and $\mathbf{K}_1^{-1}$. The matrix $\mathbf{H}$ itself is a 3×3 transformation matrix

$$\mathbf{H} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ p_1 & p_2 & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{p} & 1 \end{pmatrix}, \tag{6.4}$$

which can be either decomposed into or composed from a 2×2 affine transformation matrix $\mathbf{A} = \left(\begin{smallmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{smallmatrix}\right)$ that includes rotation, scaling and shearing, a translation vector $\mathbf{t} = (t_x, t_y)^T$ and a vector $\mathbf{p} = (p_1, p_2)$ describing the perspective projection. If $\mathbf{H}$ describes a purely affine transformation then $\mathbf{p} = \mathbf{0}$. As the homography operates on homogeneous coordinates it is only defined up to scale. Once the scale is fixed (see next section), the solution for the homography has then 8 degrees-of-freedom.

If $\mathbf{H}$ is known, the projection of $\mathbf{x}$ into image 2 (in homogeneous coordinates) can be computed by a matrix multiplication as in Equation 6.3. One can compute the image coordinates of the projected point $\mathbf{x}'$ in *non*-homogeneous coordinates directly as

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \tag{6.5}$$

$$y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}. \tag{6.6}$$

The remaining parts of this chapter deals with the homography estimation from point correspondences. These points correspondences are usually established automatically by determining visual feature correspondences across images.

6.2 Homography Estimation

6.2.1 Direct Linear Transform

The standard way to compute the homography from given point correspondences is the *Direct Linear Transform* (DLT) (Hartley and Zisserman 2003, pp. 88-110). At least four point correspondences are needed to estimate a full perspective transform between two images by solving a system of linear equations. To achieve this Equation 6.5 is reformulated:

$$h_{11}x + h_{12}y + h_{13} - h_{31}xx' - h_{32}yx' - h_{33}x' = 0 \tag{6.7}$$

One can see that Equation 6.7 is linear dependent on the unknowns h_{ij} and can be written as a homogeneous linear system. To write the former equations in matrix form the vector $\mathbf{h}$ is derived from $\mathbf{H}$ by flattening the homography matrix into a vector as

$$\mathbf{h} = (h_{11}, h_{12}, h_{13}, h_{21}, h_{22}, h_{23}, h_{31}, h_{32}, h_{33})^T. \tag{6.8}$$

Thus, Equation 6.7 can be written as dot product

$$(x, y, 1, 0, 0, 0, -xx', -yx', -x')\mathbf{h} = 0. \tag{6.9}$$

The same procedure for Equation 6.6 yields

$$(0, 0, 0, x, y, 1, -xy', -yy', -y')\ \mathbf{h} = 0. \tag{6.10}$$

As $\mathbf{H}$ is a transformation of points in homogeneous coordinates there are multiple equivalent solutions only differing by a scale factor. Therefore, once the scale is fixed, $\mathbf{h}$ can be described by 8 degrees of freedom. This can be done by fixing the last value of the transformation matrix $h_{33} = 1$, resulting in a inhomogeneous system of linear equations that can be solved with standard techniques such as Gaussian elimination. Alternatively, one may impose the constraint $||\mathbf{h}|| = 1$[1]. We will assume the latter approach in the following.

As 8 linear equations are needed to solve such a linear system and each point correspondence yields two equations, four point correspondences are sufficient to solve this system for a projective homography. The coefficient matrix $\mathbf{A}$ is built from these equations as

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x'_1 & -y_1x'_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y'_1 & -y_1y'_1 & -y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x'_2 & -y_2x'_2 & -x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y'_2 & -y_2y'_2 & -y'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3x'_3 & -y_3x'_3 & -x'_3 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3y'_3 & -y_3y'_3 & -y'_3 \\ x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4x'_4 & -y_4x'_4 & -x'_4 \\ 0 & 0 & 0 & x_4 & y_4 & 1 & -x_4y'_4 & -y_4y'_4 & -y'_4 \end{pmatrix} \mathbf{h} = \mathbf{0} \tag{6.11}$$

where x_1, y_1 ... x_4, y_4 denote the image coordinates of the four point correspondences. Note that, no three points of these correspondences must be collinear otherwise this linear system has no valid solution.

The homography $\mathbf{h}$ can be obtained by solving the linear system with Singular Value Decomposition (SVD). As the solution is the eigenvector with the smallest eigenvalue – itself part of a orthonormal basis – the constraint $||\mathbf{h}|| = 1$ is intrinsically satisfied by this particular method and also the trivial solution $\mathbf{h} = \mathbf{0}$ is prevented. Moreover, this approach can be further extended to exploit more than 4 point correspondences yielding more than 8 equations. The SVD then solves the resulting overdetermined linear system in least-square manner. Once $\mathbf{h}$ is known, $\mathbf{H}$ is obtained by reshaping the 1×9 vector $\mathbf{h}$ into a 3×3 matrix.

The estimation of the homography by solving $\mathbf{Ah} = \mathbf{0}$ effectively minimizes the *Algebraic*

[1]This has the advantage that solutions with $h_{33} = 0$ are possible. See Hartley and Zisserman (2003, p. 91).

Error. The drawback is (1) that the Algebraic Error does not have a geometric meaning and (2) it is also not invariant under certain transformations. For instance, the Algebraic Error of two solutions derived from point sets that only differ by a global rotation and translation will be different (Zhang 1997). Consequently, other error functions with real geometric meaning such as *Reprojection Error, Transfer Error, Symmetric Transfer Error* and *Sampson Error* can be used for a more accurate estimation of the homography (Hartley and Zisserman 2003, pp. 93-104) but this requires non-linear optimization. In this case the outcome of the Direct Linear Transform may be used as initialization, the homography is then subsequently optimized with more expensive iterative optimization techniques.

While the term homography usually denotes projective transformations only, the DLT algorithm can be easily adapted to solve for affine transformations as well. This requires only three point correspondences and simplifies the equations – eventually leading to a faster computation. As an advantageous side effect, in this case the Algebraic Error minimized by the DLT algorithm equals the geometric error (Hartley and Zisserman 2003, p. 96).

6.2.2 Normalization

Hartley and Zisserman (2003, pp. 107-110) further state that for numerical stability of the Direct Linear Transform it is mandatory that the point coordinates are normalized before the homography is estimated. Otherwise the entries in $\mathbf{A}$ are of different orders of magnitudes – e.g., due to the products in column 7 and 8 in Equation 6.11 – and the solution is numerically unstable. To address this issue Hartley and Zisserman strongly recommend to pre-process the point coordinates (x_i, y_i), (x'_i, y'_i) and normalize them such that the points are centered at $\mathbf{0}$ and the points have an average distance to their center of $\sqrt{2}$:

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \quad , \quad \overline{y} = \frac{1}{N}\sum_{i=1}^{N} y_i \quad , \quad d_{mean} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i-\overline{x})^2+(y_i-\overline{y})^2} \tag{6.12}$$

The normalized points are then given as $\tilde{\mathbf{x}} = \begin{pmatrix}\tilde{x}\\ \tilde{y}\end{pmatrix} = \begin{pmatrix}x-\overline{x}\\ y-\overline{y}\end{pmatrix}\frac{\sqrt{2}}{d_{mean}}$. This normalization is simply a shift followed by scaling and can be written as matrix $\mathbf{N}$:

$$\mathbf{N} = \begin{pmatrix}\frac{\sqrt{2}}{d_{mean}} & 0 & 0\\ 0 & \frac{\sqrt{2}}{d_{mean}} & 0\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & -\overline{x}\\ 0 & 1 & -\overline{y}\\ 0 & 0 & 1\end{pmatrix} = \begin{pmatrix}\frac{\sqrt{2}}{d_{mean}} & 0 & -\frac{\sqrt{2}}{d_{mean}}\overline{x}\\ 0 & \frac{\sqrt{2}}{d_{mean}} & -\frac{\sqrt{2}}{d_{mean}}\overline{y}\\ 0 & 0 & 1\end{pmatrix} \tag{6.13}$$

This procedure is done for the points $\mathbf{x}$ as well as the points $\mathbf{x}'$ such that the points in the two images are normalized independently, each with a different shift and scaling factor. The homography is then computed from point correspondences with normalized coordinates. The resulting homography describes the mapping between the points of the two images both in their own canonical coordinate system. To apply the obtained homography matrix to unnormalized image coordinates it must be denormalized subsequently. The steps are as follows:

1. Normalize all points $\mathbf{x}_i$, $\mathbf{x}'_i$, to $\tilde{\mathbf{x}}_i$, $\tilde{\mathbf{x}}'_i$, obtain $\mathbf{N}$, $\mathbf{N}'$.
2. Compute homography $\tilde{\mathbf{H}}$ from normalized points by applying the Direct Linear Transform.
3. Denormalize homography $\tilde{\mathbf{H}}$ to $\mathbf{H}$ with $\mathbf{H} = \mathbf{N}'^{-1}\tilde{\mathbf{H}}\mathbf{N}$.

6.2.3 Limitations

The DLT algorithm allows to estimate $\mathbf{H}$ from point correspondences assuming that *all* of the chosen point correspondences are correct. In practice, these correspondences are often determined automatically and many or even the majority of them are incorrect. Consequently, robust estimators are required to correctly estimate a homography in the presence of a high number of false correspondences.

The *breakdown point* describes the minimum percentage of bad data points that may completely spoil the result of an estimator. For example, in case of Least-Mean-Squares a single outlier may lead to an arbitrarily wrong estimation, thus its breakdown point is 0%. Least-Median-Of-Squares requires the majority of the data samples to be consistent with a possible model, resulting in a breakdown point of 50%. In the following we describe the RANSAC framework for robust estimators that has a much lower breakdown point. Its ability to handle outliers depends on the model and on the particular variant. In practice it is mostly only limited by the ability to verify a model as it requires a certain number of supporting data points to declare a model as valid.

6.3 Conventional RANSAC

Random Sample and Consensus (RANSAC) (Fischler and Bolles 1981) is a general framework for robust estimation of arbitrary models from noisy data. The underlying assumption is that among all data samples there are samples that are consistent with a possible model – termed *inliers* – and other samples that are not – termed *outliers*. Neither the actual model nor inliers and outliers are known in advance – most of the time not even their ratio. The objective of RANSAC is therefore to find the particular model that is consistent with the most samples. This implies that a robust estimation is needed that is capable of handling inliers and outliers appropriately. To achieve this, RANSAC requires (1) an estimator for the particular model type, (2) an appropriate error function and (3) a threshold on the error that determines if a certain sample is consistent with the current model or not.

In the following the term *samples* denotes the measurements – in our case these are point correspondences given by visual word correspondences and their locations. A *sample set* consists of m samples, i.e., m point correspondences. A random selection of sample sets therefore implies that multiple sample sets are drawn and each sample set in turn is determined by drawing m point correspondences at random from the set of all point correspondences.

The core of RANSAC is surprisingly simple and can be summarized by the following steps:

1. **Hypothesize:** A model hypothesis is estimated from randomly drawn sample sets. The cardinality of each sample set is minimal with respect to the model. For instance, when estimating affine transformations a single sample set consist of 3, in case of projective transformation of 4 randomly drawn points. These are the minimum number of points needed for an unambiguous estimation. Thus, a single sample set is sufficient to hypothesize a potential model. This procedure maximizes the chance that the sample set contains inliers only and is *uncontaminated*.
2. **Verify:** The hypothesis, i.e., the estimated model is verified by testing it against all samples. Samples that agree with the model are termed *inliers*, others *outliers*. In its simplest form the number of inliers is used as score indicating the quality of the model.
3. **Repeat:** This procedure is repeated and terminated either after a fixed number of iterations or once the likelihood of finding a better model becomes low. RANSAC then returns the model with the highest score. As now not only the model parameters but also the corresponding inliers are known, the model may be refined with other techniques based on the reduced but clean set of samples that are consistent with the model.

In the end RANSAC's objective is to find those model parameters $\theta^\star$ that minimize the error $\rho(\theta)$ over all samples i:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \sum_i \rho(\theta) \tag{6.14}$$

The minimization works by repeated sampling and estimation of candidate models followed by immediate verification. Due to its random nature this process must be repeated often. Given the inlier ratio $\epsilon = \frac{\#inliers}{\#inliers+\#outliers}$, the chance of obtaining an uncontaminated sample set of m randomly drawn points is given as ϵ^m. Therefore, the probability η of obtaining at least one uncontaminated sample set of size m within n repetitions is given as

$$\eta = 1 - (1 - \epsilon^m)^n. \tag{6.15}$$

Vice versa, the minimum number of RANSAC iterations n needed to ensure that at least a single uncontaminated set of samples is drawn with a probability $> \eta$ depends on the ratio of inliers to outliers ϵ:

$$n = \left\lceil \frac{\log(1-\eta)}{\log(1-\epsilon^m)} \right\rceil \tag{6.16}$$

While Equation 6.16 nicely guides in determining the number of iterations, unfortunately in practice the inlier ratio is often unknown. Besides that, one can see that with decreasing inlier ratio and by increasing the desired confidence the required number of iterations explodes.

There are several RANSAC variants that address this issue. While all of these variants vary significantly in their nature there are two common principles: (1) Faster hypothesis generation by better sampling: The aim is to increase the chance to draw an all-inlier sample early that will yield a good model. The increased likelihood in turn allows earlier stopping. (2) Faster hypotheses verification: The hypothesis is either verified on a subset or pre-tested before it is evaluated on all samples. Faster verification allows to test more hypotheses in the same time.

Despite of the former two principles it is of *significant* benefit to reduce the model complexity. Reducing the model complexity – e.g., constraining it from projective transformations to affine transformation – directly means a reduction of the sample cardinality m needed for an unambiguous estimation of a model. In the subsequent Section 6.5 we discuss our RANSAC variant based on 1-point correspondences (i.e., $m = 1$), which reduces the model complexity by several order of magnitudes. Given the inlier ratio ϵ, the probability of drawing an all inlier sample increases from ϵ^4 (projective transformation) or ϵ^3 (for an affine transformation) to plain ϵ.

6.4 Related Work

RANSAC has gained wide-spread popularity mostly because of its robustness to noisy data and is used in numerous applications. We would like to point out that RANSAC and other robust estimators such as Least-Median-of-Squares (Rousseeuw 1984) or MINPRAN (Stewart 1995) are closely related. In the following we focus on variants within the RANSAC framework as it is widely used in the context of homography estimation and highlight important works. Each of these variants addresses different shortcomings of a naive RANSAC implementation.

The choice of the cost function has significant impact on the robust estimator, thus various were investigated: Fischler and Bolles (1981) essentially used a binary "top-hat" scoring function: inliers induce no costs and outliers a constant penalty. Torr and Zisserman (2000) proposed the M-SAC estimator using a truncated error function that measures how good the data actually fits the model and penalizes outliers by a constant term. They extend this scheme towards MLESAC using a maximum likelihood estimate as error function, assuming Gaussian noise on the point correspondences. Lebeda et al. (2012) proposed the use of the truncated quadratic error, as it is more robust to the choice of the decision threshold than M-SAC at little computational costs.

Further improvements have been achieved by guided sampling instead of uniform sampling of the point correspondences: PROSAC (Chum and Matas 2005) speeds the estimation up by taking the individual quality of feature correspondences into account - increasing the chance to find good hypotheses early. NAPSAC (Myatt et al. 2002) assumes that inliers tend to be spatially close to each other and draws individual point correspondences from the same neighborhood when sampling. Ni et al. (2009) exploit by their GROUPSAC scheme that inlier correspondences are often naturally grouped. As each sample consists of multiple point correspondences, drawing the individual point correspondences from the same groups increases the chance of obtaining an all-inlier sample. Rodehorst and Hellwich (2006) formulate the model estimation as genetic algorithm (GASAC) that replaces the random sampling by an evolutionary strategy.

Increased robustness and speed is obtained by exploiting additional constraints: Márquez-Neila et al. (2008) check the spatial consistency of each hypothesis by determining the relative order of points. Sattler et al. (2009) discard point correspondences in advance that have insufficient support from other correspondences in their neighborhood. This scheme, termed SCRAMSAC, results in fewer correspondences, which in turn increases speed and also tends to increase the inlier ratio as well.

RANSAC has been further accelerated by randomizing not only the hypothesis generation but also the verification (Chum and Matas (2002), Capel (2005), Chum and Matas (2008a)). By employing preemptive tests on small randomly chosen subsets from the whole set of samples, poor hypotheses can be directly discarded. Only promising hypotheses are evaluated on the whole set of samples. Nistér (2003) developed a preemptive RANSAC scheme suited for real-time applications due its constant time-budget. Preemptive RANSAC selects the best hypothesis from a fixed number of samples and thus depends on a minimum inlier ratio to succeed.

Chum et al. (2003) proposed LO-RANSAC that performs a local optimization step as soon as it finds promising hypothesis. The local optimization aims to find an accurate model immediately once a promising hypothesis is found, eventually leading to shorter run times. Lebeda et al. (2012) extensively discuss this approach and further address several flaws of LO-RANSAC leading to lower computational costs in their LO-RANSAC$^+$ scheme.

Rabin and Delon (2010) address the problem that RANSAC commonly only selects the single best hypothesis and modify it to deal with multiple objects (MAC-RANSAC).

A comprehensive overview of several variants and implications as well as a unified view of the RANSAC framework is given in Raguram et al. (2013).

6.5 1P-WGC-RANSAC

Most works on improving RANSAC focus on cases where a relatively high number of inliers can be expected. In the literature often $\geq 25\%$ inliers are assumed. However, in image retrieval it is common practice to employ RANSAC as a post-retrieval verification step. Here, its input are visual word correspondences between the query image and a retrieved candidate image from an image database. The visual vocabulary may be large; as a result a relatively small number of correspondences between these images can be established – unlike when the full feature descriptors are used for matching. However, due to memory constraints, the descriptors themselves are usually never stored in image databases but represented by the corresponding visual words. As the re-ranking is by definition a post-retrieval step, a major concern is speed. Thus, re-computing descriptors is intractable and the re-ranking must operate on visual word correspondences. Unfortunately, even with these correspondences there are often many mismatches on background clutter – e.g. on foliage – such that in practice there are often very few inliers. For this challenging setting the traditional approach that draws four points to estimate a homography often fails or yields degenerated results.

In this section we describe a RANSAC variant based on the approach of Philbin et al. (2007) that uses single feature correspondences to estimate a transformation between two images. The associated scale and dominant orientation of the two local features of each correspondence is used to estimate a similarity transform, i.e., translation, rotation and uniform scaling.

6.5.1 Model generation

Given two local features $\mathbf{x}$ and $\mathbf{x}'$, their locations $(\mathbf{x}_x, \mathbf{x}_y), (\mathbf{x}'_x, \mathbf{x}'_y)$, their scales $\sigma(\mathbf{x})$, $\sigma(\mathbf{x}')$ and their dominant orientations $ori(\mathbf{x})$, $ori(\mathbf{x}')$ – as determined by the feature detector – we can derive the transformation in closed form from their correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$. The scaling f is determined by the scale ratio between the interest points $f = \frac{\sigma(\mathbf{x}')}{\sigma(\mathbf{x})}$ and the rotation angle is given as $\theta = angle\_diff(ori(\mathbf{x}_i), ori(\mathbf{x}'_i))$ (See Appendix B for details).

The hypothesized transformation is then given by a chain of transformations: The point $\mathbf{x}$ is shifted to point $\mathbf{x}'$ as denoted by the translation $\mathbf{T}(\mathbf{x}'-\mathbf{x})$, followed by scaling and rotation of θ degrees around $\mathbf{x}'$. The latter two induce themselves a chain of transformations given by the translation $\mathbf{T}(-\mathbf{x}')$ to the origin followed by rotation $\mathbf{R}(\theta)$ and scaling $\mathbf{S}(f)$ and finally the back-shifting $\mathbf{T}(\mathbf{x}')$. Here, $\mathbf{T}$, $\mathbf{R}$ and $\mathbf{S}$ denote the corresponding 3×3 transformation matrices. Putting it together, we can compute the hypothesized transformation $\mathbf{H}$ in closed form as

$$\mathbf{H} = \mathbf{T}(\mathbf{x}')\mathbf{S}(f)\mathbf{R}(\theta)\mathbf{T}(-\mathbf{x}')\mathbf{T}(\mathbf{x}'-\mathbf{x}) = \begin{pmatrix} a & b & -a\mathbf{x}_x - b\mathbf{x}_y + \mathbf{x}'_x \\ -b & a & b\mathbf{x}_x - a\mathbf{x}_y + \mathbf{x}'_y \\ 0 & 0 & 1 \end{pmatrix} \tag{6.17}$$

where for brevity a and b are substitutes with $a = f\cos(\theta)$, $b = f\sin(\theta)$.

The major benefit is that a single feature correspondence is sufficient to generate a hypothesis. Evaluating all these correspondences makes this procedure deterministic, fast and robust to small inlier ratios. To summarize, a RANSAC scheme based on 1-point-correspondences has the following advantages over the traditional 3- or 4-point RANSAC:

- The underlying model is simple. We chose to estimate translation, scaling and rotation. Such similarity transform has 4 degrees of freedom and only 3 if it is constrained to translation and scaling only. A single local feature correspondence carries enough information to derive a hypothesis for such a model. Due to their simplicity estimated models are less likely to be degenerated - unlike projectivities that have to be carefully checked for sanity.
- The estimation of each hypothesis can be done analytically. Given two local features the translation as well as scaling and rotation can be directly derived in closed form from the localization information determined by the feature detector. In conventional RANSAC one has to estimate $\mathbf{H}$ by solving a linear system as described in Section 6.2.1, which is much more costly.
- Each hypothesis is intrinsically either correct or incorrect as it is derived from a single

feature correspondence which itself is either correct or incorrect. In contrast, when sampling more than a single point correspondence one has to deal with those cases where only some of the randomly drawn correspondences are correct as well as the case where points are collinear. The former is solved by repeating the RANSAC steps multiple times, leading to an explosion of the number of iterations (see Equation 6.16). The latter requires additional checks before **H** is estimated.

- While RANSAC randomly samples correspondences and repeats this up to hundreds of thousands times, a 1-point-RANSAC only uses each correspondence for the estimation of a transformation once. Thus, the randomized sampling is turned into a *deterministic procedure*. The computational complexity for the estimation itself therefore linearly depends on the number of feature matches. Unrelated images that usually have a few (false) correspondences only, will therefore require very little computational effort.
- If desired, the homography estimation can be employed in a cascade-like manner: The 1-point RANSAC may be used to quickly filter the total set of samples and discard most outliers. In a subsequent step a fully perspective transformation with 8 degrees-of-freedom may then be estimated by a traditional method such as Least-Mean-Squares or Least-Median-of-Squares from the set of inliers only.

6.5.2 Error Function

Various error functions have been proposed for the robust estimation of homographies. We chose the Symmetric Transfer Error as it is symmetric and can be efficiently computed without requiring iterative refinement of **H** as other error functions do (Hartley and Zisserman 2003). The Transfer Error measures the Euclidean distance between a measured point $\mathbf{x}'_i$ and the projection of its counterpart $\mathbf{x}_i$ by the estimated homography given as $\mathbf{H}\mathbf{x}_i$. The Symmetric Transfer Error is defined as the Transfer Error induced both by **H** and its back-projection $\mathbf{H}^{-1}$ (see Figure 6.2 for an illustration). Here, $\mathbf{x}_i$ and $\mathbf{x}'_i$ denote the points of the i-th the correspondence $\mathbf{x}' \underset{i}{\leftrightarrow} \mathbf{x}$. Thus, the (squared) Symmetric Transfer Error for a single correspondence $\mathbf{x}' \underset{i}{\leftrightarrow} \mathbf{x}$ is given as

$$\varepsilon_i(\mathbf{H})^2 = d(\mathbf{H}\mathbf{x}_i, \mathbf{x}'_i)^2 + d(\mathbf{H}^{-1}\mathbf{x}'_i, \mathbf{x}_i)^2 \tag{6.18}$$

where $d(\mathbf{a}, \mathbf{b})$ denotes the Euclidean distance between the projections of the homogeneous points **a** and **b** to the image plane. To avoid the costly computation of the square root usually the squared error is computed only.

The symmetric transfer error is superior to the one-sided transfer error as it assumes that measurement errors of points occur in both images. The error minimized is symmetric, thus swapping the image pair does not change its output. It can also be shown that the minimization of the symmetric transfer error is invariant under similarity transforms in contrast to the algebraic error (Hartley and Zisserman 2003, p. 106). However, it implies that $\mathbf{H}^{-1}$ is known. Therefore, a conventional RANSAC not only has to estimate **H** but also needs to determine the

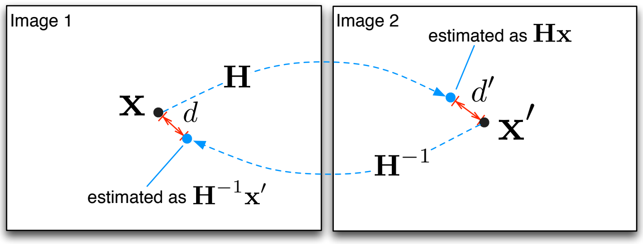


Image 1
Image 2
$\mathbf{H}$
$\mathbf{x}$
d
estimated as $\mathbf{H}^{-1}\mathbf{x}'$
estimated as $\mathbf{H}\mathbf{x}$
d'
$\mathbf{x}'$
$\mathbf{H}^{-1}$

Figure 6.2: The Symmetric Transfer Error of a single point correspondence: It measures the distances d and d' (red) between the real points $\mathbf{x}$ and $\mathbf{x}'$ (black dots) and the corresponding projections (blue dots). Adapted from Hartley and Zisserman (2003).

inverse $\mathbf{H}^{-1}$, which at least doubles its computational costs[1]. In contrast the 1-point-based RANSAC can easily derive the inverse analytically when estimating $\mathbf{H}$. Thus, this particular advantage has double impact.

6.5.3 Scoring

RANSAC searches for the homography $\mathbf{H}^*$ that minimizes the following objective function over all data samples i:

$$\mathbf{H}^\star = \operatorname*{argmin}_{\mathbf{H}} \sum_i \rho_i\big(\varepsilon_i(\mathbf{H})\big) \tag{6.19}$$

Thus, the error $\varepsilon_i(\mathbf{H})$ is not evaluated directly but by the indirection of a scoring function. Much work has been dedicated to the investigation of different scoring functions. The particular choice of the error function may have a significant impact on the overall performance, as it is critical for a robust estimation. We use the truncated quadratic cost function (Torr and Zisserman 1998, Lebeda et al. 2012) that has the same robustness as the MLESAC error function (Torr and Zisserman 2000) for large distances but uses a quadratic error for small distances:

$$\rho(d) = \begin{cases} d^2 & \text{if } d^2 < T^2 \\ T^2 & \text{otherwise} \end{cases} \tag{6.20}$$

The threshold T denotes the maximum distance inliers are allowed to deviate from the ideal projected point. If a correspondence has an error $\rho(d) < T$ it is treated as inlier otherwise as outlier. Thus, the truncated quadratic error (Equation 6.20) scores inliers depending on their distance to the ideal point while outliers get a constant penalty. The constant penalty is important for robust estimation. If outliers would get a penalty depending linearly or even quadratic on the distance to the ideal point, the error and thus the overall estimation may be significantly influenced by outliers, which is the contra-position of the original goal of RANSAC. Consequently, the truncation of the error function is an intrinsic requirement of RANSAC. For comparison, Least-Mean-Squares (LMS) minimizes the same objective function as in Equation 6.19 but with

[1]The computation of the inverse is more expensive than estimating $\mathbf{H}^{-1}$ from reversed correspondences.

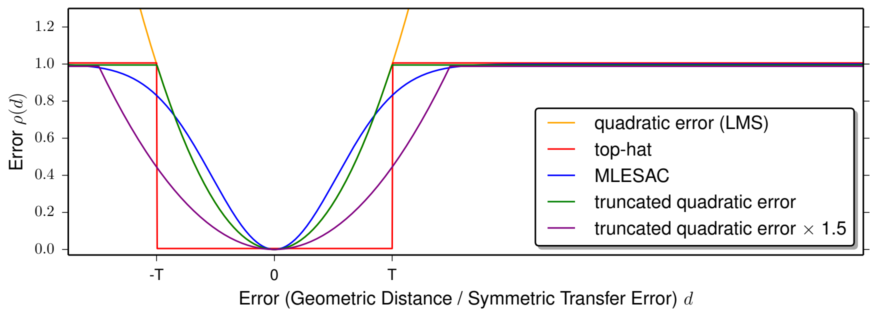


Figure 6.3: Various cost functions used for scoring the quality of estimated homographies. For illustration; in practice the error/distance d cannot be negative.

the non-truncated quadratic error $\rho(d) = d^2$.

In Figure 6.3 the truncated quadratic error function is compared to the traditional inlier counting scheme (inverse top-hat function) denoted as *top-hat* score and a MLESAC-like score function. Intuitively it is clear that the truncated quadratic function will avoid ties, that is different homographies with the same score (=number of inliers in this case) as determined by the top-hat function. In contrast to the MLESAC-error function the truncated quadratic has the further advantage that it can be directly computed from the quadratic error without taking the square root. The particular choice of the threshold T within the error function in Equation 6.20 is obviously critical for the performance of the application. Throughout the following experiments we use the truncated quadratic score function widened by a factor of 1.5 as suggested by Lebeda et al. (2012). The slightly relaxed decision allows a robust estimation of homographies across a wide range of settings.

However, without further care the threshold depends on the image resolution. To address this, we adopt the normalization technique as described in Section 6.2.2 for our 1P-WGC-RANSAC variant - even though numerical stability is not of concern. The representation of points in normalized image coordinates allows to choose the threshold independent of the image resolution.

6.5.4 Weak Geometric Consistency Constraint

It has been shown in various works that a weak constraint based on the geometric consistency of local feature matches improves retrieval (Jégou et al. 2009a). The intuition behind this constraint assumes that if an object undergoes rotation and scaling, it is likely that the local features captured on this object do change accordingly in orientation and scale. Such desirable behavior obviously depends on the capabilities and the quality of the feature detector, which must be both rotation- and scale-invariant in this case. In practice, the weak geometric consistency (WGC) assumption allows to discard false correspondences by checking if the local feature orientations and scale changes are consistent across images.

In this spirit, we introduce a WGC-constraint in our RANSAC scheme, which (1) immediately discards false local feature correspondences and (2) speeds up the error function as non-consistent feature matches are directly treated as outliers. Only correspondences from features with orientations and scales that are consistent with the estimated transformation may be scored as inliers. We found that this constraint has little impact on the quality of the re-ranking, it is neither significantly better nor worse. However, it acts as filtering that can be employed *before* the inliers are determined. If a feature correspondence violates the WGC constraint it is directly treated as outlier. Thus, the error function within the RANSAC framework is potentially speeded up as there is no need to compute the perspective mapping for these false correspondences.

It is important to note that such constraint based on the weak-geometric consistency is possible for all RANSAC variants. However, to derive the orientation and scale change for an arbitrary homography, a costly homography decomposition into scaling and rotation matrices is required. In contrast, our 1-point RANSAC directly obtains the rotation and scale factor derived from the feature correspondence being used to estimate $\mathbf{H}$.

WGC Constraint: A local feature correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$ is only consistent according to the WGC constraint if their induced rotation $\theta_{\mathbf{x}\leftrightarrow\mathbf{x}'} = \mathit{angle\_diff}(\mathit{ori}(\mathbf{x}), \mathit{ori}(\mathbf{x}'))$ and their induced scaling $f_{\mathbf{x}\leftrightarrow\mathbf{x}'} = \frac{\sigma(\mathbf{x}')}{\sigma(\mathbf{x})}$ is consistent with the estimated scaling $f_{\mathbf{H}}$ and rotation $\theta_{\mathbf{H}}$ of the currently evaluated hypothesis $\mathbf{H}$. That is, the correspondence is consistent iff $|\mathit{angle\_diff}(\theta_{\mathbf{x}\leftrightarrow\mathbf{x}'}, \theta_{\mathbf{H}})| < \tau_{angle}$ and $s_{min} f_{\mathbf{H}} < f_{\mathbf{x}\leftrightarrow\mathbf{x}'} < s_{max} f_{\mathbf{H}}$. If a correspondence violates this WGC-constraint, it is directly discarded as the participating local features do not reflect the estimated scaling or rotation of the image content.

In other words, the contribution to the overall error of the homography by a particular correspondence is only evaluated if the WGC-constraint holds. Otherwise we set the error of the correspondence directly to $\rho(\infty)$. The WGC-constraint accelerates RANSAC especially in case of many false correspondences and small inlier ratios - a desirable behavior for re-ranking search results on large datasets.

Throughout this work we used $\tau_{angle} = 30°$, $s_{min} = 0.5$ and $s_{max} = 2.0$ as these values gave empirically good results. Obviously, tighter thresholds potentially discard more false correspondences at the risk of losing inliers. We observed that the re-ranking with the proposed WGC-constrained RANSAC takes about 30% less time compared to a non-WGC-constrained RANSAC (see the Experiments in Section 6.6) as the re-ranking may be stopped earlier. Most important, it is also significantly more robust for small vocabularies as the WGC-constraint discards false visual word correspondences immediately.

6.5.5 Impact of Local Optimization

In our RANSAC scheme each feature correspondence yields a hypothesis for the transformation between images. The 10 best hypotheses with the lowest error are kept for further refinement. If a hypothesis has more than 15 inliers these are refined by a *local optimization (LO)* step: Given the set of almost outlier-free point correspondences determined by our 1P-WGC-RANSAC, a fully

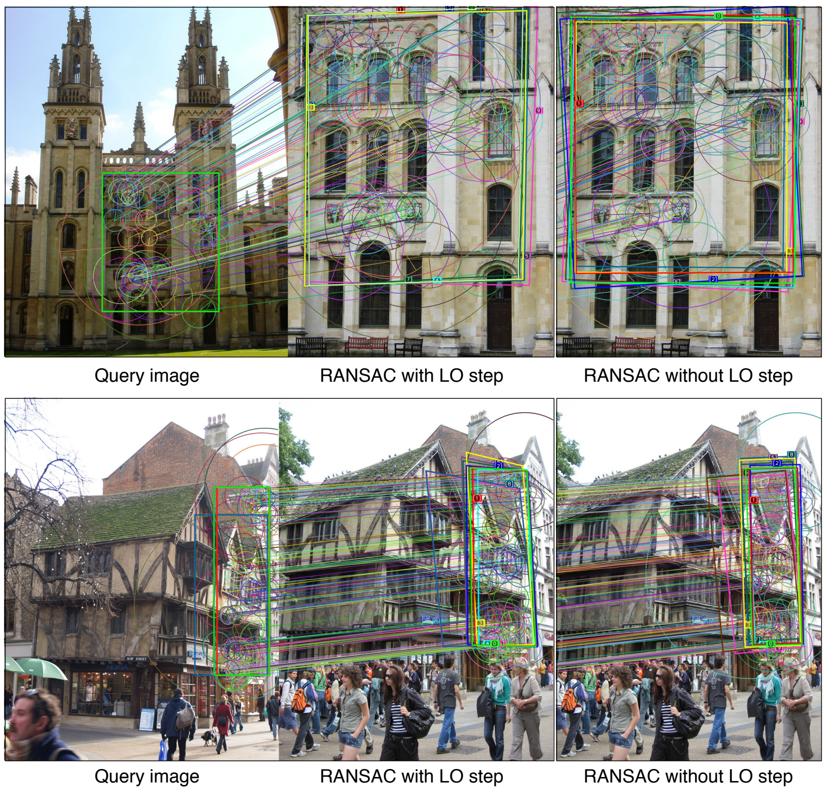


Figure 6.4: Comparison of estimated homographies with and without projective re-estimation. The left image shows the query image. The top 10 homography hypotheses are shown as colored rectangle (best hypothesis is green) determined by the bounding box of all inliers. The middle columns shows these hypotheses on the candidate image after projective re-estimation. The right column shows these hypotheses without projective re-estimation.

projective transformation between images is estimated via Least-Median-of-Squares (Rousseeuw 1984) and a subsequent non-linear minimization of the projection error via the Levenberg-Marquardt method. The final homography is that transformation that has the smallest error among all the old and refined transformations.

While RANSAC is in general considered as slow and costly this is not entirely true. In fact we found that most of the time was spent for projective re-estimation. Moreover, while this refinement improves the visual quality of the estimated transformation it has little effect on the induced ranking. To illustrate this, Figure 6.4 shows the top 10 hypotheses *without* projective re-estimation. The similarity of these hypotheses indicates that for re-ranking the re-estimation maybe omitted. Thus, we propose a new variant 1P-WGC-RANSAC *without* subsequent LO step that is much faster than a variant estimating a fully projective transformation between images. We argue that the projective re-estimation is necessary for applications such as 3-D reconstruction or panorama stitching.

However, the re-ranking of image search results does not benefit from a precise image registration. One reason is that the search itself is usually constrained to discrete visual words that are most likely less tolerant on viewpoint changes than the descriptors themselves. In addition, a precisely estimated homography can hardly be used to discard additional false positives. Thus, for our 1P-WGC-RANSAC variant we can omit the subsequent LO step. The re-ranking without projective refinement is much faster and without loss of quality in terms of mAP.

6.6 Evaluation

We compare our approach to that of Philbin et al. (2007) and Arandjelović and Zisserman (2012b) on the Oxford5K dataset (Philbin et al. 2007) following the respective test protocol: For each query, the top 1000 retrieval results are re-ranked with an early stop if 20 images in a row could not be verified successfully. Images are scored by the sum of the IDF weights of all inliers. Verified images (≥ 4 inliers) are placed above unverified images in the result list.

All search results were obtained with either SIFT or RootSIFT features, visual words from an approximate k-means vocabulary and tf-idf weighting. The results for our approach are shown in Table 6.1. Here, both "SP" and "RANSAC" denote that the subsequent spatial re-ranking of the former initial result list was performed. One can see that our implementation, using SIFT descriptors computed from Difference-of-Gaussian interest points[1], yields slightly higher (1M visual words) or even significantly higher scores (100K words) than that of Philbin et al. (2007) where SIFT descriptors computed from Hessian-affine interest points were used. Quite surprisingly, the performance after re-ranking with the smaller vocabulary of 100K words is close to the one with 1M words. This demonstrates that our proposed scheme is able to deal with a small vocabulary, its less discriminative correspondences and small inlier ratios.

When re-ranking the entire result list (see Table 6.1, bottom two sections) one can see a slight improvement over the re-ranking with early stop criterion. The highest score among the re-ranking results is a mAP of 0.753 obtained with a vocabulary of 100K visual words and WGC-constrained re-ranking. This stresses the importance of the WGC constraint and also shows that there are still a few relevant images located at the very end of the original result list.

We obtain similar and consistent results for our re-ranking approach on the FlickrLogos-32 dataset (see Table 6.2). Here, the retrieval by bag-of-words (with tf-idf and burstiness weighting, denoted as "tf-idf-sqrt") computed from RootSIFT features serves as baseline. This setup is described more extensively in Section 7.2.5 in the next chapter. Similar to the Oxford dataset we perform the re-ranking on the 200 top-ranked images with an early stop if 20 images in a row could not be verified successfully. The results demonstrate that the geometric re-ranking further improves the result as well. As for the Oxford dataset, the projective re-estimation does not improve the performance but does take more time. It further refines the homography but

[1] Note that for this comparison we trained the vocabulary on the Oxford dataset itself and used a magnifier of 9 for the region described by the SIFT descriptor (see Section 2.1.1). The relatively high scores – especially those of the large vocabularies – are definitely caused by overfitting the visual vocabulary on the Oxford dataset, which also holds for the results of Philbin et al. (2007) and Arandjelović and Zisserman (2012b).

	Method	Voc.	mAP	Time	#img	
SIFT	Philbin et al. (2007), bow	100K	0.535	–	–	
	Philbin et al. (2007), bow+SP	100K	0.597	–	–	
	bow, tf-idf, SIFT	100K	0.571	–	–	
	1P-RANSAC, +LO	100K	0.678	133s	47,442	
	1P-RANSAC,	100K	0.680	50s	47,442	
	1P-WGC-RANSAC, +LO	100K	0.692	115s	30,885	
	1P-WGC-RANSAC,	100K	0.691	38s	30,885	
	Philbin et al. (2007), bow	1M	0.618	–	–	
	Philbin et al. (2007), bow+SP	1M	0.645	–	–	
	Arandjelović and Zisserman (2012b) SIFT, bow	1M	0.636	–	–	
	Arandjelović and Zisserman (2012b) SIFT, bow+SP	1M	0.672	–	–	
	bow, tf-idf, SIFT	1M	0.647	–	–	
	1P-RANSAC, +LO	1M	0.712	47s	8,753	
	1P-RANSAC,	1M	0.711	11s	8,753	
	1P-WGC-RANSAC, +LO	1M	0.704	45s	5,501	
	1P-WGC-RANSAC,	1M	0.703	9s	5,501	
RootSIFT	Arandjelović and Zisserman (2012b) RootSIFT, bow	1M	0.683	–	–	
	Arandjelović and Zisserman (2012b) RootSIFT, bow+SP	1M	0.720	–	–	
	bow, tf-idf, RootSIFT	1M	0.675	–	–	
	1P-RANSAC, +LO	1M	0.729	52s	9,520	
	1P-RANSAC,	1M	0.729	11s	9,520	
	1P-WGC-RANSAC, +LO	1M	0.723	49s	5,813	
	1P-WGC-RANSAC ★	1M	0.723	10s	5,813	
	1P-RANSAC, no early stop: top 1000	1M	0.741	33s	54,537	No early stopping
	1P-WGC-RANSAC, no early stop: top 1000	1M	0.747	33s	54,537	
	1P-RANSAC, no early stop: all images	1M	0.742	90s	201,934	
	1P-WGC-RANSAC, no early stop: all images ▲	1M	0.749	90s	201,934	
	bow, tf-idf, RootSIFT	100K	0.610	–	–	
	1P-RANSAC, no early stop: top 1000	100K	0.721	54s	55,000	
	1P-WGC-RANSAC, no early stop: top 1000	100K	0.736	52s	55,000	
	1P-RANSAC, no early stop: all images	100K	0.733	141s	264,094	
	1P-WGC-RANSAC, no early stop: all images	100K	0.753	136s	264,094	

Table 6.1: Comparison of spatial re-ranking results for the Oxford5K dataset following the protocol of Philbin et al. (2007). The upper part shows results obtained with SIFT, the lower part these from RootSIFT. #*img.* denotes the number of image re-ranked summed over all queries.

is not able to discard additional false positives. We assume that a simple geometric verification based on 4 degrees-of-freedom is sufficient to filter out false positives. Unlike for the Oxford dataset the WGC constraint does not improve the re-ranking yet it accelerates it by allowing earlier stopping. Yet, when re-ranking the entire result list the WGC constraint slightly improves the results. This underlines that re-ranking does not require to estimate fully affine/projective homographies and due to its speed 1P-WGC-RANSAC is beneficial for spatial verification.

To measure the time we performed all the experiments for both datasets on the same machine (Intel Xeon X5550, see Appendix D for details) using a single thread for execution of our `C++` program. All timings in Tables 6.1 and 6.2 measure the wall time in seconds for re-ranking all the queries of the respective dataset. Each timing was obtained as the median of the wall time from 10 runs and captures I/O, the determination of the correspondences from pre-computed

Method	Voc.	mAP	Time	Images re-ranked
bow, tf-idf-sqrt, RootSIFT	100K	0.448	–	–
1P-RANSAC, +LO	100K	0.513	849s	169,273
1P-RANSAC	100K	0.513	295s	169,273
1P-WGC-RANSAC, +LO	100K	0.510	635s	129,251
1P-WGC-RANSAC	100K	0.510	257s	129,251
bow, tf-idf-sqrt, RootSIFT	1M	0.545	–	–
1P-RANSAC, +LO	1M	0.565	458s	89,579
1P-RANSAC	1M	0.565	110s	89,579
1P-WGC-RANSAC, +LO	1M	0.568	416s	55,226
1P-WGC-RANSAC	1M	0.568	90s	55,226
1P-RANSAC, no early stop: all images	1M	0.566	1264s	3,137,060
1P-WGC-RANSAC, no early stop: all images	1M	0.580	1258s	3,137,060

Table 6.2: Evaluation: Spatial re-ranking results for the 960 queries of the FlickrLogos-32 dataset. All retrieval and re-ranking results were obtained with RootSIFT descriptors.

visual words and RANSAC itself. In summary the 1-point-RANSAC without local optimization is between about 2.5 and 4.5 times faster than with the costly projective refinement. The WGC constraint further accelerates the re-ranking: it is slightly faster and allows to stop the re-ranking earlier once no more images can be verified. In practice, this yields another speed-up by a factor of 1.05 to 1.3.

The throughput of our 1P-WGC-RANSAC is high: For instance see the re-ranking run ★ in Table 6.1: Here, 5,813 images were re-ranked with a throughput of $\approx$ 606 images/s or a time consumption of about 1.6 ms per image. Note that we heavily optimized our application towards stream-like re-ranking. Thus re-ranking twice the number of images is much cheaper than performing the re-ranking for twice the number of queries. Furthermore, image with low rank in the original result list returned by bag-of-words tend to share fewer visual words with the query image. Thus, compared to the re-ranking mentioned above, re-ranking the *full* result set for every query does take longer but the average time per query actually decreases. For example, run ▲ in Table 6.1 takes 9 times longer than the formerly mentioned run but re-ranks 34 times more images with a throughput of $2,243$ images/s or a time consumption of about 0.4 ms. These results clearly indicate the strength of our re-ranking approach and its suitability for real-time applications.

We also would like to stress the fact that this RANSAC variant is actually much faster than other methods not based on RANSAC, which were explicitly designed for fast re-ranking. For example, Tolias and Avrithis (2011) propose a "speeded-up re-ranking" method based on hierarchical hough voting. They report a time consumption of roughly 7.7ms per image for the re-ranking on the Oxford dataset, almost five times higher than ours.

6.7 Summary

In this chapter we have presented a RANSAC variant that estimates similarity transforms between images based on 1-point correspondences. Using a single correspondence for hypothesis generation allows to enumerate all possible hypotheses resulting in a deterministic procedure.

First, we have shown that a projective re-estimation – after an initial estimation of a transformation with four degrees of freedom – does not improve the spatial re-ranking results. While this refinement improves the visual quality of the estimated projection, it does neither change the ranking significantly nor discards additional false positives. Thus, for geometric re-ranking a plain similarity transform is sufficient, which significantly accelerates the re-ranking procedure.

Second, we employed a weak geometric consistency constraint on the inliers that speeds up the hypotheses evaluation. Feature correspondences that do not undergo the same change in rotation and scale as the evaluated transformation are directly discarded without computing their contribution to the actual error of the projection. As the WGC constraint can be tested quickly the hypotheses evaluation is further accelerated.

We extensively compared our approach to existing approaches and demonstrated its state-of-the-art performance. Especially for small vocabularies our approach significantly outperforms that of Philbin et al. (2007) by a large margin. Moreover, our 1P-WGC-RANSAC is extremely fast, making it suitable for real-time applications.

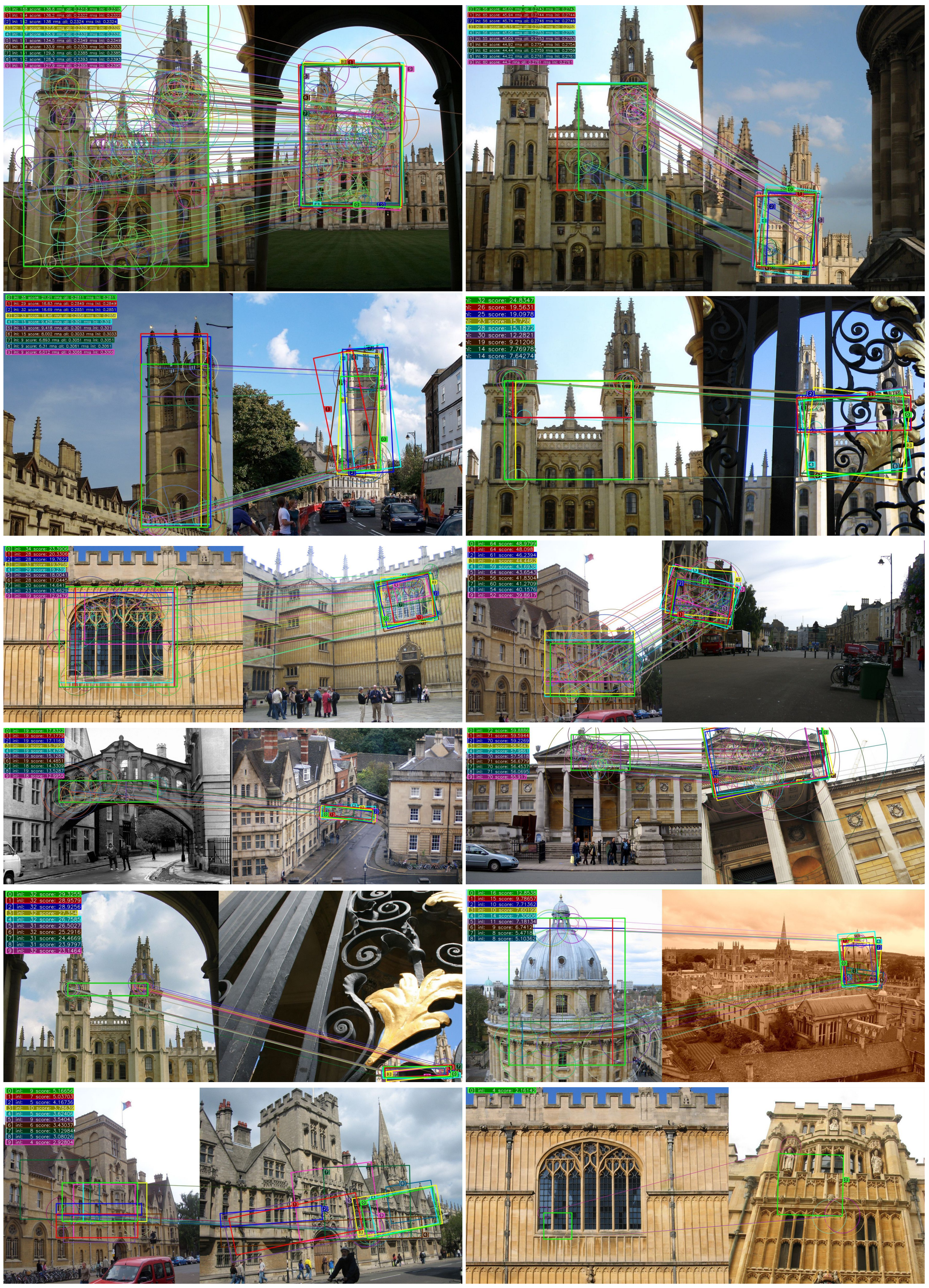


Figure 6.5: Several examples of estimated transformations with 1P-WGC-RANSAC on the Oxford dataset. The top 10 hypothesis are projected as colored rectangles into the images. The top-left corner shows the corresponding scores. **Top row**: Easy cases, the top 10 hypotheses almost converged and are hard to distinguish. **2nd to 5th row**: More challenging cases. There are more variations but still the top 10 hypotheses are in most cases quite similar. **Bottom row**: Failures. The hypotheses are invalid as the underlying buildings are different yet have similar structure.

7 Bundle min-Hashing

It has been observed several times that the retrieval performance of bag-of-words based methods improves much more by reducing the number of mismatching visual words than by reducing quantization artifacts. Popular incarnations of this insight are extremely large vocabularies of millions of visual words, or signatures such as Hamming Embedding (Jégou et al. 2009a) to discard false correspondences. In other words, the precision of the visual description seems to be more important than its recall, because low recall may be recovered by doing a second retrieval round, i.e., by query expansion.

In this chapter we present a robust feature bundling technique, previously described in Romberg et al. (2012), Romberg and Lienhart (2013a) and Romberg and Lienhart (2013b). It allows for approximate similarity search of sparse sets of visual words. It builds on visual words, but does not describe each visual word individually. Instead, it aggregates spatial neighboring visual words into feature bundles. A search technique for such bundles based on min-Hashing allows for similarity search without requiring exact matches. Compared to individual visual words these bundles carry more information such that fewer false positives are retrieved. This leads to a much smaller and cleaner result set.

We summarize our contributions as follows: We discuss and evaluate our retrieval technique based on feature bundles and extensively compare its performance to existing approaches on three different datasets, each of them representing a specific application scenario. In addition we demonstrate that the recall of a system targeting high precision for object retrieval can be increased by exploiting synthetically generated images in combination with query expansion and database augmentation. Finally, the former techniques are combined and exploited for scalable logo recognition in a system that significantly outperforms the current state-of-the-art.

The remainder of this chapter is organized as follows: Section 7.1 describes related work relevant in the context of this paper. The proposed feature bundling approach is described and evaluated extensively in Section 7.2. The method for boosting recall by generating artificial variants of images is presented in Section 7.3. In Section 7.4 we describe our logo detection system that exploits the former techniques, followed by the conclusions in Section 7.5.

7.1 Related Work

In this section we briefly survey the existing literature on image and object retrieval. The related work on logo retrieval already presented in Section 5.1 also applies here. We focus on retrieval techniques and further highlight the related work relevant in the context of min-Hashing.

Visual Words and Bundling We would like to emphasize the fact, that in contrast to our work most existing approaches to feature bundling are indeed post-retrieval verification steps where the internal geometry of a bundle is used to discard false correspondences *after* retrieval.

An early approach in this spirit by Sivic and Zisserman (2003) exploited the number of matching neighboring features to discriminate true feature matches from random matches. Wu et al. (2009) proposed to bundle multiple SIFT features that lie in the same MSER region into a single description. However, this work uses individual visual words for retrieving candidate images, the bundles are only used together with a weak geometric similarity criterion for post-retrieval verification. Cao et al. (2010) propose to learn the most informative projections that map the visual words from the 2-D space into feature histograms termed "spatial bag-of-words". Jégou et al. (2009c) present a similar approach, which is more unbiased to certain image layouts. Here, the original feature histograms is split by random projections into multiple smaller "mini bag-of-features". The most similar images in an image database are determined by separate lookups and an aggregating scoring. Zhang et al. (2009) mine descriptive visual phrases by analyzing the local neighborhood of local features in order to obtain a more discriminative visual description than single visual words.

Min-Hashing (mH) Min-Hashing is a locality-sensitive hashing technique for approximate similarity search of sparse sets. Originally developed by Broder (1997) for the detection of duplicate text documents, Chum et al. (2007) adopted it for the visual domain. Here, min-Hashing was used for near-duplicate image detection. Later, Chum et al. (2008) extended it further to the approximation of weighted set overlap as well as histogram intersection. In each of these settings an image is modeled as a sparse set of visual word occurrences. As the average number of visual words per image is much smaller than the vocabulary size for large vocabularies, the resulting feature histograms are sparse and are converted to sets representing whether a visual word is present or not. Min-Hashing then allows to perform a nearest-neighbor search among all such sparse sets within an image database. This approach is described more detailed in Section 7.2.1.

Geometric min-Hashing (GmH) A conceptually similar approach to ours is Geometric min-Hashing of Chum et al. (2009). However, its statistical preconditions regarding the sparsity of feature sets are totally different to our setting. There are two major differences: (1) GmH samples central features by min-Hash functions from the set of all features of an image. Thus, there is no guarantee that a small object is actually captured by the visual description. (2) For each randomly drawn central feature the local neighborhood is described by a single sketch that incorporates the central feature as well as the min-Hashes of the neighboring features. In the end, this makes GmH very memory efficient, but not suitable for generic image retrieval because of low recall. Consequently, the authors use it to quickly retrieve images from a large database in order to build initial clusters of highly similar images as in (Chum and Matas 2010). These clusters are used as “seeds”; each member image is then used as query for a traditional image search to find more cluster members that could not be retrieved by GmH.

Partition min-Hashing (PmH) Lee et al. (2010) introduce a scheme that partitions the image into regular grid cells. Unlike for normal min-Hashing, the min-Hashes and sketches are computed for each partition independently. The search proceeds by determining the sketch collisions for each of the partitions. This procedure is conceptually similar to a sliding window search as partitions may overlap and are processed step by step. The authors show that Partition min-Hashing is significantly faster than standard min-Hashing and also has identical collision probabilities for sketches as min-Hashing in the worst case, but theoretically better recall and precision if the duplicate image region only covers a small area. However, in our experiments we found that Partition min-Hashing is not significantly better than min-Hashing on different datasets.

7.2 Bundle min-Hashing

We build our bundling technique on min-Hashing mainly for two reasons: (1) Feature bundles can be naturally represented as sparse sets and (2) min-Hashing does not imply a strict ordering of the participating local features or a hard matching criterion to find similar bundles – such as requiring identical visual words for all participating local features. Due to image noise, viewpoint and lighting changes, the individual local features, their detection, and their quantization are rather unstable and vary across images. If a strict matching criterion like the identity of two sets was used, a single addition or deletion of a local feature within a bundle would yield mismatching descriptions. We therefore exploit the min-Hashing scheme for the description of feature bundles as it allows to search for similar, yet not identical bundles.

To summarize, Bundle min-Hashing is an efficient approximate search method for images that have similar bundles. It has higher memory requirements than pure near-duplicate search methods, but at the same order of magnitude as bag-of-words. Most important, its search accuracy is close to bag-of-words, but with orders of magnitudes lower response ratio and much higher precision.

7.2.1 Min-Hashing

Min-Hashing (mH) is a locality-sensitive hashing technique that allows for approximate similarity search of sparse sets (Broder 1997). It models an image as a sparse set of visual word occurrences. As the average number of visual words per image is much smaller than the vocabulary size for large vocabularies, the resulting feature histograms are sparse and are converted to binary histograms. These binary histograms are compactly represented by the set of visual words of an image – describing which visual words are present at least once.

If a linear search over all sets in a database was feasible, the overlap $ovr(I_1, I_2)$ of two such sets I_1 and I_2 – also known as *Jaccard similarity* – could be used to determine the similarity of those sets. The overlap is given as the intersection of these sets over the union of those:

$$ovr(I_1, I_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}. \tag{7.1}$$

For instance, a linear search could search for sets in a database and rank them according to the overlap. However, in most cases a linear search over a database is infeasible. Fortunately, the min-Hashing scheme provides an efficient way to index these sets based on this overlap criterion. Given the set $I = \{v_1, ..., v_l\}$ of visual words of an image I, the min-Hash function is defined as

$$mh(I) = \underset{v_i \in I}{\operatorname{argmin}}\ h(v_i) \tag{7.2}$$

where h is a hash function that maps each visual word v_i *deterministically* to a random value from a uniform distribution. Thus, the min-Hash mh itself is a visual word, namely that word that yields the minimum hash value – hence the name min-Hash. The probability that a min-Hash function mh will have the same value for two different sets I_1 and I_2 is equal to the overlap of the two sets I_1 and I_2:

$$P(mh(I_1) = mh(I_2)) = ovr(I_1, I_2) = \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|} \tag{7.3}$$

Note that an individual min-Hash value not only represents a randomly drawn word that is part of the set, but each min-Hash also implicitly "describes" the words that are *not* present and would have generated a smaller hash - because otherwise it would have been a different min-Hash value.

The approximate search for similar sets is then performed by finding sets that share min-Hashes. As single min-Hashes yield true matches as well as many false positives or random collisions, multiple min-Hashes are grouped into k-tuples, called *sketches*. This aggregation

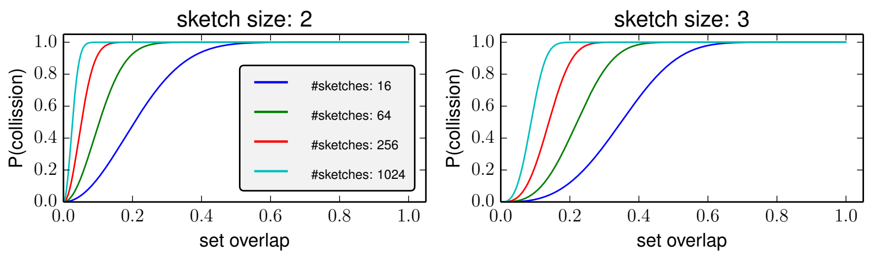


Figure 7.1: Collision probabilities in min-Hashing with sketches of size 2 (left) and 3 (right).

increases precision drastically at the cost of recall. To improve recall, this process is repeated n times. Thus, independently drawn min-Hashes are grouped into n tuples of length k. The probability of two different sets having at least one of these n sketches in common is then given by

$$P(collision) = 1 - (1 - ovr(I_1, I_2)^k)^n. \tag{7.4}$$

This probability function depends on the set overlap and is shown for two exemplary sketch sizes k and varying number of sketches n in Figure 7.1. In practice the overlap between non-near-duplicate images that still show the same object is small. In fact, the average overlap for a large number of partial near-duplicate images was reported to be 0.019 in Lee et al. (2010). This clearly shows that for applications which target the retrieval of partial-near-duplicates e.g., visually similar objects rather than (pure) near-duplicates, the most important part of that probability function is the behavior close to 0.

The indexing of sets and the approximate search are performed as follows: To index sets their corresponding sketches are inserted into hash tables by hashing the sketches itself into hash keys. This turns the exact search for a part of the set (the sketch) into a simple lookup. To retrieve the sets similar to a query set, one computes the corresponding sketches and searches for the sets in the database that have one or more sketches in common with the query. A lookup of each query sketch determines whether this sketch is present in the hash table, which we denote as "collision" in the following.

The lookups can be done efficiently in constant time as hash table offer access in amortized $\mathcal{O}(1)$. If there is a query sketch of size k that collides with a sketch in the hash table, then the similarity of their originating sets is guaranteed to be > 0, because at least k of the min-Hash functions agreed. To avoid collisions resulting from unrelated i.e., independent min-Hash functions, the sketches are put into separate hash tables: the m-th sketch is inserted into the m-th hash table ($m \in \{1, ..., n\}$).

7.2.2 Bundle min-Hashing

The idea of our bundling technique is simple: We describe the neighborhoods around local features by bundles, which aggregate the visual word labels of the corresponding visual features.

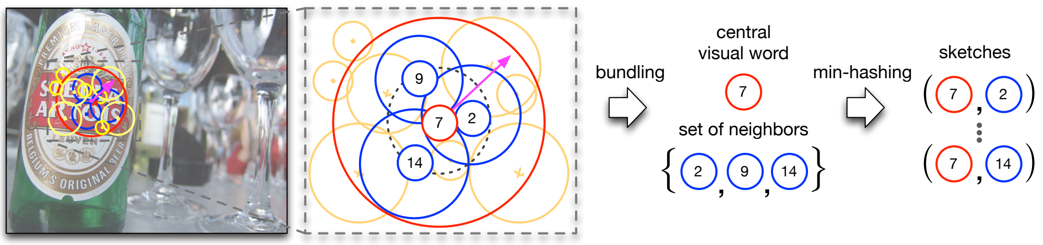


Figure 7.2: Bundle min-Hashing: The neighborhood around a local feature, the *central feature* (red), is described by a feature bundle. Features that are too far away or on scales too different from that of the central feature are ignored during the bundling (yellow). The features included in such a bundle (blue) are represented as set of visual word occurrences and indexed by min-Hashing (see Section 7.2.2).

The bundling starts by selecting *central features*, i.e., all features in an image with a sufficient number of local features in their neighborhood. Analogous to the feature histogram of a full image, the small neighborhood surrounding each central feature represents a "micro-bag-of-words". Such a bag-of-words vector will be extremely sparse because only a fraction of all features in the image is present in that particular neighborhood. Since the features of a bundle are spatially close to each other, they are likely to describe the same object or region of interest.

More specifically, given a feature $\mathbf{x}_i$ its corresponding feature bundle $b(\mathbf{x}_i)$ is defined as the set of spatially close features for a given feature $\mathbf{x}_i$:

$$b(\mathbf{x_i}) = \{\mathbf{x}_j | \mathbf{x}_j \in N(\mathbf{x}_i)\} \tag{7.5}$$

where $N(\mathbf{x}_i)$ is the *neighborhood* of feature $\mathbf{x}_i$, which is described at the end of this section. We further assume that for all features $\mathbf{x}_i$ in an image the descriptor vectors have been quantized to the corresponding visual words $v_i = q(\mathbf{x}_i)$.

The bundle $b(\mathbf{x}_i)$ is then represented by the corresponding set of visual words of all features included in that bundle:

$$W_i(b(\mathbf{x}_i)) = \{\ q(\mathbf{x}_j) \mid \mathbf{x}_j \in b(\mathbf{x}_i)\} \tag{7.6}$$

The resulting set W_i is then subsequently indexed by regular min-Hashing and represents a "micro"-bag-of-words as it only contains very few items.

In extensive experiments we observed the following: First, sketches of size 2 perform much better than larger sketches. Second, we found that the performance increases drastically if the first sketch element is not determined by min-Hashing but rather set to the visual word of the central feature itself. That is, for each bundle the m-th sketch is given as 2-tuple

$$(v_i,\ mh_m(W_i(b(\mathbf{x}_i)))\ ) \tag{7.7}$$

where v_i denotes the visual word label of the central feature and mh_m denotes the min-Hash returned by the m-th min-Hash function from the set of all visual words W_i present in bundle

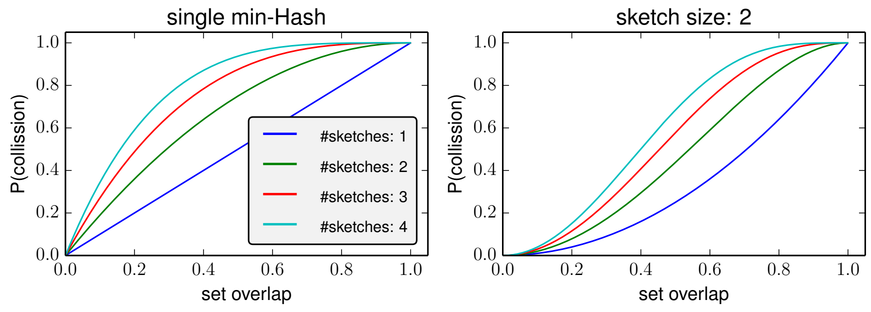


Figure 7.3: Collision probabilities given the set overlap between bundles. **Left**: Collision probability of a single min-Hash as used by Bundle min-Hashing. **Right**: Collision probability of sketches of size 2.

$b(\mathbf{x_i})$. The full process is illustrated in Figure 7.2.

The major advantage can be seen when comparing the collision probabilities of a single min-Hash and sketches of size 2 as shown in Figure 7.3. With our approach (using the central feature plus a single min-Hash) two bundles that have an overlap of only 0.2 of their corresponding visual word sets, have a 59% chance that one of 4 sketches collide (see Figure 7.3, left plot). This means, while there are multiple feature bundles that need to be described, each with several sketches, only very few sketches are needed per bundle to achieve a high probability to retrieve similar sets. This keeps the memory requirements for the indexing low. Further redundancy is added as images contain multiple bundles that may overlap. If some bundles do not match (collide) across images, there is the chance that other bundles in the same images collide.

Bundling Strategy Just like Geometric min-Hashing (Chum et al. 2009) or NAPSAC (Myatt et al. 2002) we assume that local features, which are spatially close to each other likely describe the same object. Therefore, our bundling strategy $N(\mathbf{x}_i)$ is as follows: Given a central feature we bundle it with its direct spatial neighbors. We require that at least two other features are present in its neighborhood and that these must be on a similar scale (see Figure 7.4). This is in line with the observation that true feature correspondences are often at the same scale (Jégou et al. 2009a).

Thus, each feature that is closer to a given central feature $\mathbf{x}_i$ than a given cut-off radius r_{max} is included in the respective bundle $b(\mathbf{x}_i)$: The radius r_{max} is chosen relative to the scale or patch size of the central feature s_i. The minimum and maximum scales s_{min} and s_{max} control the scale band considered for determining the neighbors relative to the scale of the central feature. Figure 7.2 illustrated the bundling criterion for $s_{min} = 0.5$, $s_{max} = 2.0$ and $r_{max} = 1.0$ (red circle, in this case equivalent to the radius of the central feature).

This bundling strategy has the following implications: First, the number of bundles is bounded by the number of local features within an image. Moreover, the bundling process ignores features that have no spatial neighbors within their neighborhood. This effectively

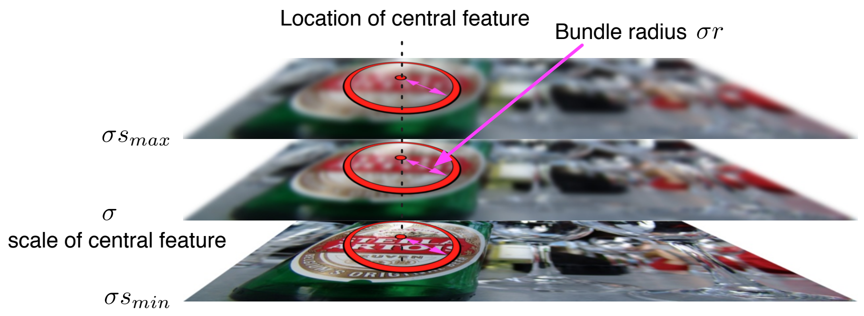


Figure 7.4: A bundle aggregates those features within radius σr and within the scale range $[\sigma s_{min}, \sigma s_{max}]$ as given by the location and scale σ of the central feature.

decreases the number of bundles below the number of features in an image. In turn fewer sketches need to be stored in the hash tables resulting in smaller memory consumption.

Bundling Implementation The features within a certain distance to a central feature can be efficiently determined by orthogonal range search techniques, which allow sub-linear search. In our application we used a 3-dimensional kd-tree to index local features by their location and scale. For bundling we determine all features that are within a "cylinder" given by the radius and the scale interval (see Figure 7.4).

Min-Hash Functions A min-Hash function is a mapping of a visual word $v \in \mathbb{N}$ to a hash value $h(v) \in \mathbb{R}$. For statistical correctness, the min-Hashes and thus the hash functions must be independent from each other. However, computing the min-Hashes may be expensive, especially as min-Hashing requires several hundreds to thousands of sketches and therefore min-Hashes. In early publications (Chum et al. 2007; 2008) the min-Hash values were re-used: individual min-Hashes were part of multiple different sketches. Addressing the efficiency, Chum and Matas (2012) recently proposed a technique of computing min-Hashes in advance and assigning them to all images that contain the respective min-Hash word at the same time.

One possibility to compute min-Hashes is by generating multiple random permutations on the range of all visual words and storing them in a lookup table. Given such permutation the hash for each visual word in a set could be obtained by a simple lookup. However, for large vocabularies and thousands of hash functions this lookup table is large. For instance, the permutation table for a vocabulary of $1M$ visual words and 1000 hash functions would require approx. 3.7 GB of memory – rendering this method rather undesirable for min-Hashing. While our particular Bundle min-Hashing variant only requires up to 4 hash function the lookup tables are still larger than CPU caches and lookups get slow.

Therefore, we use randomizing hash functions instead of precomputed permutation tables to compute the hashes (see Appendix C for details). These hash functions return a uniformly drawn random value deterministically determined by the given visual word and a seed that is kept fixed. This implementation is both substantially more memory efficient and faster than lookup tables. There is no guarantee that all visual words will produce different values but this approximation seems “good enough” while being much faster than a lookup table. We use this memory-friendly way to compute min-Hashes for all methods that use min-Hashing despite that for Bundle min-Hashing the permutation table could be pre-computed as only a very small number of permutations is needed.

Adjustable Search The representation of bundles – and also of min-Hash based methods in general – by multiple independent sketches has an advantageous side-effect: it facilitates a search tunable from high precision to high recall that can be adjusted *at query time* without post-retrieval steps or redundant indexing. Once bundles have been indexed with n sketches per bundle, the strictness of the search may be changed by varying the number of sketches at query time from $1...n$. As the m-th sketch was inserted into the m-th hash table, querying sketches from $1...m$ will yield only bundles were the corresponding sketches and hash functions in tables $1...m$ agreed at least once. As the sketch collision probability is proportional to the set overlap, bundles that have a high overlap with the query will be retrieved earlier than bundles with smaller overlap. Thus, by varying the number of query sketches one can adjust the strictness of the search. This can be seen from Table 7.1: with an increasing number of sketches used for querying the mean precision mP decreases and the corresponding mean recall mR increases.

7.2.3 Ranking and Filtering

Once the images which share similar bundles with the query are determined, they may be ranked by their similarity to the query. One possibility is to compute a similarity based on the number of matching bundles between these images. However, it is difficult to devise a meaningful similarity measure since the number of bundles found in images is usually very small. In preliminary experiments we evaluated several ways to compute a similarity score between query and retrieved images, based on the number of sketch collisions or number of matching bundles, either as absolute or normalized values in various ways. It turns out that the simple absolute count of sketch collisions was always on par with more complex similarity measures between images and their bundles.

However, a ranking based on the cosine similarity between the full bag-of-words histograms of the query image and the retrieved images performs significantly better than a ranking based on the sketch collision counts, as it is difficult to derive a good measure for image similarity based on a few collisions only. Thus, in our experiments we rank all retrieval results by the cosine similarity between the bag-of-words histograms describing the full images.

In other words, the retrieval by feature bundles is effectively a filtering step: The bundles are used to quickly fetch a small set of images that are very likely relevant. These images are then ranked by the cosine similarity between bag-of-words histograms obtained with a vocabulary of 1M words (see Section 7.2.5). We also address the problem of visual word burstiness by taking the square root of each tf-idf histogram entry as proposed by Jégou et al. (2009b). This is crucial for logo recognition as logos often contain text and text-like elements, which are known to be prone to yield repeated visual words termed "visual words bursts".

7.2.4 Influence of Parameters

For techniques based on min-Hashing there are several parameters that influence the collision probability of sketches and therefore the performance of our bundling approach. In the following we describe common trade-offs.

Visual Vocabulary The quantization of high-dimensional descriptors to discrete visual words is a lossy quantization. The vocabulary size usually depends on the application, i.e., there is no simple rule of thumb to choose the number of visual words. For near-duplicate retrieval but also for the retrieval of similar objects it has been shown that large vocabularies are beneficial despite the larger quantization error. This suggests that for retrieval it is more important to suppress false correspondences than obtaining a large number of tentative correspondences.

Sketch Size The number of min-Hashes that are aggregated into k-tuples directly control the collision probability. With a larger sketch size the collision probability of random collisions decreases drastically but also leads to lower recall. In practice mostly sketches of size 2 and 3 are used (Chum et al. 2008; 2009, Lee et al. 2010) as larger sizes have impracticable low recall and a single min-Hash just represents a single visual word. For Bundle min-Hashing we use sketches of size 2 but set the first component to the value of the central visual words.

Number of Sketches In contrast to the sketch size, increasing the number of sketches increases the collision probability. In other works, a few dozen up to a few thousand sketches are used depending on the representation. In our work, we compute multiple bundles for multiple features in the image and therefore we want to minimize the memory needed to store them. We observed that 2 sketches are sufficient for reasonable performance and 3 or 4 sketches slightly improve it further.

Locality The features that are eventually bundled into a single description are sampled from a region which size depends on the central feature. Thus, the region size implicitly influences the number of features and therefore the "noise ratio" when min-Hashes are computed for that region. Intuitively, features close to each other (e.g., with overlap) are correlated, while features lying far from each other may be treated as approximately independent. In the same manner the used interest point detector also has a major influence. Detectors that fire on blob-like regions usually yield more distributed interest points than a corner detector that fires on every peak of corner-like structures.

7.2.5 Experiments

In the following we describe the setup of our experiments and their results. As we focus on the retrieval of small objects we test and optimize our approach on the FlickrLogos-32 dataset (Romberg et al. 2011). Then we further demonstrate its performance on the Uk-Bench dataset (Nistér and Stewénius 2006) consisting of near-duplicate images and the Oxford dataset (Philbin et al. 2007) targeting object retrieval.

Visual Features

For our experiments we used SIFT descriptors computed from interest points found by the Difference-of-Gaussian (DoG) detector. For the evaluation on FlickrLogos dataset, as well as for our logo recognition system (Section 7.4) we used the improved RootSIFT descriptors. On the Oxford and UkBench dataset we used the traditional SIFT descriptors – unless mentioned otherwise – in order to make our results comparable to those in the literature.

We used an interest point detector that yields circular feature patches. If the bundling scheme was used with features from affine covariant regions, the elliptic regions determine the features to be included into a bundle.

The bundling parameters we show are tuned for a particular detector (DoG) and therefore for its detection characteristics. It is likely that bundling parameters need to be specifically adapted to the employed interest point detector as each detector varies in the number of detections, the distribution of interest points (e.g., blob-like, corner-like) and the behavior of the non-maximum-suppression. Therefore one has to adjust the bundling parameter to the sparsity of the neighborhoods depending on the interest point detector.

For the visual vocabulary creation we used approximate k-means (see Section 2.2.3). Vocabularies and IDF weights were computed from the training and validation set of FlickrLogos-32.

Evaluation

Bag-of-words First we compare the performance of various approaches based purely on the cosine similarity between bag-of-words on the FlickrLogos-32 dataset. Thus, we evaluate the retrieval performance of a plain bag-of-words search with varying vocabularies and varying patch sizes of the descriptors. We are especially interested in the impact of extremely large visual vocabularies on the performance. Thus, we vary the vocabularies from 10,000 (10K) to 4,000,000 (4M) words.

The results are shown in Figure 7.5. In a previous work (Romberg et al. 2012) we have already shown that IDF-weighting is always beneficial in the bag-of-words framework, even for large vocabularies greater than 1 million words. Thus tf-idf weighting was used in all cases. As found in prior works, large vocabularies show significantly better performance. The peak is consistently at 500K/1M words.

The patch size that is described by a SIFT descriptor linearly depends on the scale and a

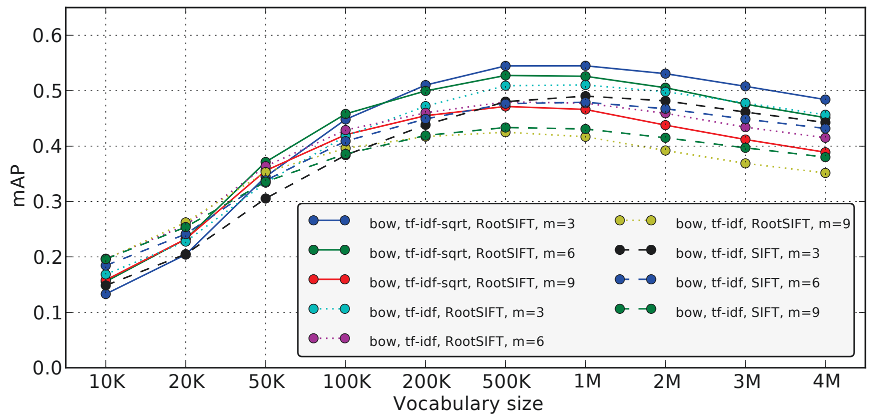


Figure 7.5: Retrieval score (mAP) for several bag-of-words variants on the FlickrLogos-32 dataset.

magnification factor m. We further test how this magnifier changes the performance. Larger patches carry more discriminative information but are less repeatable and also unable to describe very small objects. This might be the reason why the standard magnification factor of 3 performs best. In the following we perform all experiments with features computed with this specific magnification factor.

In addition we compare the performance of bag-of-words based on standard SIFT with that of the relatively new RootSIFT variant (Arandjelović and Zisserman 2012b). Clearly, the bag-of-words based on RootSIFT consistently outperforms the SIFT-based bag-of-words. Finally, we compare the tf-idf weighing to the burstiness measure proposed in Jégou et al. (2009b) where the square root is taken for each element of the tf-idf weighted histogram (denoted as "tf-idf-sqrt"in Figure 7.5). This burstiness measure further improves the retrieval performance as it down-weights repeating and thus less informative visual words ("bursts").

For further experiments on FlickrLogos-32 we therefore use visual words computed from RootSIFT descriptors and re-rank the results retrieved by feature bundles by the cosine similarity between bag-of-words histograms with square-rooted tf-idf weights. In order to make our results comparable to others in literature we use regular SIFT descriptors for evaluating on UkBench and Oxford and omit the burstiness measure (plain tf-idf instead). In all cases the best-performing vocabulary of 1M words is used for re-ranking, disregarding which vocabulary was used when building the feature bundles.

Feature Bundles We evaluate the performance of our bundling strategy with regards to mAP and response ratio and compare it to a retrieval with bag-of-words and tf-idf weighting.

In order to find the best bundle configurations we have performed extensive evaluations on the parameters of the bundle configuration. Due to limited space, we cannot show a detailed evaluation for all parameters. Instead, we report several well-performing bundle configurations (with respect to mAP) in Table 7.1. For further experiments we use the bundle configuration with $s_{min} = 0.5$, $s_{max} = 2.0$ and $r_{max} = 1.0$ as it has the best mAP and moreover a smaller memory footprint than other configurations.

Similar to bag-of-words the bundles profit from large vocabularies, but the peak is consistently across configurations at $200K$-$500K$ words. Most important, the bundles roughly have equal performance as bag-of-words, but have a two orders of magnitude lower response ratio (RR) as shown in Table 7.1 and also in Figure 7.6.

Note that we re-rank the result lists determined by bundle min-Hashing by the cosine similarity as given by the bag-of-words model. As the bundling is by definition only able to find correspondences between images that share visual words, the result set of the retrieval by feature bundles is a true subset of the result set obtained with bag-of-words retrieval. This clearly demonstrates the discriminative power of feature bundles for efficient filtering before more expensive post-retrieval steps are applied to the result set.

#sketches	s_{min}	s_{max}	r_{max}	Voc.	mAP	AvgTop4	mP	mR	RR	⌀#bundles	rel. storage
bag-of-words, tf-idf-sqrt weighting				200K	0.510	2.88	0.010	**0.952**	0.912	2468.1 words	100 %
bag-of-words, tf-idf-sqrt weighting				500K	0.545	3.06	0.010	0.932	0.845	2468.1 words	100 %
bag-of-words, tf-idf-sqrt weighting				1M	0.545	**3.16**	0.011	0.911	0.763	2468.1 words	100 %
4	0.5	2.0	1.0	200K	**0.554**	3.14	0.239	0.639	0.025	1640.9	266 %
3	0.5	2.0	1.0	200K	0.545	3.13	0.265	0.623	0.022	1640.9	199 %
2	0.5	2.0	1.0	200K	0.527	3.07	0.312	0.592	0.018	1640.9	133 %
1	0.5	2.0	1.0	200K	0.479	3.04	**0.396**	0.520	**0.012**	1640.9	**66 %**
4	0.5	2.0	1.0	1M	0.506	3.21	0.534	0.545	0.010	1640.9	266 %
3	0.5	2.0	1.0	1M	0.493	3.20	0.569	0.528	0.009	1640.9	199 %
2	0.5	2.0	1.0	1M	0.463	3.17	0.614	0.491	0.007	1640.9	133 %
1	0.5	2.0	1.0	1M	0.408	3.09	0.698	0.426	0.006	1640.9	66 %
4	0.7	1.42	1.5	200K	0.549	3.13	0.208	0.642	0.029	1826.7	296 %
3	0.7	1.42	1.5	200K	0.539	3.11	0.232	0.624	0.025	1826.7	222 %
2	0.7	1.42	1.5	200K	0.522	3.07	0.272	0.592	0.020	1826.7	148 %
1	0.7	1.42	1.5	200K	0.468	3.01	0.355	0.512	0.013	1826.7	74 %
4	0.7	1.42	1.5	1M	0.502	3.16	0.498	0.544	0.010	1826.8	296 %
3	0.7	1.42	1.5	1M	0.487	3.16	0.533	0.523	0.009	1826.8	222 %
2	0.7	1.42	1.5	1M	0.456	3.14	0.578	0.484	0.008	1826.8	148 %
1	0.7	1.42	1.5	1M	0.397	3.06	0.666	0.414	0.006	1826.8	74 %
4	0.7	1.42	2.0	200K	0.546	3.12	0.179	0.647	0.034	2104.5	341 %
3	0.7	1.42	2.0	200K	0.537	3.11	0.202	0.627	0.029	2104.5	256 %
2	0.7	1.42	2.0	200K	0.516	3.07	0.240	0.590	0.023	2104.5	171 %
1	0.7	1.42	2.0	200K	0.456	3.01	0.317	0.501	0.015	2104.5	85 %
4	0.5	2.0	2.0	200K	0.538	3.12	0.161	0.645	0.037	2255.8	366 %
3	0.5	2.0	2.0	200K	0.524	3.08	0.182	0.617	0.032	2255.8	274 %
2	0.5	2.0	2.0	200K	0.496	3.04	0.216	0.571	0.025	2255.8	183 %
1	0.5	2.0	2.0	200K	0.432	2.97	0.289	0.475	0.015	2255.8	91 %
4	0.5	1.0	1.0	200K	0.532	3.14	0.367	0.587	0.015	1125.9	182 %
3	0.5	1.0	1.0	200K	0.519	3.11	0.402	0.568	0.013	1125.9	137 %
2	0.5	1.0	1.0	200K	0.492	3.08	0.454	0.530	0.011	1125.9	91 %
1	0.5	1.0	1.0	200K	0.437	2.99	0.555	0.461	0.008	1125.9	46 %

Table 7.1: Comparison of bag-of-words retrieval with Bundle min-Hashing on the FlickrLogos-32 dataset: The upper part shows the scores of three different bag-of-words retrieval runs. The remaining rows show selected well-performing bundle configurations for 1, 2, 3 and 4 sketches per bundle. The columns s_{min}, s_{max}, r_{max} and Voc. denote the bundling parameters (as described in Section 7.2.2) and the vocabulary size. The scores follow in the order of mean Average Precision, average top 4 score, mean precision, mean recall and response ratio. The column *⌀#bundles* denotes the average number of bundles per image. The last column estimates the memory consumption and shows the number of hash table entries (sketches) relative to the number of visual words per image. The top-most bundle configuration was used in the remainder of this chapter.

Min-Hash, Partition min-Hash, Geometric min-Hashing We extensively compare our approach to min-Hashing (mH) as well as Partition min-Hash (PmH) and Geometric min-Hashing (GmH) on three different datasets. While these approaches were originally specifically meant for (partial) near-duplicate image search, this comparison shows how well these methods are suited for object retrieval or logo retrieval when used with typical parameters. For all experiments the sketch size was set to 2; n denotes the number of sketches. In case of Partition min-Hash 4×4 and 10×10 denote 16 and 100 overlapping partitions whereas np denotes the number of sketches per partition. The overlap was set 50% in all runs. For Geometric min-Hashing we follow the setup of Chum et al. (2009).

As already mentioned, we re-rank each preliminary result set of all approaches by the cosine similarity (see Section 7.2.3). We would like to point out that min-Hash, Partition min-Hash as well as Geometric min-Hashing *significantly* benefit from this. Bundle min-Hashing benefits as well but the effect is less pronounced.

In Figure 7.6 the results for the previously selected Bundle min-Hashing configuration is compared to the former approaches and bag-of-words. For retrieval of near-duplicate images there is little difference between most approaches. However, for object search on Oxford and on FlickrLogos-32 the differences are pronounced. Bag-of-words has high scores in every settings at the cost of a huge response ratio i.e., a single query still retrieves 80%+ of the whole database. On the other hand, min-Hashing, Partition min-Hash and Geometric min-Hashing have a desirable low response ratio but also low recall which in turn leads to low mAP. In contrast to the former min-Hash-based approaches (with high precision and low recall) and bag-of-words (with low precision and high recall) Bundle min-Hashing seems as an intermediate approach combining the best of both worlds: It has low response ratio, high precision and also high mAP.

Speed The computational performance of Bundle min-Hashing of indexing or searching depends linearly on the number of sketches and the number of local features in that image. The overall application speed seems mostly I/O and memory bound. In fact, choosing the right hash table implementation has a major impact on the overall performance.

To determine the performance of Bundle min-Hashing in terms of speed, we measured the wall time (single-threaded) of our C++ application on a machine with Intel Xeon X5550 CPU (see Appendix D for details). The timings include all operations such as bundling, min-Hashing and insertion into hash tables. This explicitly includes all I/O operations but excludes feature computation and quantization. For instance, with the configuration with $s_{min} = 0.5$, $s_{max} = 2.0$ and $r_{max} = 1.0$ and 4 sketches, indexing of the FlickrLogos-32 dataset (4280 images) takes about 96.8s (≈23ms per image) while processing the 960 queries takes about 13.4s (≈14ms per image).

Scalability We further test how the retrieval is affected once 100,000 distractor images – randomly chosen Flickr images – are added to the database (denoted as "+100K" in Figure 7.6). For this scenario we used the more memory-conservative scheme with 2 sketches per bundle that performs almost as good as 4 sketches per bundle but requires only half the hash tables.

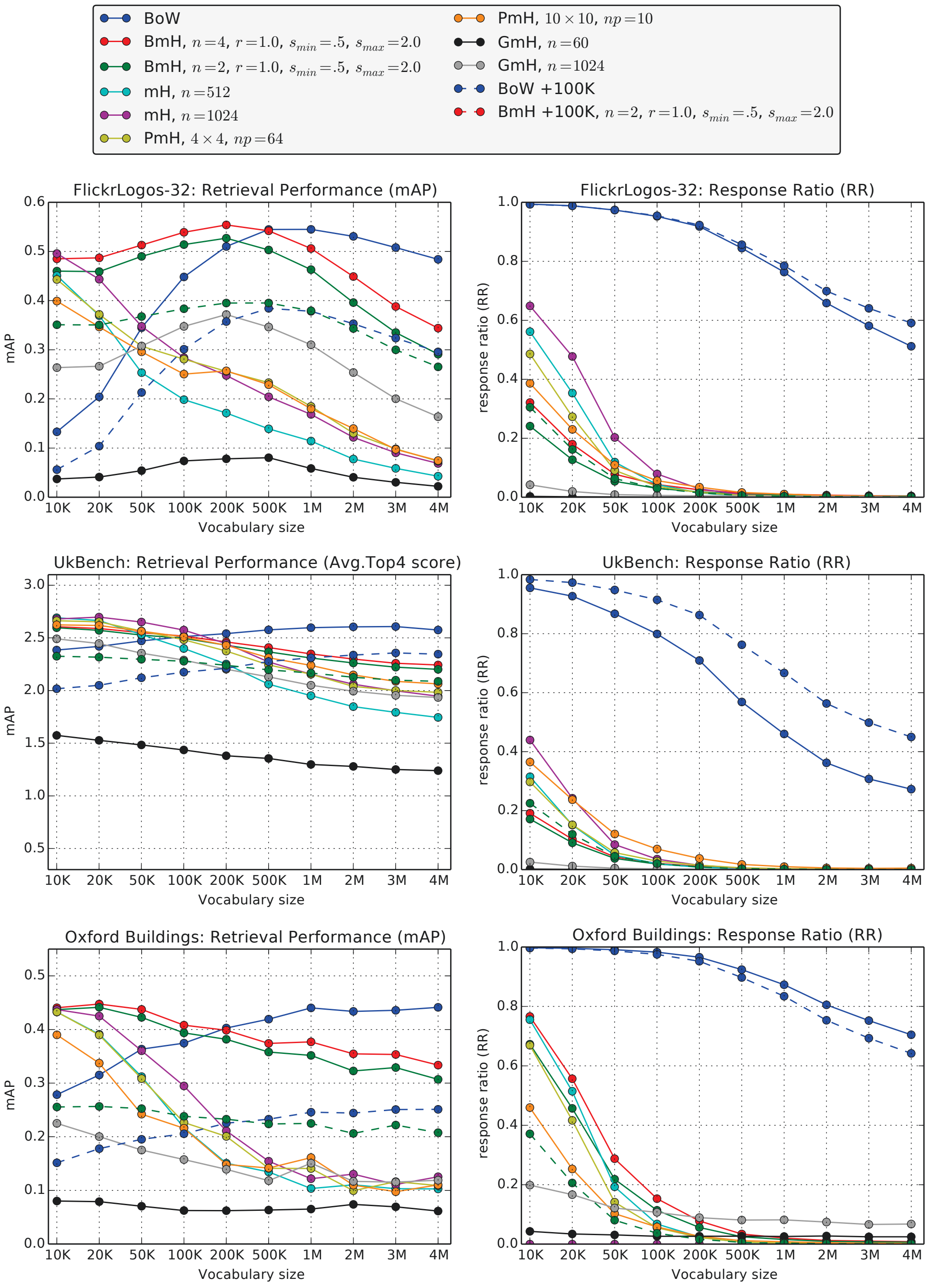


Figure 7.6: From top to bottom: Retrieval results on FlickrLogos-32, UkBench and the Oxford buildings dataset. The left column shows the mAP, the right columns shows the corresponding response ratio of each approach. The performance of Bundle min-Hashing (BmH) is on par with bag-of-words (BoW) and outperforms min-Hashing (mH), Partition min-Hashing (PmH) and Geometric min-Hashing (GmH). Its response ratio is an order of magnitude lower than bag-of-words and comparable to the former min-Hash-based methods.

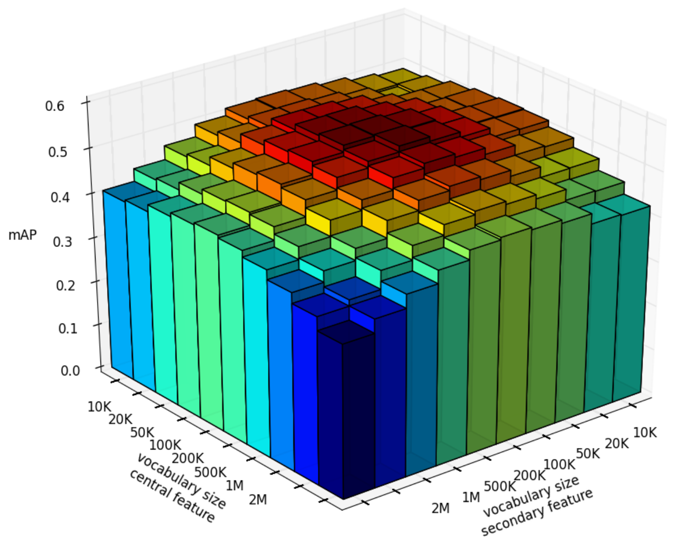


Figure 7.7: Impact of varying the vocabulary sizes when quantizing central and secondary features with different visual vocabularies.

Once distractor images are added to the database and its size increases, one can see a drop in performance consistently for of all retrieval methods. This behavior is well-known and has been shown several times in the literature before (Philbin et al. 2007, Jégou et al. 2009a). As in previous works we also observe a consistent drop of performance for both bag-of-words and Bundle min-Hashing across all datasets. However, Bundle min-Hashing seems less affected, especially because Bundle min-Hashing with two sketches per bundle already outperforms or is on par with bag-of-words. This suggests – in line with Zhang and Chen (2009) – that higher-order descriptions that combine multiple features into their representation, and as such Bundle min-Hashing as well, are less affected than first-order descriptions like bag-of-words once the database grows. The visual representation seems more distinctive and thus deteriorates slower.

Varying Vocabularies There are popular object recognition schemes that model objects by a root template and several part templates at twice the resolution of the root template. Inspired by these, we test whether we can improve the bundle description in a similar manner. In that sense, we treat our central features as root features and the secondary features within the neighborhoods as part features. Instead of varying the image resolution, we use different vocabulary sizes to encode the visual appearance of central and secondary features.

We evaluate the retrieval performance with our best-performing bundle configuration when using varying vocabularies from 10K to 2M visual words to quantize central and secondary features. The results on the FlickrLogos-32 dataset are shown in Figure 7.7. One can see a slight improvement when combining vocabularies of 200,000 and 500,000 visual words instead of a single vocabulary. However, there is no clear preference for a larger vocabulary for the central vs. the secondary features and the improvements are rather small. As the quantization

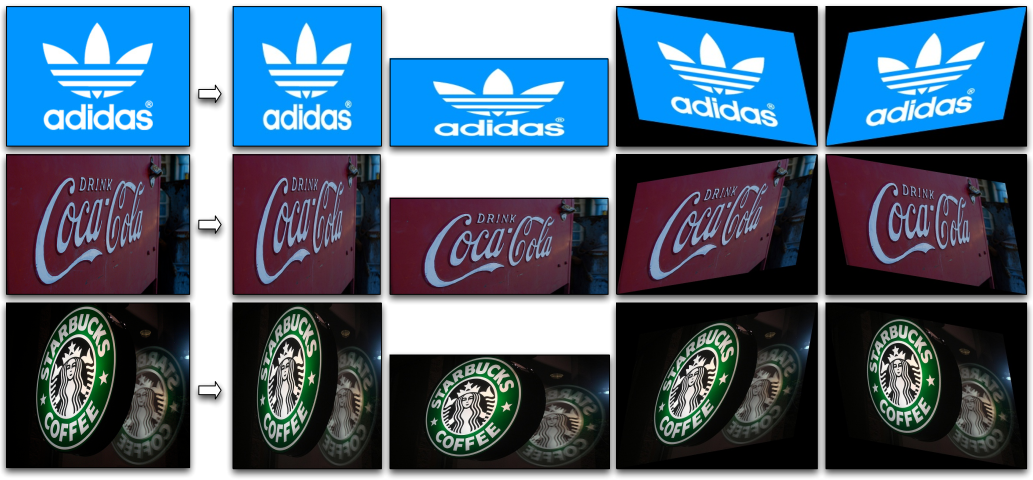


Figure 7.8: Warping: Applying affine transformations to images yields new images with synthetically introduced variation. Images taken from the FlickrLogos-32 dataset.

with different vocabularies may have a significant computational burden and memory overhead, we do not use this scheme in the following but rather stick to a single vocabulary.

7.3 Warping

While current local features are by design scale invariant and also somewhat robust to changes in lighting and image noise, it is well-known that local features such as SIFT are particularly susceptible to changes in perspective. With increasing vocabulary size this effect gets more severe: descriptors computed from image patches that are actually identical but seen from a different perspective are quantized to different – and therefore unrelated – visual words.

There exist several partial solutions to this problem. The most popular is query expansion (QE) where the top-ranked retrieved images are exploited to augment the original query. The augmented query is then re-issued in order to retrieve images that have not been found in the first round. Consequently, query expansion fails – and causes the results to be worse than without – if the top-retrieved images are false positives. This may happen if the query is actually challenging or only few true positives are contained in the database.

We propose a different method to overcome this problem, especially suited for small objects where it is crucial to find the few true matching visual words. It is a purely data-driven approach that synthesizes new images from existing images by applying transformations to the image itself, a process also termed "warping" (see Figure 7.8 for examples).

We explore three different ways to exploit image warping:

1. *Synthetic Query Expansion (SynQE)*: Multiple versions of the query image may be synthesized simulating the query as it may be seen under different conditions and perspectives.

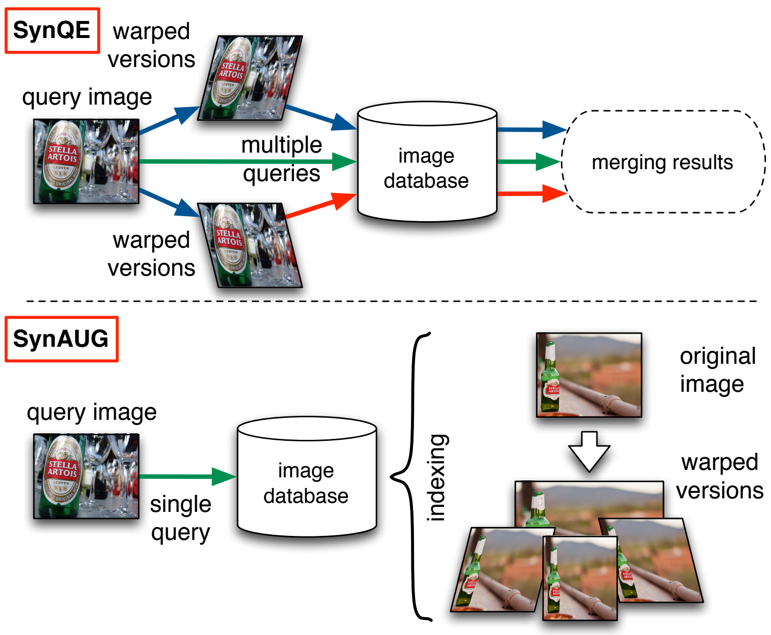


Figure 7.9: Top: Synthetic query expansion. **Bottom**: Synthetic database augmentation.

Each image is then treated as an individual query; their corresponding result lists are then merged into a single list. This method is illustrated in the upper half of Figure 7.9.

2. *Synthetic Database Augmentation (SynAUG)*: The database is augmented by adding new generated images synthesized from each original database image. This is especially useful if it is desired that a query containing certain predefined objects – such as logos – should find the true results with high probability from a limited set of manually managed reference images. This is method is illustrated in the lower half of Figure 7.9.
3. Synthetic Query Expansion + Synthetic Database Augmentation *SynQE + SynAUG*: The combination of (1) and (2). This can be seen as counterpart to ASIFT (Morel and Yu 2009) working with discrete visual words and an inverted index or another database instead of comparing raw descriptors between two images.

We choose the following simple transformations to synthesize new images: $S_x(\alpha)$, $S_y(\alpha)$, $S_x(\alpha)R(45°)S_x(\alpha)$ and $S_x(\alpha)R(-45°)S_x(\alpha)$. $S_x(\alpha)$ denotes the matrix for scaling by factor α in x-direction, $S_y(\alpha)$ analog in y-direction and $R(45°)$ denotes the matrix for rotation by 45°. The last two transformations are opposed shearings along x direction. Note that the two shearings along y-direction are equivalent. The inverse transformations of the former four are added as well, resulting in a total of eight transformations. Examples of the first four transformations are shown in Figure 7.8.

Intuitively, the synthesizing of images creates variations from a single image. The more variation is captured within the index the more likely the retrieval of an arbitrary query will

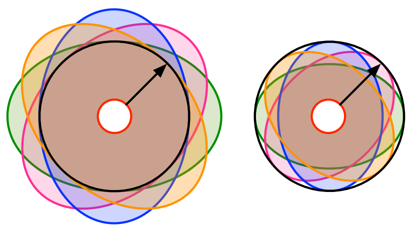


Figure 7.10: Visualization of the image patches described by visual words computed from warped images *back-projected* into the original image. The original image patch is shown as black circle. **Left**: $\alpha = 0.7$. **Right**: $\alpha^{-1} \approx 1.43$. Images to scale.

succeed. To illustrate the effect of such warping on an individual local feature the used transformations are visualized in Figure 7.10. The original patch described by a local feature is shown as black circle. Once images are warped with a scaling factor of $\alpha < 1$ the image is effectively down-scaled such that the actual described image region is larger than the original patch. Similar, if the images are up-scaled ($\alpha > 1$) the actual described image patch is smaller than the region described by the original local feature. The elliptic shape depends on the transformation, and while obtained differently it reminds one of affine covariant regions as obtained by Hessian-affine or Harris-affine feature detectors (Mikolajczyk and Schmid 2004). In fact, our technique effectively simulates a global affine transformation applied to the whole image while affine covariant detectors estimate an affine transformation per local feature.

In practice, the following issue needs to be addressed: We observed artifacts when computing local features from warped images. The feature detector often fires close to or directly on the boundaries of transformed images. Even while detections on edge-like structures are suppressed during feature detection, due to image noise these still occur on those boundaries. Moreover, placing the transformed images within an empty background (black) implicitly yields local contrast extrema between the image corners and that background. Examples of the resulting artifacts are visualized in Figure 7.11.

To the best of our knowledge there is no clear way to avoid this issue directly during interest point detection. Thus, we discard features closer to the boundary of the transformed image than half of their radius in a separate step. Obviously, images that have been stretched in x- or y-direction only (with $R(0°)$) do not need post-processing.

For our synthetic query expansion and database augmentation scheme it is important and for the combination of both it is mandatory to discard such detections. The corresponding visual words do not carry useful information as they mostly describe black background. As a consequence, these lead to spurious false visual word correspondences between unrelated images, which in turn deteriorate the retrieval. Once these detections are discarded the retrieval with features from warped images behaves, as one would expect.

For SynQE multiple queries are issued to the index yielding multiple separate result lists. These are merged subsequently: images contained in multiple result lists get the maximum

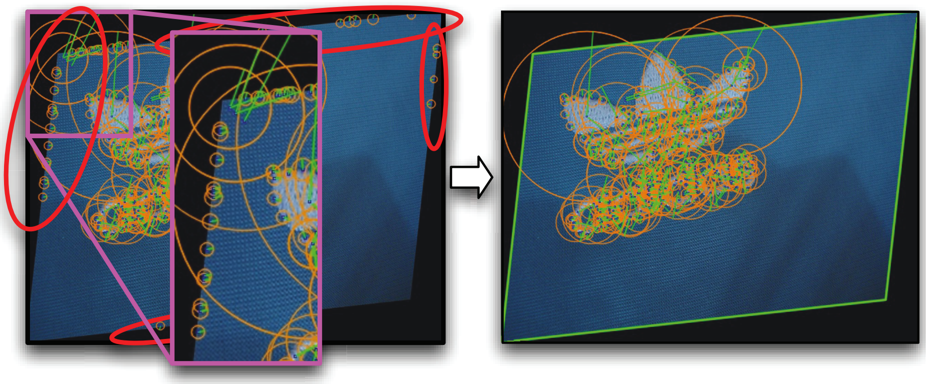


Figure 7.11: Left: Local feature patches (orange circles) with their associated orientation (green line) as extracted from warped images. The detector often fires on the boundaries and introduces artifacts (marked red). **Right**: Features retained after eliminating those too close to the boundary (green contour).

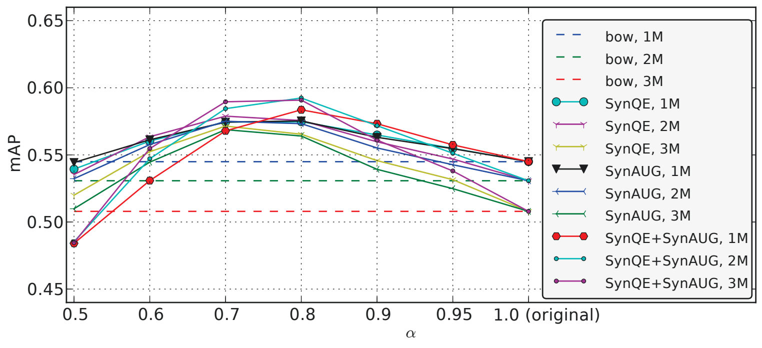


Figure 7.12: FlickrLogos-32: Impact of synthetic query expansion and database augmentation on BoW retrieval performance.

of each individual cosine similarity score as proposed by Arandjelović and Zisserman (2012a). Similar for SynAUG: once a synthetic image is found it votes with its score for the original image and the maximum of all votes is taken as final similarity measure.

We test these techniques with a bag-of-words retrieval as described in Section 7.2.5 (RootSIFT, tf-idf-sqrt) and vocabularies of 1M, 2M and 3M words. The scaling parameter α is varied from 0.95 to 0.5 to determine which group of transformations works best for simulating the perspective change in practice.

The corresponding results on the FlickrLogos dataset are shown in Figure 7.12. Both SynQE and SynAUG improve the retrieval performance with a maximum at $\alpha = 0.7/0.8$. The combination of both, i.e., SynQE+SynAUG slightly increases the performance further. An even larger visual vocabulary of 2M visual words increases the performance significantly over its baseline (11.6%) but somewhat surprisingly only slightly above those of the vocabulary with 1M words.

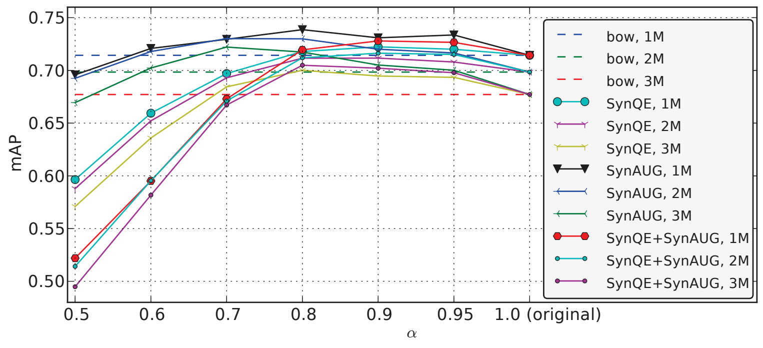


Figure 7.13: Oxford5K: Impact of synthetic query expansion and database augmentation on bag-of-word retrieval performance.

The results on the Oxford5K dataset are shown in Figure 7.13. Here, SynQE and SynAUG also improve retrieval performance though less pronounced. Consistently SynAUG performs slightly better than SynQE. Again, the performance of the vocabularies with 2M and 3M words increases significantly over their baselines (bag-of-words without SynQE/SynAUG). While these do not perform better than the 1M vocabularies the most interesting behavior is that the performance of all vocabularies and methods seems to be "saturated" at around a mAP of 0.73. In other words, the actual choice of the vocabulary size has reduced impact. Thus, larger vocabularies that scale better with larger image databases can be used with little loss of mAP.

To summarize, the results on both datasets underline that discrete visual descriptions benefit from synthetic image generation - especially for small object retrieval such as logos. In the following we refer to the transformation group with $\alpha = 0.7$ when referring to SynQE and SynAUG.

7.4 Logo Recognition

As our feature bundling approach targets object retrieval, demonstrating its performance by a logo recognition system seems natural. For that, we assemble the building blocks we have discussed before (visual features, vocabularies, feature bundling, geometric re-ranking, synthetic query expansion and database augmentation) and present our final logo recognition system.

Indexing The logo classes that our system should be able to detect are described by a set of images showing these logos in various poses. We coin this set *reference set* and use the images within the training and validation set of the FlickrLogos-32 dataset for this purpose. Feature bundles are computed for each image in the reference set and inserted into the hash table associated with the information to which class a reference image belongs. Optionally, we can employ synthetic database augmentation: Artificially generated transformed versions of the original images are used to augment the reference set.

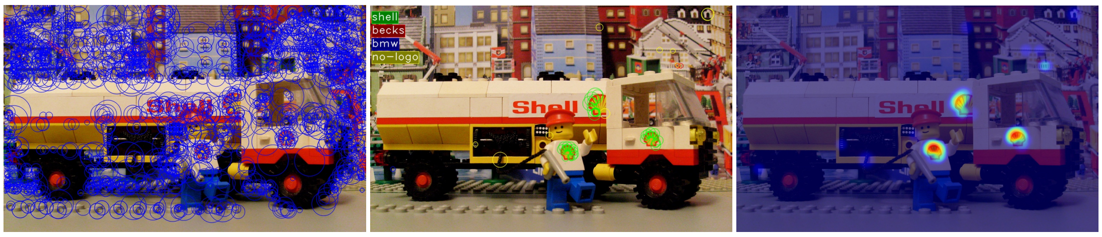


Figure 7.14: Logo detection by searching for similar bundles via Bundle min-Hashing. **Left**: Local features (blue circles) of an image showing the Shell logo. **Middle**: The bundles where Bundle min-Hashing (*without* SynQE, SynAUG or spatial re-ranking) found similar bundles associated with a certain class (color-coded) in the index. A few false positives are found in the background. **Right**: The heat map shows the bundle hits visualized with a multi-scale multi-bandwidth Kernel Density Estimation incorporating both the scale of the bundles as well as the respective number of collisions. Due to the latter the false positive detections have negligible impact.

Testing An image is being tested for the presence of any of the logo classes by computing feature bundles and performing lookups in the hash table to determine the reference images that share the same bundles. The retrieved list of images is then re-ranked as described in Section 7.2.3. Optionally, synthetic query expansion may be applied: Multiple transformed versions of the original query image are used to query the database multiple times or the database as described in Section 7.3. Afterwards the fast spatial re-ranking with 1P-WGC-RANSAC without projective refinement (see Section 6.5) is applied to the retrieved list. Finally an image is classified by a k-nn classifier: A logo of the class c is considered to be present if the majority of the top k retrieved images is of class c. In our experiments we empirically chose $k = 5$.

Experimental Setup We follow the evaluation protocol as a given in Romberg et al. (2011). That is, training and validation set including non-logo images are indexed by the respective method. The whole test set including logo and logo-free images (3960 images in total) is then used to compute the classification scores.

Results Table 7.2 shows the obtained results for various approaches. The first system we compare to, is our own approach (Romberg et al. 2011) that performs logo detection by detecting spatial configurations of local feature triples as described in Chapter 5. Revaud et al. (2012) use a bag-of-words-based approach coupled with learned weights that down-weight visual words that appear across different classes.

It can be seen that a bag-of-words-based search as described in Section 7.2.5 followed by 5-nn majority classification already outperforms these two approaches significantly. In fact, our approach using bag-of-words to retrieve the logos and performing a majority vote among the top 5 retrieved images already outperforms the best results reported in the literature so far.

One can see, that Bundle min-Hashing also significantly outperforms the former two systems out of the box. The difference between a ranking based on sketch collision counts ("collision

Method	Precision	Recall
Romberg et al. (Romberg et al. 2011)	0.98	0.61
Revaud et al. (Revaud et al. 2012)	≥ 0.98	0.73
bag-of-words, 100K	0.988	0.674
bag-of-words, 1M	0.991	0.784
bag-of-words, 1M, SP	0.996	0.813
bag-of-words, 1M, SP+SynQE	0.994	0.826
bag-of-words, 1M, SP+SynAUG	0.996	0.825
BmH, 200K, collision count	0.688	0.411
BmH, 200K, CosSim	0.987	0.791
BmH, 1M, collision count	0.888	0.627
BmH, 1M, CosSim	0.991	0.803
BmH, 1M, CosSim+SP	0.996	0.818
BmH, 1M, SP only	0.996	0.809
BmH, 1M, CosSim+SP+SynQE	**0.999**	**0.832**
BmH, 1M, CosSim+SP+SynAUG	0.996	0.829

Table 7.2: FlickrLogos-32: Logo recognition results.

count) and a ranking based on cosine similarity ("CosSim") makes clear that the result lists obtained by BmH needs to be re-ranked to ensure that the top-most images are indeed the most similar ones. We compared Bundle min-Hashing with a vocabulary 200K words (having the highest mAP, see Table 7.1) to Bundle min-Hashing with a larger vocabulary of 1M words (slightly lower mAP). In case of our logo recognition system, we are only interest in the top 5 results. The vocabulary of 1M words slightly improves the results but moreover it also reduces the complexity of the system as it eliminates the need for two different vocabularies for bundling and re-ranking. In addition, the response ratio of this system is 100 times smaller (response ratio = 0.0096) than that of bag-of-words.

It can be further seen that both synthetic query expansion (SynQE) and synthetic database augmentation (SynAUG) consistently improve the classification performance for both bag-of-words and Bundle min-Hashing. Compared to the retrieval setting (see Section 7.3), the effect is less pronounced as we only measure its impact on the 5 top-ranked results.

Finally, we demonstrate how Bundle min-Hashing accurately localizes the logos in Figure 7.14 and 7.16. For completeness, the true positives per class for our best system are further shown in Figure 7.15.

Failure cases While Bundle min-Hashing works remarkably well for a wide range of object types and object sizes, it is by definition dependent on the performance of local features. Thus, the chance of capturing the visual appearance of an object by aggregating multiple features decreases with decreasing number of detected features on the object. This is especially true for low-contrast or low-structured objects (e.g., the apple or pepsi logo) were only a few local features are detected at best.

A further issue in practice is the non-distinctiveness of local features on image content such as text characters. Text usually generates many non-informative visual words yielding spurious

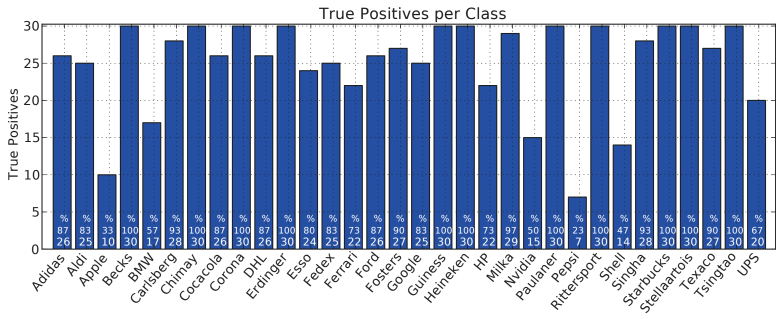


Figure 7.15: True positives per class for the best performing system from Table 7.2 (BmH, 1M, CosSim+SP+SynQE).

matches and random collisions in the hash table. An example for this issue can be seen in the right-most image of Figure 7.16. While Bundle min-Hashing is *much* less affected than e.g., a regular bag-of-words voting scheme it is still not completely unimpaired giving raise to future improvements.

Index pruning When analyzing the key-value distributions of the hash tables in our index we found that there were only ≈ 1.04 values stored per sketch on average. In other words, most of the sketches were generated only once. As the hash table lookup assumes that similar images share similar bundles and therefore sketches, we can further assume that if a sketch was generated only once from all images in the training+validation set, it will likely not have an impact when unknown images are tested for the presence of logos. Consequently, we remove those entries from our index resulting in *significantly* less items stored within the index. Our results show a striking advantageous effect of the highly distinctive image search with Bundle min-Hashing and a subsequent re-ranking with the cosine similarity (Section 7.2.3).

The mAP after removing those keys from the hash table that have less than m bundles is shown in Table 7.3. The results are rather surprising: the performance slightly improves even though the hash table contains approximately 30 ($m = 2$) or 140 times ($m = 3$) fewer entries. Note that the number of remaining keys is even smaller than the visual vocabulary itself. Therefore, the Bundle min-Hashing seems to serve as feature (pre-)selection technique; the features are then selected by their occurrence frequency. Bundle min-Hashing thus acts as a highly effective pre-filter for retrieval. In the end, this scheme is a lossy yet highly effective index compression.

To give an idea of the implications we measured the rough actual memory consumption of our application before and after index pruning shown in columns "memory" in Table 7.3. With 4 sketches the memory usage decrease from 1.5 Gigabytes to a fourth ($m = 2$) down to about 15% with $m = 3$. Note that we measured the total memory consumption containing all allocated memory[1] of our application and the hash tables may not have been perfectly shrunk

[1]The memory consumption was measured as "Private Bytes" in Sysinternals Process Explorer on the same

sketches	original			index pruning					
				m=2			m=3		
	mAP	keys	memory	mAP	keys	memory	mAP	keys	memory
4	0.554	20,311,553	1,544 MB	0.558	749,303	390 MB	0.559	147,382	239 MB
3	0.545	15,234,817	1,198 MB	0.549	561,755	380 MB	0.549	110,143	213 MB
2	0.527	10,156,293	852 MB	0.529	374,437	254 MB	0.528	73,609	190 MB
1	0.478	5,078,358	484 MB	0.475	187,228	155 MB	0.471	36,577	135 MB

Table 7.3: Impact of index pruning: mAP on FlickrLogos for BmH configurations as shown in Table 7.1 when sketches occurring less than m times are removed from hash tables. The total number of sketches stored in hash tables is denotes as *keys*.

to fit the actual number of keys after the pruning. The large savings in memory consumptions are interesting for further investigations for scalability on large databases and may also enable to run Bundle min-Hashing efficiently on devices with little memory, such as mobile phones.

7.5 Summary

In this chapter we described a robust feature bundling technique for image-, object- and logo retrieval. We extensively evaluated our approach on three different datasets. In addition, we presented a method that uses synthetically generated variations of images to increase recall, by either querying the index multiple times or by indexing multiple variants of each image.

Finally, we demonstrate a relatively simple but highly effective logo recognition system assembled from the presented components. It uses Bundle min-Hashing as underlying retrieval technique to find highly distinctive local feature bundles in a database of reference images. Synthetic query expansion and synthetic database augmentation are used to further increase recall. The 1P-WGC-RANSAC subsequently re-ranks the retrieval results with respect to their spatial consistency to the query. Finally, the top-ranked images from the set of reference images are used to decide whether the query image actually shows a logo.

The bundle representation is highly effective; future work may adopt it for new applications. Potential improvements could be the incorporation of the internal geometry, such as the relative spatial layout of the local features. This may be either encoded into the bundle description itself or used within post-retrieval verification step.

That in mind, we would like to point out that Bundle-min-Hashing shares close resemblance with NAPSAC due to the sampling of features from local neighborhoods. One may imagine that these two methods are fused into a holistic geometry-aware retrieval system. The promising results of the index pruning procedure give raise for future optimizations for scalability.

machine as described in Appendix D. One MB denotes 1024^2 bytes (a Mebibyte).

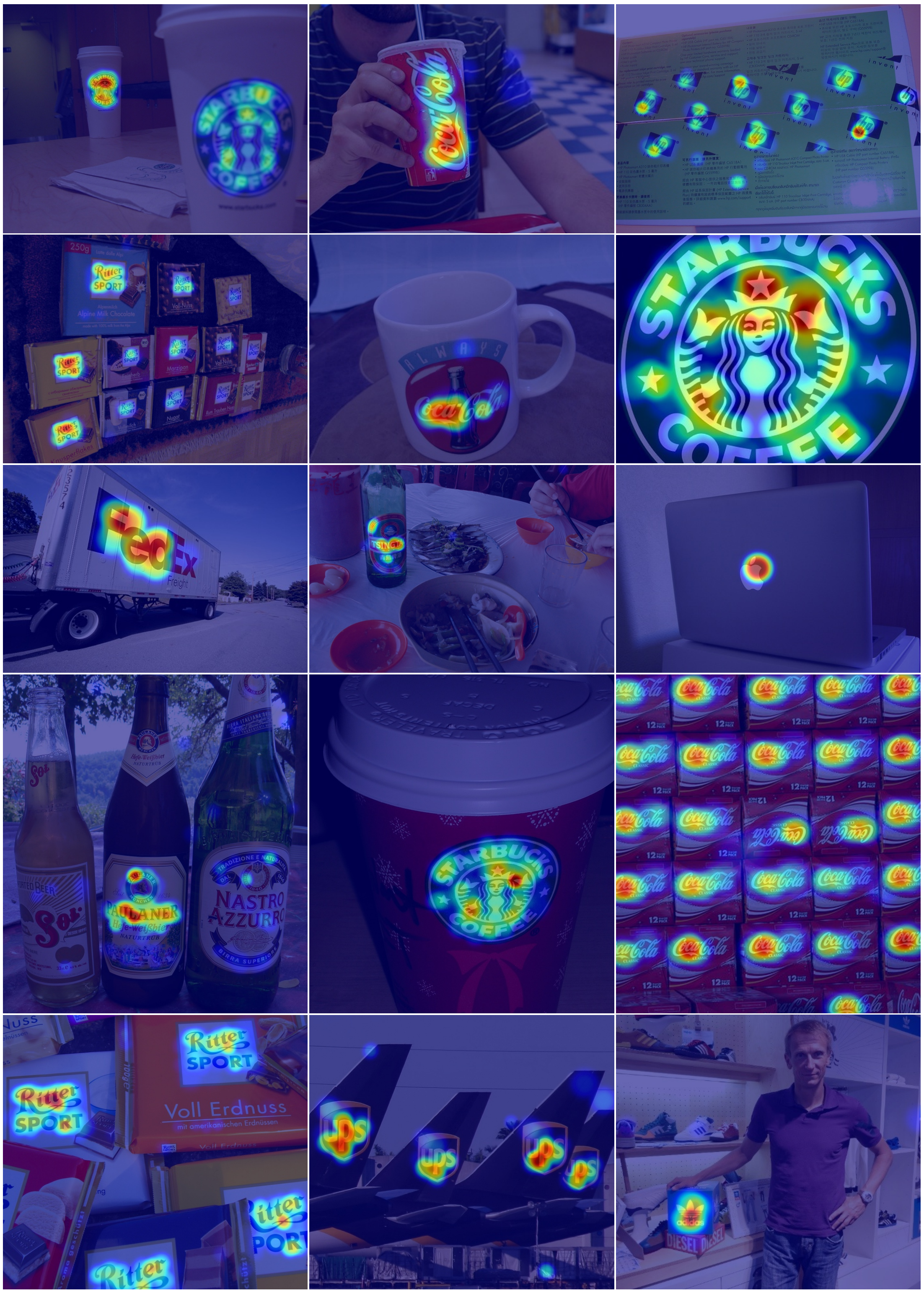


Figure 7.16: Examples of detections by Bundle min-Hashing for various logos. The heat map shows the bundle hits visualized with a multi-scale multi-bandwidth Kernel Density Estimation incorporating both the scale of the bundles as well as the respective number of collisions.

Part IV

Conclusion

8 Conclusion

8.1 Summary

In this thesis, we presented several retrieval techniques for content-based image retrieval. We used publicly available real-world databases for the evaluation of various scenarios.

First, we have presented a framework for deriving a multimodal representation for images. Here, the information from different modalities or domains are fused into a single description. This description is obtained by learning and inferring a hierarchy of topic distributions in a fully unsupervised manner by the multimodal multilayer pLSA. As a result, the unified representation not only represents multiple modalities but also acts as a data reduction method projecting the high-dimensional input vectors into a latent topic space with much fewer dimensions. We thoroughly evaluated the multimodal multilayer pLSA on two large-scale databases by combining visual features and tags as well as two visual features. We showed that the combination of multiple modalities in our mm-pLSA model consistently improves the results significantly over the baseline that uses just a single modality.

As the global optimization of the model due to multiple modalities and layers is computationally demanding, we proposed a fast initialization strategy. This strategy can be easily and quickly computed in a step-wise forward procedure computing one model at a time. Moreover, the fast initialization strategy alone already outperforms the baseline significantly and may serve as a technique of its own. Once the mm-pLSA is initialized by the outcome of the fast initialization strategy, the model can be further optimized across all modalities and layers.

The second part of this thesis focused on object retrieval – in particular on the retrieval and detection of small objects such as logos.

We presented a logo detection system based on retrieval techniques. By choosing retrieval techniques over learning-based classifiers we are able to scale the systems up to potentially thousands of classes. Our approach is based on a highly distinctive signature capturing both the visual appearance and the spatial layout of local features. Logos are modeled by distinctive configurations of local features. We described a method to automatically derive feature pairs and feature triples from training images. In addition, we proposed a specialized data structure – the cascaded index – for efficient testing whether the spatial configurations of an unknown image do belong to a logo of one of the known brands. This approach was evaluated on a newly created dataset of logos and tuned for high precision showing that the representation of local feature triples is highly distinctive. Therefore, this scheme effectively prevents false positives.

The second major contribution is a novel RANSAC variant. Based on 1-point correspondences it estimates similarity transformations between images and further employs a weak geometric constraint for a more robust yet faster re-ranking. We also show that a costly refinement of the estimated transformation is not necessary for re-ranking and can be omitted. We demonstrate that this approach outperforms non-WGC-constrained variants – especially when used with small vocabularies as the WGC-constraint effectively discards false correspondences. To summarize, our 1P-WGC-RANSAC variant has state-of-the art performance while being extremely fast.

In addition, we present a novel feature bundling technique based on min-Hashing. In our proposed Bundle min-Hashing scheme local features within the neighborhood around central features are aggregated into bundles. Subsequently, the bundles are encoded with min-Hashing and their representation is stored in hash tables. Our approach by min-Hashing makes our bundling technique suitable for retrieval as we can efficiently query hash tables for similar bundles without requiring exact matches. We perform extensive experiments on three datasets and demonstrate that Bundle min-Hashing has equal performance in terms of mean Average Precision but has orders of magnitude lower response ratio and also deteriorates slower than bag-of-words in the presence of noise. In addition, its precision is high and even tunable at query time depending on the application needs. It outperforms several other min-Hash based approaches in the literature by having both higher mean average precision and better recall.

Finally, we use our Bundle min-Hashing scheme as retrieval technique within a logo recognition framework. Here, an image is tested for the presence of logos by querying a database of reference images. Once the query image shares similar bundles with reference images, a majority vote among the most similar reference images is used to decide which logo is present. The proposed scheme is simple, fast and highly effective. We show that it outperforms other existing approaches while having desirable properties such as high speed due to constant-time lookups, high precision and a very low response ratio.

8.2 Outlook

There are several possible directions for future work originating from the work in this thesis. Some of those are sketched in the following.

Our multimodal multilayer pLSA presented in Chapter 4 may be adopted or serve as inspiration for future multilayer models. The incorporation of geometry into the model is likely to improve the visual description. We expect similar reward when removing the hardly tenable independence assumption between individual visual words.

The cascaded index we proposed in Chapter 5 may be of further use in various applications. In general, its underlying concept and the corresponding search strategy is applicable to higher order features of any kind – whenever an "is part of" relationship links the combinatorial spaces of lower and higher order descriptions.

One can further imagine employing our 1P-WGC-RANSAC variant we described in Chapter 6 in a cascade-like manner. A RANSAC based on 1-point correspondences with WGC-constraint could be used to discard false correspondences in a first step. The subsequent estimation of a affine and projective transformations may then be performed on almost outlier-free samples. Combining it with further pre-tests as in the randomized RANSAC framework may eventually lead to a cascade over multiple stages.

We believe the Bundle min-Hashing scheme presented in Chapter 7 is applicable to a wide range of settings and may deserve further research. The bundle representation naturally increases the distinctiveness of a visual representation compared to individual features. Due to its ability for robust approximate similarity search, it may be used to increase the distinctiveness of features that are not as strict as traditional local features. Especially the bundling scheme may be extended to encode the internal geometry of feature bundles to increase the precision even further.

Besides that, we feel that gradient representations such as SIFT or HOG alone are not capable to describe all kinds of objects. Especially objects with little texture – such as the Pepsi or Apple logo – may benefit from visual descriptions capturing their shape and color. While such descriptions have been studied extensively since decades, these have not been as successful as gradient-based descriptions. One problem is that these descriptions are often not as distinctive as the latter. That in mind, the Bundle min-Hashing framework may be exploited to aggregate features of different types into bundles. Due to the combined representation and the min-Hashing scheme, Bundle min-Hashing offers good opportunities to capture visual appearances of different modalities into a highly distinctive signature. Such bundles may improve object retrieval especially for objects without distinctive texture or colorful appearance.

A
Publications

Publications in Conferences and Journals

Parts of the work presented in this thesis have been published in the following papers:

1. Romberg, S. and Lienhart, R. (2013). Bundle Min-Hashing, *International Journal of Multimedia Information Retrieval*, **2**(4): 243–259, Springer.
2. Romberg, S. and Lienhart, R. (2013). Bundle Min-Hashing for Logo Recognition, In *Proceedings of ACM International Conference on Multimedia Retrieval 2013*, pp. 113–120, ACM.
3. Romberg, S., August, M., Ries, C.X. and Lienhart, R. (2012). Robust Feature Bundling, In *Advances in Multimedia Information Processing - PCM 2012, Lecture Notes in Computer Science*, **7674**:45–56, Springer.
4. Romberg, S., Lienhart, R. and Hörster, E. (2012). Multimodal Image Retrieval, *International Journal of Multimedia Information Retrieval*, **1**(1): 31–44, Springer.
5. Richter, F., Romberg, S., Hörster, E. and Lienhart, R. (2012). Leveraging Community Metadata for Multimodal Image Ranking, *Multimedia Tools and Applications*, **56**(1): 35–62, Springer.
6. Romberg, S. (2011). From Local Features To Local Regions, In *Proceedings of ACM Multimedia 2011, Doctorial Symposium*, pp. 841–844, ACM.

7. Romberg, S., Pueyo, L.G., Lienhart, R., van Zwol, R. (2011). Scalable Logo Recognition in Real-World Images, In *Proceedings of ACM International Conference on Multimedia Retrieval 2011*, Article No. 25, ACM.
8. Ries, C.X., Romberg, S. and Lienhart, R. (2010). Towards universal visual vocabularies, In *Proceedings of IEEE International Conference on Multimedia and Expo 2010*, pp. 1067–1072, IEEE.
9. Richter, F., Romberg, S., Hörster, E. and Lienhart, R. (2010). Multimodal Ranking for Image Search on Community Databases, In *Proceedings of ACM International Conference on Multimedia Information Retrieval 2010*, pp. 63–72, ACM.
10. Lienhart, R., Romberg, S. and Hörster, E. (2009). Multilayer pLSA for Multimodal Image Retrieval, In *Proceedings of ACM International Conference on Image and Video Retrieval 2009*, Article No. 9, ACM.
11. Romberg, S., Hörster, E. and Lienhart, R. (2009). Multimodal pLSA on Visual Features and Tags, In *Proceedings of IEEE International Conference on Multimedia and Expo 2009*, pp. 414–417, IEEE.

Patents

Stefan Romberg is co-inventor of the triangle representation and the cascaded index presented in Chapter 5. These are covered by US Patent US2012/0263385 A1 (published Oct. 18, 2012).

B
Branchless *angle_diff()*

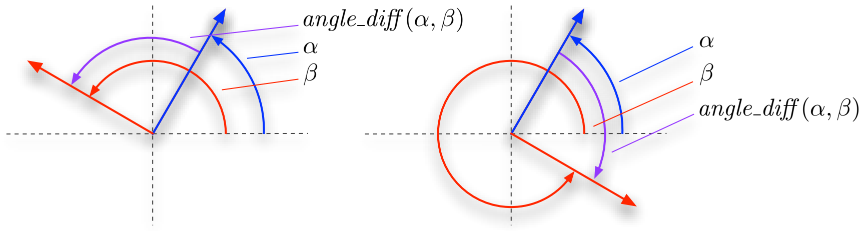


Figure B.1: The difference of two angles is the smallest enclosed angle.

We define the difference of angles as the smallest angle enclosed by the two vectors α and β (see Figure B.1). The difference is anti-symmetric ($angle\_diff(\alpha, \beta) = -angle\_diff(\beta, \alpha)$) where negative angles mean clock-wise rotation. It follows that its output range is in $[-180°, 180°[$.

While a naive implementation of $angle\_diff()$ to compute the difference of angles as used in Section 5.2.2 and Section 6.5 is easy, its performance is not satisfying especially if this basic operation is invoked frequently. By analyzing different variants we were able to derive an optimized branchless version. Originally $angle\_diff(\alpha, \beta) = mod((\alpha - \beta) + 180, 360) - 180$, which shifts the difference, takes the modulo, and shifts it back to the desired range. This seemingly disadvantage of shifting and back-shifting is indeed handy and canceled out later.

Here, the *mod* function denotes the modulo operation in the mathematical sense. That is, $-2 \ \% \ 5 = 3$. In many programming languages the `mod()` function or simply the % operator will not handle negative values as in the mathematical sense (e.g., $-2 \ \% \ 5$ yields -2). However, a modulo operator $xmod$ that handles negative values correctly can be implemented as $xmod(x, y) = x - y \cdot floor(\frac{x}{y})$. Putting this into the original equation (and fixing $y = 360$) results in $angle\_diff(\alpha, \beta) = x - 360 \cdot floor(x \cdot \frac{1}{360}) - 180$ where $x = \alpha - \beta + 180$. This can be further rearranged to $angle\_diff(\alpha, \beta) = x - 360 \cdot floor((x + 180) \cdot \frac{1}{360})$ now with $x = \alpha - \beta$, i.e., effectively 1 subtraction less.

The key observation is that $floor((x + 180) \cdot \frac{1}{360} = floor(\frac{x}{360} + 0.5)$. Thus, this particular expression is effectively a "round to nearest" operation, which can be replaced by the corresponding hardware instruction. Additionally this saves one add and one set instruction. The final expression then is $angle\_diff(\alpha, \beta) = ((\alpha - \beta) - (360 \cdot round((\alpha - \beta) \cdot 0.002\overline{7})$.

Using SSE intrinsics this can be coded in `C` as sequence of instructions that are close to pure assembler but more portable (see Listing B.1). At least SSE 4.1 is required for the *_mm_round_ss* intrinsic.

```c
#include <smmintrin.h>  // For SSE 4.1

/// Computes the difference of two angles *angle0* and *angle1*
/// (degrees) such that the resulting angle is in `[-180, 180[`.
inline float angle_diff(float angle0, float angle1)
{
    __m128 a = _mm_set_ss(angle0);
    __m128 b = _mm_set_ss(angle1);
    __m128 x = _mm_sub_ss(a, b);
    b = _mm_set_ss(0.0027777777778f);
    a = _mm_mul_ss(x, b);
    a = _mm_round_ss(a, a, _MM_FROUND_TO_NEAREST_INT);
    b = _mm_set_ss(360.0f);
    a = _mm_mul_ss(a, b);
    a = _mm_sub_ss(x, a);
    return _mm_cvtss_f32(a);
}
```

Listing B.1: Fast branchless *angle_diff()* function using SSE intrinsics.

This function is branchless and between 3 and 10 times faster than other variants we have tested. Moreover, this scheme can be easily extended to compute four angle differences in parallel.

C Hash Function for min-Hashing

The following hash function (see Listing C.1) was used to compute the min-Hashes for the min-Hash-based retrieval methods as described in Chapter 7.

It is based on a random number generator that uses the Multiply-With-Carry scheme. The state of the generator is initialized by a seed that is different for each hash function and kept fixed. The output of this hash function thus only depends on the fixed seed and the visual word label v.

The seed and the visual word label are first combined into a single integer number by a pairing function. We used the doubled cantor pairing function because it produced significantly better results than a simpler scheme. The multiplication of the two numbers spread the resulting numbers over a large range even when the seeds are chosen from a contiguous range.

The resulting 64-bit number is multiplied by a large constant number. This multiplication causes overflow (depending on the current value) and introduces pseudo-randomness in the lower 32 bits. These and the upper 32 bits of the former value are then mixed and returned as final 32-bit hash value.

The hash function can then be used for min-Hashing as follows: The min-Hash mh of the hash function seeded with n is determined by finding the visual word v_i of a set $\mathcal{I}$ that produces the smallest hash value:

$$mh_n = \operatorname*{argmin}_{v_i \in \mathcal{I}} hash(n, v_i) \tag{C.1}$$

```
/// Returns the hash value for input *v* given a fixed *seed*.
uint32_t hash(unsigned int seed, unsigned int v) const
{
    // combine seed and v into single number with
    // doubled cantor pairing function
    uint32_t z    = v + seed;
    uint64_t temp = z * (z + 1) + ( seed << 1 );

    // spread outcome over entire range
    temp = (uint64_t)(uint32_t) temp*4164903690U + (temp >> 32);
    return (uint32_t) temp;
}
```

Listing C.1: Hash function for min-Hashing

D
Machine Specifications for Timing Benchmarks

The following Table D.1 shows the specifications of the machine that was used for measure the timings as presented in Sections 6.6

Operating System	Windows 7, x64
CPU Name	Intel Xeon X5550 (Gainestown)
Specification	Intel(R) Xeon(R) CPU X5550 @ 2.67GHz
Number of cores	4 (+Hyperthreading: max 8)
Number of threads	8 (+Hyperthreading: max 16)
Clock Speed	2.66 GHz
Max Turbo Frequency	3.06 GHz
Instructions sets	MMX, SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2, EM64T, VT-x
L1 Data cache	4 x 32 KBytes, 8-way set associative, 64-byte line size
L1 Instruction cache	4 x 32 KBytes, 4-way set associative, 64-byte line size
L2 cache	4 x 256 KBytes, 8-way set associative, 64-byte line size
L3 cache	8 MBytes, 16-way set associative, 64-byte line size
Mainboard Model	Mac-F221BEC8
Northbridge	Intel 5520 rev. 13
Southbridge	Intel 82801JR (ICH10R) rev. 00
Memory	DDR3, 12278 MBytes, Triple Channel, 532.0 MHz

Table D.1: Machine Specifications for Timing Benchmarks

List of Figures

List of Tables

Bibliography

Alahi, A., Ortiz, R. and Vandergheynst, P. (2012). FREAK: Fast Retina Keypoint, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2012*, pp. 510–517. 11

Alhwarin, F. and Ristić-Durrant, D. (2010). VF-SIFT: Very fast SIFT feature matching, *Pattern Recognition, Lecture Notes in Computer Science*, pp. 222–231. 10

Arandjelović, R. and Zisserman, A. (2012a). Multiple queries for large scale specific object retrieval, In *Proc. of British Machine Vision Conf. 2012*, BMVA, pp. 92.1–92.11. 137

Arandjelović, R. and Zisserman, A. (2012b). Three things everyone should know to improve object retrieval, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2012*, pp. 1–11. 14, 15, 112, 113, 128

Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding, In *Proc. of ACM-SIAM Symposium on Discrete Algorithms 2007*, **8**: 1027–1035. 18

Arya, S. and Mount, D. (1993). Algorithms for fast vector quantization, In *Proc. of Data Compression Conf. 1993*, IEEE, pp. 1–17. 25

Avrithis, Y., Tolias, G. and Kalantidis, Y. (2010). Feature map hashing: sub-linear indexing of appearance and global geometry, In *Proc. of ACM Multimedia 2010*, pp. 231–240. 82

Bagdanov, A. D., Ballan, L., Bertini, M. and Del Bimbo, A. (2007). Trademark matching and retrieval in sports video databases, In *Proc. of Int. Workshop on Multimedia Information Retrieval 2007*, pp. 79–86. 81

Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D. and Jordan, M. (2003). Matching words and pictures, *Journal of Machine Learning Research*, **3**: 1107–1135. 53, 64

Bay, H., Ess, A., Tuytelaars, T. and Van Gool, L. (2008). Speeded-up robust features (SURF), *Computer Vision and Image Understanding*, **110**(3): 346–359. 10, 11

Beis, J. S. and Lowe, D. G. (1997). Shape indexing using approximate nearest-neighbor search in high-dimensional spaces, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 1997*, pp. 1000–1006. 23, 25

Belongie, S., Malik, J. and Puzicha, J. (2002). Shape Matching and Object Recognition Using Shape Contexts, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(4): 509–522. 11

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching, *Communications of the ACM*, **18**(9): 509–517. 24

Blei, D. M. and Jordan, M. I. (2003). Modeling annotated data, In *Proc. of ACM SIGIR Conf. on Research and Development in Information Retrieval*, pp. 127–134. 53

Blei, D. M. and Lafferty, J. D. (2005). Correlated topic models, In *Proc. of Advances in Neural Information Processing Systems 2005*, Curran Associates, Inc., **18**: 147–154. 53

Blei, D. M., Ng, A. Y. and Jordan, M. I. (2003). Latent Dirichlet Allocation, *Journal of Machine Learning Research*, **3**(4-5): 993–1022. 53

Bosch, A., Zisserman, A. and Munoz, X. (2006). Scene classification via pLSA, *Computer Vision – ECCV 2006 Lecture Notes in Computer Science*, Springer, **3954**: 517–530. 53

Broder, A. (1997). On the resemblance and containment of documents, In *Proc. of Compression and Complexity of Sequences 1997*, pp. 21–29. 118, 120

Brown, M. and Lowe, D. G. (2002). Invariant features from interest point groups, In *Proc. of British Machine Vision Conf. 2002*, pp. 656–665. 13

Calonder, M., Lepetit, V., Strecha, C. and Fua, P. (2010). BRIEF: Binary robust independent elementary features, *Computer Vision – ECCV 2010 Lecture Notes in Computer Science*, Springer, **6314**: 778–792. 11

Cao, Y., Wang, C., Li, Z. and Zhang, L. (2010). Spatial-bag-of-features, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2010*, pp. 3352–3359. 118

Capel, D. P. (2005). An effective bail-out test for RANSAC consensus scoring, In *Proc. of British Machine Vision Conf. 2005*, BMVA, pp. 78.1–78.10. 105

Chisholm, E. and Kolda, T. (1999). New term weighting formulas for the vector space method in information retrieval, *Technical Report TN 37831-6367*, Computer Science and Mathematics Division, Oak Ridge National Laboratory, Oak Ridge. 33

Chum, O. and Matas, J. (2002). Randomized RANSAC with $T_{d,d}$ test, *Image and Vision Computing*, **22**(10): 837–842. 105

Chum, O. and Matas, J. (2005). Matching with PROSAC - Progressive Sample Consensus, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2005*, **1**: 220–226. 104

Chum, O. and Matas, J. (2008a). Optimal randomized RANSAC, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**(8): 1472–82. 105

Chum, O. and Matas, J. (2010). Large-Scale Discovery of Spatially Related Images, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(2): 371–377. 119

Chum, O. and Matas, J. (2012). Fast computation of min-Hash signatures for image collections, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2012*, pp. 3077–3084. 124

Chum, O., Matas, J. and Kittler, J. (2003). Locally optimized RANSAC, *Pattern Recognition*, *Lecture Notes in Computer Science*, Springer, **2781**: 236–243. 105

Chum, O., Perdoch, M. and Matas, J. (2009). Geometric min-Hashing: Finding a (thick) needle in a haystack, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 17–24. 119, 123, 126, 131

Chum, O., Philbin, J., Isard, M. and Zisserman, A. (2007). Scalable near identical image and shot detection, In *Proc. of ACM Int. Conf. on Image and Video Retrieval 2007*, pp. 549–556. 118, 124

Chum, O., Philbin, J. and Zisserman, A. (2008). Near duplicate image detection: min-hash and tf-idf weighting, In *Proc. of British Machine Vision Conf. 2008*, BMVA, pp. 50.1-50.10. 118, 124, 126

Cour, T., Zhu, S., Han, T., Wang, X. and Yang, M. (2011). Contextual weighting for vocabulary tree based image retrieval, In *Proc. of IEEE Int. Conf. on Computer Vision 2011*, pp. 209–216. 33

Cranston, C. and Samet, H. (2007). Indexing point triples via triangle geometry, In *Proc. of IEEE Int. Conf. on Data Engineering 2007*, pp. 936–945. 82

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2005*, **1**: 886–893. 16

Datta, R., Joshi, D., Li, J. and Wang, J. (2008). Image retrieval: ideas, influences, and trends of the new age, *ACM Computing Surveys*, **40**(2): 1–60. 30

Day, W. H. and Edelsbrunner, H. (1984). Efficient algorithms for agglomerative hierarchical clustering methods, *Journal of Classification*, **1**(1): 7–24. 19

Delhumeau, J., Gosselin, P., Jegou, H. and Perez, P. (2013). Revisiting the VLAD image representation, In *Proc. of ACM Multimedia 2013*, pp. 653–656. 15

Dempster, A., Laird, N. and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, **39**(1): 1–38. 56, 58

Deng, J., Dong, W., Socher, R., Li, L., Li, K. and Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 248–255. 37

Deselaers, T., Keysers, D. and Ney, H. (2008). Features for image retrieval: an experimental comparison, *Information Retrieval*, **11**(2): 77–107. 10

Everingham, M., Gool, L. V., Van Gool, L., Williams, C., Winn, J. and Zisserman, A. (2009). The Pascal Visual Object Classes (VOC) Challenge, *Int. Journal of Computer Vision*, **88**(2): 303–338. 37

Fellbaum, C. (1998). WordNet: An electronic lexical database, *Technical report*, Cambridge, MA: MIT Press. 65

Felzenszwalb, P., Girshick, R., McAllester, D. and Ramanan, D. (2010). Object detection with discriminatively trained part-based models, *IEEE Transactions on Pattern Analysis* *and Machine Intelligence*, **32**(9): 1627–1645. 16, 68, 80

Fergus, R., Fei-fei, L., Perona, P. and Zisserman, A. (2005). Learning object categories from Google's image search, In *Proc. of IEEE Int. Conf. on Computer Vision 2005*, **2**: 1816–1823. 53

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM*, ACM, **24**(6): 381–395. 102, 104

Forgy, E. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications, *Biometrics*, **21**: 768–769. 17

Frey, B. J. and Dueck, D. (2007). Clustering by passing messages between data points, *Science*, **315**(5814): 972–976. 19

Friedman, J., Bentley, J. and Finkel, R. (1977). An algorithm for finding best matches in logarithmic expected time, *ACM Transactions on Mathematical Software*, **3**: 209–226. 24

Fu, J., Wang, J. and Lu, H. (2010). Effective logo retrieval with adaptive local feature selection, In *Proc. of ACM Multimedia 2010*, pp. 971–974. 81

Grabner, M., Grabner, H. and Bischof, H. (2006). Fast approximated SIFT, *Computer Vision – ACCV 2006 Lecture Notes in Computer Science*, Springer, **3851**: 918–927. 10, 11

Greif, T., Hörster, E. and Lienhart, R. (2008). Correlated topic models for image retrieval, *Technical report*, Report 2008-09, Department of Computer Science, Augsburg University, July 2008. 53

Harris, C. and Stephens, M. (1988). A Combined Corner and Edge Detector, In *Proc. of Alvey Vision Conf. 1988*, pp. 147–151. 13

Hartley, R. and Zisserman, A. (2003). *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press. 98, 99, 100, 101, 107, 108

Hofmann, T. (1999). Probabilistic latent semantic analysis, In *Proc. of Conf. on Uncertainty in Artificial Intelligence*, Morgan Kaufmann Publishers Inc., pp. 289–296. 51, 53, 54

Hofmann, T. (2001). Unsupervised learning by probabilistic Latent Semantic Analysis, *Journal Machine Learning, Springer*, **42**(1-2): 177–196. 51, 52, 53, 54, 56

Hörster, E. (2009). *Topic models for image retrieval on large-scale databases*, PhD thesis, Augsburg University. 37, 53, 55, 59

Hörster, E., Greif, T., Lienhart, R. and Slaney, M. (2008). Comparing local feature descriptors in pLSA-based image models, *Pattern Recognition, Lecture Notes in Computer Science, Proc. of the DAGM Symposium*, **5096**: 446–455. 15, 53, 64

Hörster, E., Lienhart, R. and Slaney, M. (2007). Image retrieval on large-scale image databases, In *Proc. of ACM Int. Conf. on Image and Video Retrieval 2007*, pp. 17–24. 53

Hörster, E., Lienhart, R. and Slaney, M. (2008). Continuous visual vocabulary models for pLSA-based scene recognition, *In Proc. of ACM Int. Conf. on Content-based Image and Video Retrieval 2008*, pp. 319–328. 56

Jégou, H., Douze, M. and Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search, *Computer Vision – ECCV 2008 Lecture Notes in Computer Science*, Springer, **5302**: 304–317. 82

Jégou, H., Douze, M. and Schmid, C. (2009a). Improving bag-of-features for large scale image search, *Int. Journal of Computer Vision*, **87**(3): 316–336. 33, 82, 109, 117, 123, 133

Jégou, H., Douze, M. and Schmid, C. (2009b). On the burstiness of visual elements, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 1169–1176. 33, 125, 128

Jégou, H., Douze, M. and Schmid, C. (2009c). Packing bag-of-features, In *Proc. of IEEE Int. Conf. on Computer Vision 2009*, pp. 2357–2364. 118

Jia, Y., Wang, J., Zeng, G., Zha, H. and Hua, X. (2010). Optimizing kd-trees for scalable visual descriptor indexing, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2010*, pp. 3392–3399. 23

Jiang, Y., Meng, J. and Yuan, J. (2011). Grid-based local feature bundling for efficient object search and localization, In *Proc. of IEEE Int. Conf. on Image Processing 2011*, pp. 113–116. 81

Joly, A. and Buisson, O. (2009). Logo retrieval with a contrario visual query expansion, In *Proc. of ACM Multimedia 2009*, pp. 581–584. 81

Kalantidis, Y., Pueyo, L. G., Trevisiol, M., Van Zwol, R. and Avrithis, Y. (2011). Scalable triangulation-based logo recognition, In *Proc. of ACM Int. Conf. on Multimedia Retrieval 2011*, Article No. 20. 82

Kanungo, T., Member, S., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R. and Wu, A. Y. (2002). An efficient k-means clustering algorithm: analysis and implementation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**(7): 881–892. 23

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C., Silverman, R. and Wu, A. Y. (2000). The analysis of a simple k-means clustering algorithm, In *Proc. of Symposium on Computational Geometry*, ACM, pp. 100–109. 23

Kennedy, L., Naaman, M., Ahern, S., Nair, R. and Rattenbury, T. (2007). How Flickr Helps us Make Sense of the World: Context and Content in Community-Contributed Media Collections, In *Proc. of ACM Multimedia 2007*, pp. 631–640. 64

Lebeda, K., Matas, J. and Chum, O. (2012). Fixing the locally optimized RANSAC, In *Proc. of British Machine Vision Conf. 2012*, BMVA, pp. 95.1–95.11. 104, 105, 108, 109

Lee, D., Ke, Q. and Isard, M. (2010). Partition min-hash for partial duplicate image discovery, *Computer Vision – ECCV 2010 Lecture Notes in Computer Science*, Springer, **6311**: 648–662. 81, 119, 121, 126

Lejsek, H., Ásmundsson, F. H., Jónsson, B. T. and Amsaleg, L. (2006). Scalability of local image descriptors, In *Proc. of ACM Multimedia 2006*, pp. 589–598. 10, 15

Letessier, P., Buisson, O. and Joly, A. (2011). Consistent visual words mining with adaptive sampling, In *Proc. of ACM Int. Conf. on Multimedia Retrieval 2011*, Article No. 49. 81

Leutenegger, S., Chli, M. and Siegwart, R. Y. (2011). BRISK: Binary robust invariant scalable keypoints, In *Proc. of IEEE Int. Conf. on Computer Vision 2011*, pp. 2548–2555. 10, 11

Li, B., Xiao, R., Li, Z. and Cai, R. (2011). Rank-SIFT: Learning to rank repeatable local interest points, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2011*, pp. 1737–1744. 10

Li, Y., Wang, W. and Gao, W. (2006). A robust approach for object recognition, *Advances in Multimedia Information Processing - PCM 2006 Lecture Notes in Computer Science*, Springer, **4261**: 262–26. 53

Lienhart, R., Romberg, S. and Hörster, E. (2009). Multilayer pLSA for multimodal image retrieval, In *Proc. of ACM Int. Conf. on Image and Video Retrieval 2009*, Article No. 9. 6, 52, 71

Lienhart, R. and Slaney, M. (2007). pLSA on large scale image databases, *In Proc. of IEEE Int. Conf. on Acoustics, Speech and Signal Processing 2007*, **4**: IV/1217–1220. 53, 64

Lindeberg, T. (1993). Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention, *Int. Journal of Computer Vision*, **11**(3): 283–318. 12

Lindeberg, T. (1994). Scale-space theory: A basic tool for analyzing structures at different scales, *Journal of Applied Statistics*, **21**(1-2): 225–270. 12

Lindeberg, T. (2007). Scale-space, *Wiley Encyclopedia of Computer Science and Engineering*, John Wiley & Sons, **4** 2495–2504. 12

Lloyd, S. (1982). Least squares quantization in PCM, *IEEE Transactions on Information Theory*, **28**(2): 129–137. 17, 19

Lowe, D. G. (1999). Object recognition from local scale-invariant features, In *Proc. of IEEE Int. Conf. on Computer Vision 1999*, **2**: 1150–1157. 10, 11

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints, *Int. Journal of Computer Vision*, **60**(2): 91–110. 10, 11, 23, 82

Macqueen, J. (1967). Some methods for classification and analysis of multivariate observations, *Berkeley symposium on Mathematical Statistics and Probability 1965*, **1**: 281–297. 17

Mair, E., Hager, G., Burschka, D., Suppa, M. and Hirzinger, G. (2010). Adaptive and generic corner detection based on the accelerated segment test, *Computer Vision – ECCV 2010 Lecture Notes in Computer Science*, Springer, **6312**: 183–196. 10

Márquez-Neila, P., Miro, J. G., Buenaposada, J. M. and Baumela, L. (2008). Improving RANSAC for fast landmark recognition, In *Proc. of IEEE Conf. on Computer Vision and*

Pattern Recognition Workshops 2008, pp. 1–8. 105

Matas, J., Chum, O., Urba, M. and Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions, *Image and Vision Computing*, **22**(10): 761–767. 10

Mikolajczyk, K. and Schmid, C. (2004). Scale & affine invariant interest point detectors, *Int. Journal of Computer Vision*, **60**(1): 63–86. 136

Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T. and Van (2005). A comparison of affine region detectors, *Int. Journal of Computer Vision*, **65**(1-2):43–72. 10, 82

Monay, F. and Gatica-Perez, D. (2004). pLSA-based image auto-annotation: constraining the latent space, *In Proc. of ACM Multimedia 2004*, pp. 348–351. 53, 56

Morel, J. and Yu, G. (2009). ASIFT: A new framework for fully affine invariant image comparison, *SIAM Journal on Imaging Sciences*, **2**(2): 438–469. 135

Muja, M. and Lowe, D. G. (2009). Fast approximate nearest neighbors with automatic algorithm configuration, In *Proc. of Int. Conf. on Computer Vision Theory and Application 2009*, INSTICC Press, pp. 331–340. 23, 25, 28

Myatt, D., Torr, P., Nasuto, S., Bishop, J. and Craddock, R. (2002). NAPSAC: High noise, high dimensional robust estimation-it's in the bag, In *Proc. of British Machine Vision Conf. 2002*, BMVA, pp. 44.1–44.10. 104, 123

Ni, K., Jin, H. and Dellaert, F. (2009). GroupSAC: Efficient consensus in the presence of groupings, In *Proc. of IEEE Int. Conf. on Computer Vision 2009*, pp. 2193–2200. 104

Nistér, D. (2003). Preemptive RANSAC for live structure and motion estimation, In *Proc. of IEEE Int. Conf. on Computer Vision 2003*, pp. 199–206. 105

Nistér, D. and Stewénius, H. (2006). Scalable recognition with a vocabulary tree, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2006*, **2**: 2161–2168. 20, 40, 46, 127

Nowak, E., Jurie, F. and Triggs, B. (2006). Sampling Strategies for Bag-of-Features Image Classification, *Computer Vision – ECCV 2006 Lecture Notes in Computer Science*, Springer, **3954**: 490–503. 15

Page, L., Brin, S., Motwani, R. and Winograd, T. (1999). The PageRank citation ranking: Bringing order to the web, *World Wide Web Internet And Web Information Systems*, pp. 1–17. 3, 54

Park, J. H., Park, K. W., Baeg, S. H. and Baeg, M. H. (2008). π-SIFT: A photometric and scale invariant feature transform, In *Proc. of IEEE Int. Conf. on Pattern Recognition 2008*, pp. 1–4. 10

Perdoch, M., Chum, O. and Matas, J. (2009). Efficient representation of local geometry for large scale object retrieval, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 9–16. 82

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2007*, pp. 1–8. 23, 25, 26, 28, 37, 39, 97, 106, 112, 113, 115, 127, 133

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A. (2008). Lost in quantization: Improving particular object retrieval in large scale image databases, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2008*, **9**(14): 15–17. 26

Poullot, S., Crucianu, M. and Satoh, S. (2009). Indexing local configurations of features for scalable content-based video copy detection, *ACM Workshop on Large-Scale Multimedia Retrieval and Mining 2009*, pp. 43–50. 82

Rabin, J. and Delon, J. (2010). MAC-RANSAC: a robust algorithm for the recognition of multiple objects, *Symposium on 3D Data Processing, Visualization and Transmission 2010*. 105

Raguram, R., Chum, O., Pollefeys, M., Matas, J. and Frahm, J.-M. (2013). USAC: A universal framework for Random Sample Consensus, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35**(8): 2022–38. 105

Revaud, J., Douze, M. and Schmid, C. (2012). Correlation-based burstiness for logo retrieval, In *Proc. of ACM Multimedia 2012*, pp. 965–968. 139, 140

Richter, F., Romberg, S., Hörster, E. and Lienhart, R. (2010). Multimodal ranking for image search on community databases, In *Proc. of ACM Int. Conf. on Multimedia Information Retrieval 2010*, pp. 63–72. 54

Richter, F., Romberg, S., Hörster, E. and Lienhart, R. (2012). Leveraging community metadata for multimodal image ranking, *Multimedia Tools and Applications*, **56**(1): 35–62. 54

Ries, C. X., and Romberg, S. and Lienhart, R. (2010). Towards universal visual vocabularies, In *Proc. of IEEE Int. Conf. on Multimedia and Expo 2010*, pp. 1067–1072. 53

Rigoutsos, I., Wolfson, H. and Watson, I. B. M. T. J. (1997). Geometric Hashing, *IEEE Computational Science & Engineering*, **4**(4). 82

Rodehorst, V. and Hellwich, O. (2006). Genetic algorithm sample consensus (GASAC)-a parallel strategy for robust parameter estimation, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition Workshop 2006*, p. 103. 104

Rodner, E. and Denzler, J. (2009). Randomized probabilistic latent semantic analysis for scene recognition, In *Proc. of the Iberoamerican Conf. on Pattern Recognition 2009: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pp. 945–953. 53

Romberg, S., August, M., Ries, C. X. and Lienhart, R. (2012). Robust feature bundling, *Advances in Multimedia Information Processing – PCM 2012, Lecture Notes in Computer Science*, Springer, **7674**: 45–56. 6, 117, 127

Romberg, S., Garcia Pueyo, L., Lienhart, R. and van Zwol, R. (2011). Scalable logo recognition in real-world images, In *Proc. of ACM Int. Conf. on Multimedia Retrieval 2011*, Article

No. 25. 6, 41, 127, 139, 140

Romberg, S., Hörster, E. and Lienhart, R. (2009). Multimodal pLSA on visual features and tags, In *Proc. of IEEE Int. Conf. on Multimedia and Expo 2009*, pp. 414–417. 6, 52

Romberg, S. and Lienhart, R. (2013a). Bundle min-Hashing, *Int. Journal of Multimedia Information Retrieval*, Springer, **2**(4): 243–259. 6, 117

Romberg, S. and Lienhart, R. (2013b). Bundle min-Hashing for Logo Recognition, In *Proc. of ACM Int. Conf. on Multimedia Retrieval 2013*, pp. 113–120. 6, 97, 117

Romberg, S., Lienhart, R. and Hörster, E. (2012). Multimodal image retrieval, *Int. Journal of Multimedia Information Retrieval*, Springer, **1**(1): 31–44. 6, 52

Rosten, E. and Drummond, T. (2006). Machine learning for high-speed corner detection, *Computer Vision – ECCV 2006 Lecture Notes in Computer Science*, Springer, **3951**: 430–443. 10

Rousseeuw, P. J. (1984). Least Median of Squares Regression, *Journal of the American Statistical Association*, **79**(388): 871. 104, 111

Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF, In *Proc. of IEEE Int. Conf. on Computer Vision 2011*, pp. 2564–2571. 10, 11

Salton, G., Wong, A. and Yang, C. (1975). A vector space model for automatic indexing, *Communications of the ACM*, **18**(11): 613–620. 31

Sattler, T., Leibe, B. and Kobbelt, L. (2009). SCRAMSAC: Improving RANSAC's efficiency with a spatial consistency filter, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 2090–2097. 105

Schindler, G., Brown, M. and Szeliski, R. (2007). City-scale location recognition, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2007*, pp. 1–7. 21, 28, 33

Selim, S. Z. and Alsultan, K. (1991). A simulated annealing algorithm for the clustering problem, *Pattern Recognition*, **24**(10): 1003–1008. 19

Shechtman, E. and Irani, M. (2007). Matching local self-similarities across images and videos, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2007*, pp. 1–8. 11

Silpa-Anan, C. and Hartley, R. (2008). Optimised kd-trees for fast image descriptor matching, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2008*, pp. 1–8. 23, 25

Sivic, J., Russell, B. C., Zisserman, A., Freeman, W. T. and Efros, A. A. (2008). Unsupervised discovery of visual object class hierarchies, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2008*, pp. 1–8. 53

Sivic, J. and Zisserman, A. (2003). Video Google: a text retrieval approach to object matching in videos, In *Proc. of IEEE Int. Conf. on Computer Vision 2003*, pp. 1470–1477. 17, 28, 30, 33, 81, 118

Smeulders, A., Worring, M., Santini, S., Gupta, A. and Jain, R. (2000). Content-based image retrieval at the end of the early years, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(12): 1349–1380. 30

Stewart, C. (1995). MINPRAN: A new robust estimator for computer vision, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **17**(10). 104

Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2004*, pp. 506–513. 11

Tirilly, P., Claveau, V. and Gros, P. (2009). A review of weighting schemes for bag of visual words image retrieval, *Technical report*, PI-1927, TEXMEX - INRIA - IRISA, Rennes Cedex, France, April 2009. 33

Toews, M. and Wells, W. W. (2009). SIFT-Rank: Ordinal description for invariant feature correspondence, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 172–177. 10

Tolias, G. and Avrithis, Y. (2011). Speeded-up, relaxed spatial matching, In *Proc. of IEEE Int. Conf. on Computer Vision 2011*, pp. 1653–1660. 98, 114

Torii, A., Sivic, J., Pajdla, T. and Okutomi, M. (2013). Visual Place Recognition with Repetitive Structures, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2013*, pp. 883–890. 33

Torr, P. and Zisserman, A. (1998). Robust computation and parametrization of multiple view relations, In *Proc. of IEEE Int. Conf. on Computer Vision 1998*, pp. 727–732. 108

Torr, P. and Zisserman, A. (2000). MLESAC: A new robust estimator with application to estimating image geometry, *Computer Vision and Image Understanding*, **78**(1): 138–156. 104, 108

Torralba, A., Fergus, R. and Freeman, W. T. (2008). 80 million tiny images: a large dataset for non-parametric object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **30**(11): 1958–1970. 37

Tuytelaars, T. (2010). Dense interest points, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2010*, pp. 2281–2288. 11

Tuytelaars, T. and Mikolajczyk, K. (2008). Local invariant feature detectors: a survey, *Foundations and Trends in Computer Graphics and Vision*, **3**(3): 177–280. 11

Uijlings, J. R. R., Smeulders, A. W. M. and Scha, R. J. H. (2009). Real-time bag of words, approximately, In *Proc. of ACM Int. Conf. on Image and Video Retrieval 2009*, Article No. 6. 16

van de Sande, K. E. A., Gevers, T. and Snoek, C. G. M. (2010). Evaluating color descriptors for object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(9): 1582–1596. 11

Vedaldi, A. and Fulkerson, B. (2010). VLFeat: An open and portable library of computer vision algorithms, In *Proc. of ACM Multimedia 2010*, pp. 1469–1472. 16

Wu, Z., Ke, Q., Isard, M. and Sun, J. (2009). Bundling features for large scale partial-duplicate web image search, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 25–32. 82, 118

Xiao, J., Hays, J., Ehinger, K., Oliva, A. and Torralba, A. (2010). Sun database: Large-scale scene recognition from abbey to zoo, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2010*, pp. 3485–3492. 16, 37

Zhang, R., Zhang, L., Wang, X.-J. and Guan, L. (2011). Multi-feature pLSA for combining visual features in image annotation, In *Proc. of ACM Multimedia 2011*, pp. 1513–1516. 54

Zhang, S., Tian, Q., Hua, G., Huang, Q. and Li, S. (2009). Descriptive visual words and visual phrases for image applications, In *Proc. of ACM Multimedia 2009*, p. 75. 118

Zhang, Y. and Chen, T. (2009). Efficient kernels for identifying unbounded-order spatial features, In *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2009*, pp. 1762–1769. 133

Zhang, Z. (1997). Parameter estimation techniques: A tutorial with application to conic fitting, *Image and Vision Computing*, **15**: 59–76. 101

Zheng, L., Wang, S., Liu, Z. and Tian, Q. (2013). Lp-norm IDF for large scale image search, *In Proc. of IEEE Conf. on Computer Vision and Pattern Recognition 2013*, pp. 1626–1633. 33

Zitnick, C., Sun, J., Szeliski, R. and Winder, S. (2007). Object instance recognition using triplets of feature symbols, *Technical Report, MSR-TR-2007-53*, Microsoft Research. 82