FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

Efficient Retrieval of Near-Duplicate Images

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Visual Computing

eingereicht von

Thomas Pönitz
Matrikelnummer 0425863

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Privatdoz. Dipl.-Ing. Dr.techn. Martin Kampel

Wien, 21.08.2013

(Unterschrift Verfasser) (Unterschrift Betreuung)

Erklärung zur Verfassung der Arbeit

Thomas Pönitz
Hütteldorfer Straße 168/41, 1140 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasser)

Acknowledgements

This thesis was partially funded by the Austrian Science Fund (FWF): TRP140-N23, project ILAC (Image-based cLassification of Ancient Coins).

Abstract

Due to the increasing flood of digital images and the overall increase of storage capacity, large scale image databases are common these days. Managing such a vast number of digital images is not trivial and certain problems arise.

A specific problem is the task of finding near-duplicate images in such image databases. A near-duplicate image is not only a bit exact copy of a given original image, but also a modified version of the image after specific image transformations. Practically this means a near-duplicate image retrieval system can be seen as a database that can be queried with images and retrieves corresponding originals. The challenge is to develop an image description, that is robust to said image transformations. Furthermore a similarity measure for image descriptions is needed. This similarity should either be calculable very fast or image descriptions should be indexable. In the first case linear database scans are applicable, while in the second case an efficient search structure can be built.

The Bags of Visual Words method, in analogy to the Bags of Words method in text document retrieval, has proven to be particularly suited. It represents every image as a set of visual word frequencies which correspond to keyword frequencies. A visual word is derived from a local visual feature. State of the art methods often rely on SIFT features which have the drawback of relatively high computational costs. To overcome this drawback recently introduced binary features are considered as replacement in this thesis. Binary features are simply bit strings and allow to process several steps of the Bags of Visual Words method more efficiently. Generating visual words with sufficient precision is very time consuming and an alternative clustering algorithm, called kShifts, is examined in this thesis. Furthermore the established kMeans algorithm is adapted to cluster binary features and compared with the kShifts algorithm.

For evaluation the implemented algorithm is tested with commonly available image sets and compared with state of the art methods. Additionally a specific use-case in the form of a press image set consisting of approximately 1,000,000 high quality press images is studied.

Kurzfassung

Aufgrund der kontinuierlich steigenden Menge an digitalen Bildern und dem ständigen Wachstum an verfügbarer Speicherkapazität, sind umfangreiche Bilddatenbanken weit verbreitet. Die Verwaltung einer großen Menge an digitalen Bildern ist jedoch nicht trivial und mit diversen Problemen verbunden.

Eine spezielle Herausforderung ist eine Bilddatenbank auf nahezu identische Bilder zu durchsuchen. Als nahezu identisches Bild bezeichnet man jede veränderte Version, bei der das ursprüngliche Bild bestimmten Transformationen unterworfen wurde. Dies bedeutet, dass ein dafür konzipiertes System Bilder als Suchanfragen akzeptiert und gegebenenfalls entsprechende Originale zurückliefert. Die Schwierigkeit dabei ist eine Bildrepräsentation zu finden die robust in Bezug auf diese Transformationen ist. Gleichzeitig soll die Ähnlichkeit zweier Repräsentationen effizient zu berechnen sein oder die Repräsentationen sollten indizierbar sein. Im ersten Fall kann die Datenbank linear durchsucht werden, während im zweiten Fall eine effiziente Suchstruktur aufgebaut werden kann.

Die Bags of Visual Words Methode, in Analogie zu der Bags of Words Methode für die Textdokument-Suche, hat sich als geeignet herausgestellt. Dabei wird ein Bild duch die Häufigkeiten seiner enthaltenen visuellen Worte beschrieben. Diese visuellen Worte werden von lokalen Bildmerkmalen abgeleitet. Methoden auf dem aktuellen Stand der Wissenschaft verwenden SIFT Merkmale, welche den Nachteil einer aufwendigen Berechnung haben. Um dem entgegenzuwirken werden in dieser Arbeit SIFT Merkmale durch binäre Merkmale ersetzt. Binäre Merkmale bestehen aus einer einfachen Liste an Bits und ermöglichen bestimmte Schritte der Bags of Visual Words Methode effizienter durchzuführen. Das Erzeugen der visuellen Worte ist einer dieser Schritte und äußerst zeitaufwändig. Ein alternativer Clustering-Algorithmus, kShifts, wird untersucht, um diesen Schritt effizienter durchzuführen. Weiters wird der kMeans Algorithmus für binäre Merkmale angepasst und mit kShifts verglichen. Für die Evaluierung des vorgestellten Systems werden frei verfügbare Bilddatenbanken verwendet. Ein Vergleich mit aktuellen Methoden wird gezogen. Zusätzlich wird eine Bilddatenbank mit circa 1,000,000 Pressebildern als praxisorientierter Anwendungsfall untersucht.

Contents

CHAPTER 1

Introduction

According to the "Internet 2012 in numbers"[1] **7 petabytes** of photo content is added to Facebook[2] every month. This equals **300 million** photos added per day. As of September 2012, **5 billion** photos were uploaded to Instagram[3] since its start and approximately 58 photos are uploaded every second since then.

Because of this increasing flood of digital images and the overall increase of available storage capacity, large scale image databases are common these days. Managing such a vast number of digital images is not trivial and certain problems arise. The specific problem this thesis deals with, is the task of Near-Duplicate Image Retrieval (NDIR). The term is not strictly defined which is illustrated by quoting two early near-duplicate image retrieval works from 2004. Ke et al. [37] define near-duplicate image retrieval as

> "The problem of matching a slightly altered photograph to its original is termed near-duplicate image detection".

while Zhang and Chang [80] use the term Image Near-Duplicate

> "Image Near-Duplicate (IND) refers to a pair of images in which one is close to the exact duplicate of the other, but different in the capturing conditions, times, rendering conditions, or editing operations".

The first definition includes only altered images while the second considers two different images of the same object or scenery as near-duplicate. This thesis follows the first definition and thus near-duplicate images are the results of applying certain transformations to an image (e.g. rotation, cropping, compression, white balancing, etc.). A more formal definition follows in section 1.2.

[1] `http://royal.pingdom.com/2013/01/16/internet-2012-in-numbers`, accessed August 21, 2013

[2] `http://facebook.com`, accessed August 21, 2013

[3] `http://instagram.com`, accessed August 21, 2013

1.1 Motivation

When looking at the previously stated figures, it is obvious that visually analyzing these amounts of images provides a great challenge. In fact today it has to be seen as unsolved problem and is the aim of current research [27, 29, 33]. Near-duplicate image retrieval is a specific problem of this research domain. To cope with this increasing number of images state of the art near-duplicate image retrieval systems need to become more efficient as well as more discriminative. Therefore existing algorithms need to be improved and new methods have to be developed.

There exist a multitude of possible applications for near-duplicate image retrieval systems. Bueno et al. [8] name a few and the list is extended here:

Detection of Copyright Violations Near-duplicate image retrieval systems can provide an efficient solution to detect copyright violations. In this thesis a specific use case is considered in the form of high quality press images. Press agencies are organizations that sell news which commonly includes images. When these images are made online available by newspapers it is easy to download and redistribute them without permission, thus committing copyright infringement.

Duplicate elimination In large scale image databases unnecessary copies or near-duplicate versions of images may be introduced unnoticed over time. To save storage space and prevent redundant search results, and thus provide a better search experience for the user, near-duplicate image retrieval can be used to prune unwanted images.

Metadata Retrieval Smart-phones are common these days which means that people are equipped with a camera and mobile internet access. A picture taken of a painting can be seen as near-duplicate and with correct retrieval used to gather additional information. This provides an interesting application for cultural institutions.

Image Search Filtering Searching the internet for images of well known motifs leads to highly redundant results. Near-duplicate image retrieval can be used to prune the list of retrieved images.

Image-forgery Detection Digital images can be manipulated to change their perceptual meaning. When original and near-duplicate image are publicly available near-duplicate image retrieval can be used for detection of manipulated images. Figure 1.1 shows a known example for image forgery published by The New York Times[4].

Furthermore a possible application in the field of numismatics is explored. A common problem in this field is the classification of ancient coins [78], which is the topic of ongoing

[4] `http://thelede.blogs.nytimes.com/2008/07/10/in-an-iranian-image-a-missile-too-many`, accessed August 21, 2013

(a) Original image (b) Manipulated image

Figure 1.1: Discovered image forgery concerning an Iranian missile launch.

research[5]. It is determined if near-duplicate image retrieval can provide an applicable solution to this problem.

1.2 Near-Duplicate Image Retrieval

A near-duplicate image retrieval system is principally a content-based image retrieval system with some special characteristics. The term content-based image retrieval was introduced in 1992 by Kato [35] and is most commonly used for such systems. A general data flow scheme for a content-based image retrieval or near-duplicate image retrieval system is shown in figure 1.2. Either type of system is basically an image database. For each image added to the

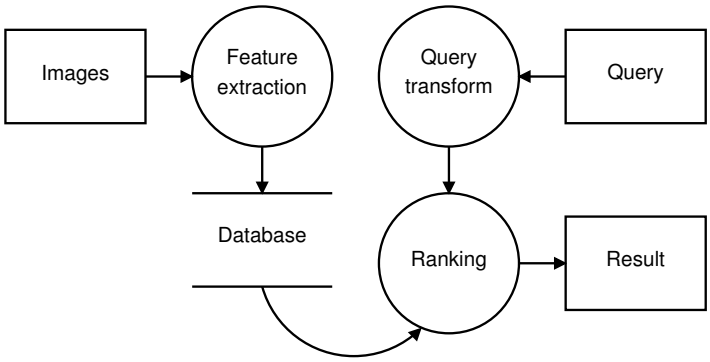


Figure 1.2: Content-based image retrieval: Data flow diagram.

database features are extracted, processed into an image description and stored. To retrieve images a query is submitted to the system which transforms it into a representation suitable for comparison with the stored image descriptions.

The **first** difference between content-based image retrieval and near-duplicate image retrieval systems is the way queries are specified. While there are three basic methods for content-based image retrieval [35], namely query by

[5] `http://www.caa.tuwien.ac.at/cvl/research/ilac`, accessed August 21, 2013

- Example (image)
- Sketch
- Keyword

only query by image is applicable for near-duplicate image retrieval.

The **second** and most important difference is tied to the way images are compared to each other. The meaning of the term similarity in the context of image retrieval is not strictly defined and can vary depending on the nature of a specific system. Often not only the detection of visual similarity is demanded but also of semantic similarity, e.g. two visually different images showing the same object should be treated as similar. This however is not a requirement for near-duplicate image retrieval systems and thus the problem of retrieving near-duplicate images can be solved efficiently relying strictly on visual features [68].

The **third** important difference is the way results are presented. For a content-based image retrieval system resulting images should be ordered by similarity, in contrast to near-duplicate image retrieval where two images can be very similar, e.g. two images of the same object or scenery with a small perspective change, but should not be treated as near-duplicate. Figure 1.3 illustrates image transforms from an original image[6] to near-duplicate images as directed graph. In this thesis an image is considered as near-duplicate to another if both are connected by a directed path. Therefore an ideal near-duplicate image retrieval system would not rank images by similarity but divide them in near-duplicates and non-near-duplicates.

1.3 Problem Description

A near-duplicate image retrieval system has to cope with specific real-world limitations such as limited storage and computational capacities while maintaining criteria regarding quality (e.g. query precision) and run-time to be useful. E.g. Jégou et al. [33] focus on three main criteria in their work – search accuracy, efficiency and memory usage. A complete list of criteria is given in the following:

Setup Effort Includes the computational costs to setup the whole system: feature calculations, training, search structures, etc.

Search Accuracy Describes the quality of the retrieval system. Common are precision and recall

$$precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|} \tag{1.1}$$

$$recall = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{relevant\ documents\}|}. \tag{1.2}$$

[6] `http://commons.wikimedia.org/wiki/File:Gloriette-_Schönbrunn.jpg` and `http://commons.wikimedia.org/wiki/File:Gloriette-IMG_0460.JPG`, accessed August 21, 2013

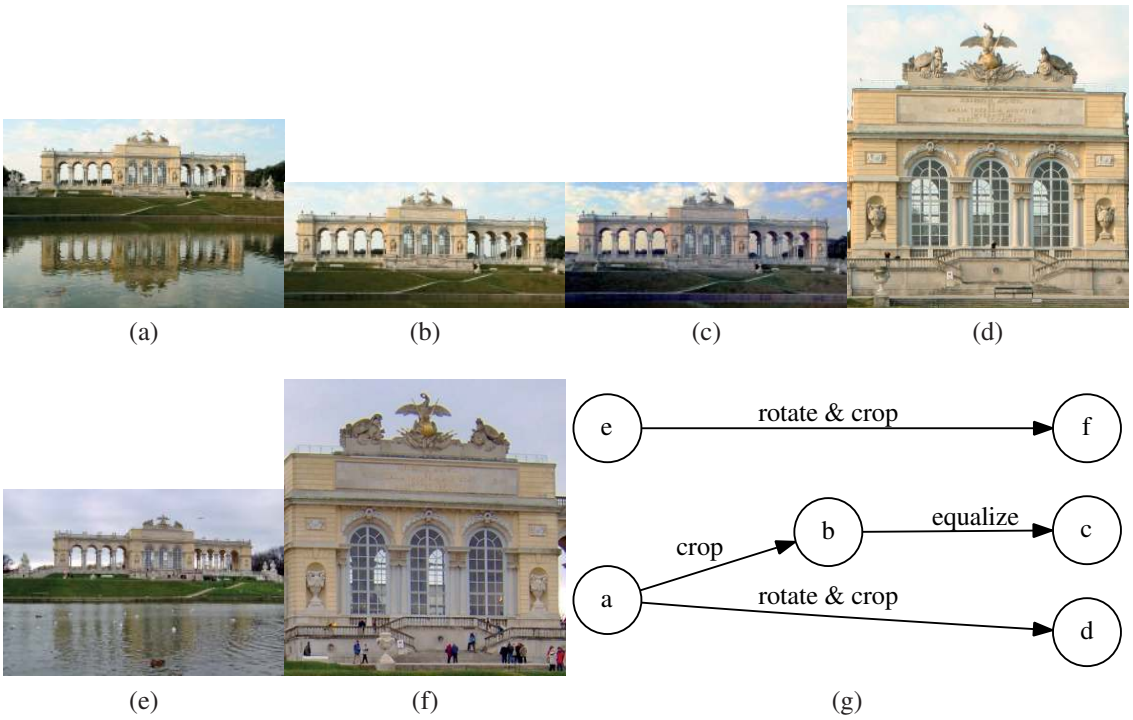


Figure 1.3: (a) - (f) are image examples and (g) shows their relationships. Even so (d) and (f) are visually similar they are not connected and therefore not seen as near-duplicates. Neither are (b) and (d) as they are not connected by a directed path. Practically however (d) could also be derived from (b) and it is indistinguishable if (a) or (b) is its direct ancestor.

Precision tells the percentage of relevant documents in the retrieved documents and recall the percentage of relevant retrieved documents from all relevant documents in the database. Obviously a recall value of 100% is easily achievable by returning all documents in the database. Therefore precision and recall must always be considered together. For near-duplicate image retrieval systems it would be logical not to add near-duplicate images to the database as it would provide no added value and decrease the efficiency of the system. In this case the recall would be either 0 or 1 (always assuming that one relevant image is in the database). Nevertheless in real-world systems near-duplicate images may be added to the database to decrease the false negative rate.

Search Efficiency The computational costs of a query. This is reflected directly by the query-run-time which is highly relevant for the end-user.

Storage Efficiency The amount of on-disk as well as in-memory data that has to be stored for the near-duplicate image retrieval system.

Update Efficiency Determines how costly it is to add or remove images, as such operations may require updates of search structures or retraining.

These criteria have to be considered and balanced against each other when designing near-duplicate image retrieval systems for real-world tasks. For example search accuracy and

search efficiency are directly connected and changing the system to increase one may have a negative impact on the other.

1.4 Aim of the Thesis

The expected result is an image retrieval framework suitable for the task of finding near-duplicate images in large scale image databases (about 1 million images). It should be horizontally and vertically scalable and able to utilize modern hardware resources such as graphics processing units (GPUs). The performance of this system should in general be competitive to state of the art methods and surpass them in specific use cases. The intended contributions are:

- Research on the usage of binary local visual features for near-duplicate image retrieval.
- An optimized implementation for binary feature to feature distance calculations.
- Efficient generation of visual words from binary features through clustering.
- Implementation of a near-duplicate image retrieval system and comparison against state of the art methods.
- A thorough evaluation of the presented system.

1.5 Methodological Approach

The chosen method is the bags of visual words approach (BoW) [18] because it provides an excellent trade-off between search accuracy and efficiency. It was introduced by Csurka et al. [18] for the task of visual categorization. The BoW approach is inspired by text retrieval methods where a document is described by the frequency of its contained keywords. For image retrieval local features are extracted and quantized to so called visual words. An image can then be described similar to a text document – by the frequency of its contained visual words. Because of this similar approach common document retrieval methods can be used to retrieve images.

A detailed look is taken at the three main stages of the BoW approach: feature extraction, generation of visual words and retrieval based on bags of visual words. In contrast to most state of the art methods which use SIFT [43] for feature extraction other local features are considered in detail. For the generation of visual words the clustering algorithm kShifts, which was introduced in [55, 56], is enhanced and compared against other state of the art clustering algorithms. For evaluation an image set provided by the IT department of the Austrian Press Agency[7] consisting of about 1,000,000 images is used. For this image set specific image transformations which are commonly applied by newspaper editors (e.g.

[7] `http://www.apa-it.at`, accessed August 21, 2013

small rotation, cropping, etc.) are applied. The modified images than act as query images to the original database which can be used to detect copyright violations. Additionally for comparison with state of the art approaches public available image sets (INRIA Copydays and Flickr-1M[8] and The Oxford Buildings Dataset[9]) are used as well.

1.6 Outline

An outline of this thesis is given in the following. **Chapter 2** reviews and discusses the state of the art in the field of near-duplicate image retrieval and the methods its based on: **First**, local visual features are discussed, as they provide the basis for various methods. **Second**, clustering in the context of near-duplicate image retrieval is examined, as different methods for clustering local visual features are presented. **Third** and last the designs, advantages and disadvantages of complete near-duplicate image retrieval systems are described. In the following **chapter 3** the methods, languages, concepts and evaluation methods are presented. This includes among other things the statistical methods that are used for the evaluation of the clustering methods and the image sets and methods that are used for the near-duplicate image retrieval evaluation. **Chapter 4** describes the developed solution and explains the implementation in detail: The adaptation of the clustering algorithms for binary features, the overall near-duplicate image retrieval system design and implementation, etc. Furthermore the results and evaluation of the implementation are presented, a critical reflection is given and open issues are discussed. Finally, **chapter 5** summarizes the thesis and provides an outlook on possible future work.

[8] `http://lear.inrialpes.fr/~jegou/data.php#holidays`, accessed August 21, 2013
[9] `http://www.robots.ox.ac.uk/~vgg/data/oxbuildings`, accessed August 21, 2013

CHAPTER 2

State of the Art

Near-duplicate image retrieval is a complex task which can be broken down into specific components, where each component belongs to a field of scientific research on its own:

- Extraction of visual features from images
- Generation of compact image descriptions out of visual features
- Efficient retrieval systems base on image descriptions

The following provides an overview of the significant scientific work regarding each component. First local features describing the visual content of images are discussed. Next an overview of clustering algorithms, which are used to group similar local features, is given. And finally near-duplicate image retrieval systems are described in detail.

2.1 Local Features: Detection and Description

Local features are fundamental to specific computer vision related tasks and following Tuytelaars and Mikolajczyk [69] can be described as follows:

> "A local feature is an image pattern which differs from its immediate neighborhood".

In 1959 Hubel and Wiesel [30] studied the visual cortex of cats and discovered that specific visual stimuli lead to stronger responses than others. This implies that in this context certain visual features are beneficial over others and that the cat's brain is adapted to these. Their work in this field won them the Nobel Prize in Physiology or Medicine in 1981[1]. Similar in computer vision some visual features are better suited for given tasks than others. Even so particular properties are desirable for every visual feature. An exhaustive list from [69] is quoted in the following:

[1] `http://www.nobelprize.org/nobel_prizes/medicine/laureates/1981`, accessed August 21, 2013

“

Repeatability Given two images of the same object or scene, taken under different viewing conditions, a high percentage of the features detected on the scene part visible in both images should be found in both images.

Distinctiveness/informativeness The intensity patterns underlying the detected features should show a lot of variation, such that features can be distinguished and matched.

Locality The features should be local, so as to reduce the probability of occlusion and to allow simple model approximations of the geometric and photometric deformations between two images taken under different viewing conditions (e.g., based on a local planarity assumption).

Quantity The number of detected features should be sufficiently large, such that a reasonable number of features are detected even on small objects. However, the optimal number of features depends on the application. Ideally, the number of detected features should be adaptable over a large range by a simple and intuitive threshold. The density of features should reflect the information content of the image to provide a compact image representation.

Accuracy The detected features should be accurately localized, both in image location, as with respect to scale and possibly shape.

Efficiency Preferably, the detection of features in a new image should allow for time-critical applications.

”

These properties entail the demands on a local feature in the context of near-duplicate image retrieval.

2.1.1 Corner Detection

Corners are among the simplest visual features and were adopted early for several computer vision tasks [28]. Because of their simplicity corners are apt for real-time applications.

In the work of Rosten and Drummond [61] a particular fast and efficient corner detection approach, which combines the Features from Accelerated Segment Test (FAST) detector with a machine learning stage, is presented. FAST is based on a fixed set of simple binary pixel tests. The machine learning stage adapts to the particular appearance of corners of the target application domain. This allows to minimize the pixel tests necessary to perform to classify an image patch as corner or non-corner. The result of the training is converted to simple C-code consisting of nested if-then-else statements which are than compiled.

A detailed comparison to other commonly used detection algorithms (Harris [28], DoG [43], SUSAN [65]) is given, where speed and repeatability are compared. Results show that FAST is faster than the other evaluated algorithms (the speed-up is about $6\times$ for SUSAN, $50\times$ for DoG and $20\times$ for Harris) and leads to high repeatability even outperforming the other algorithms for certain data-sets. One drawbacks is that the results for the speed optimized corner detection model of FAST show a significant decrease in repeatability if noise is introduces to images. with noise. However this and other drawbacks stated in their paper are not relevant in near-duplicate image retrieval as it is most commonly performed on high quality image sets.

2.1.2 SIFT

The Scale Invariant Feature Transform (SIFT) was introduced by Lowe [43]. SIFT aims to be robust against affine transformations, view point changes, addition of noise and changes in illumination. Thus SIFT is applicable for numerous real-world applications and state of the art methods in the fields of image retrieval [70], object tracking [73], object recognition [77], etc. are based on it.

Detection

SIFT has four major stages, namely: Scale-space extrema detection, Keypoint localization, Orientation assignment and Keypoint description.

The first stage is realized with Difference of Gaussians (DoG) where a Gaussian kernel is used to successively blur the original image and rescaled versions of it. When building the difference of consecutive blurred images a stack of DoG images is constructed. The resulting stack forms a 3d volume and is used in the second stage, keypoint localization.

In the 3d volume $3 \times 3 \times 3$ neighbourhoods are tested for local maxima or minima. These local extrema are called keypoints. Additional steps follow to improve keypoint quality where keypoint positions are refined in sub-pixel space as well as in continuous space, stability criteria are applied and unstable keypoints are discarded.

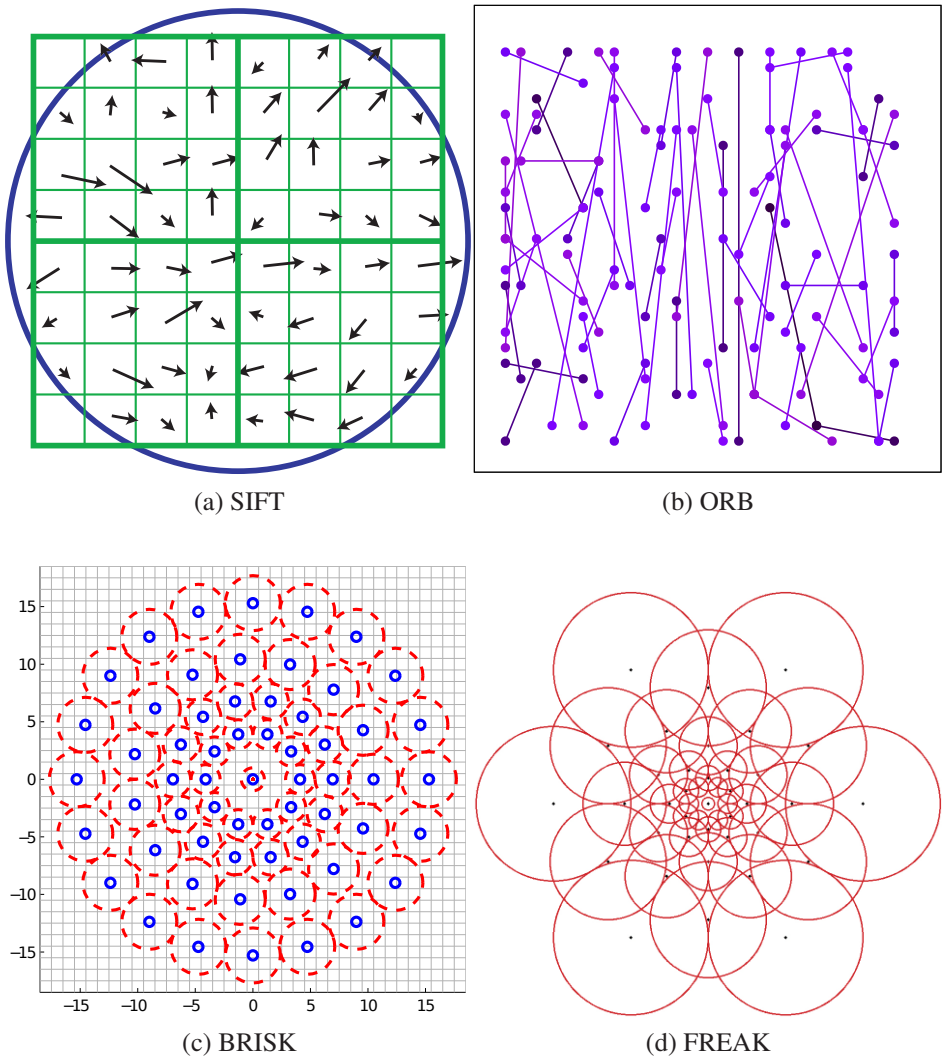
In the third stage the orientation is determined for each remaining keypoint by forming a weighted orientation histogram of the image intensity gradient vectors in a specified neighbourhood of the keypoint. Again unstable keypoints are discarded, which in this case means that no dominant orientation could be calculated.

After the first three stages keypoints with exact locations, scale values (derived from the DoG image each keypoint was detected in) and orientation are present. Based on these keypoints various descriptors (not restricted to SIFT) can be applied to extract visual features.

Description

SIFT features are calculated by partitioning the neighbourhood of the keypoint into regions, with respect to the calculated orientation, and forming a weighted image intensity gradient

histogram for each region. A typical setup consists of 4×4 regions and orientations are quantized to 8 possible orientations (see figure 2.1a). This results in feature vectors with a



(a) SIFT (b) ORB

(c) BRISK (d) FREAK

Figure 2.1: Descriptor sampling locations for SIFT [43], ORB [62], BRISK [38] and FREAK [2]. The (dashed) circles in (c) and (d) indicate the scale value of each sampling location.

dimensionality of $4 \times 4 \times 8 = 128$. According to the needed precision feature vectors are typically stored as 32 bit floating point or 8 bit fixed point values, leading to a vector byte size of 512 or 128.

Evaluation

A comparison of feature detectors is given by Tuytelaars and Mikolajczyk [69] and more recently by Miksik and Mikolajczyk [46], who provide a comparison of feature detectors and descriptors. The results, especially the latter, show that SIFT is still competitive but that there are interesting alternatives especially with lower run-time, which will be discussed in the following. Even so state of the art near-duplicate image retrieval algorithms still rely mostly on SIFT. E.g. [8, 33, 70, 72] use at least the description stage of SIFT, while

the detection stage is sometimes replaced with other scale aware detection algorithms, e.g. Harris-Laplace or Hessian-Laplace [45].

2.1.3 SURF

Given the success of SIFT, algorithms were developed that should build upon its strengths while exceeding it in its weaker spots. One of the major drawbacks of SIFT is its run-time which renders it too slow flow for certain (real-time) applications. Therefore Speeded Up Robust Features were introduced by Bay et al. [5].

Detection

Among the main contributions of SURF is the replacement of DoG with box filters and integral images, approximating second order Gaussian derivatives. Integral images are also known as summed area table and were introduced in the field of computer graphics by Crow [17] in 1984. The integral images allow a very efficient evaluation of the box filters, thus speeding up the keypoint detection stage significantly. For different scales different sizes of the box filters are applied.

Description

For the descriptor stage the orientation of the detected keypoint is calculated first. Therefore the responses for two Haar-wavelets (one with horizontal and the other with vertical orientation) in a circular region around the keypoint are calculated. This can again be done efficiently using integral images. The orientation is derived directly from these responses. If rotation invariance is not needed this step is simply skipped and the simplified algorithm is called U-SURF, the U standing for upright. Similar to the default SIFT setup the region around the keypoint is partitioned into 4×4 regions, again with respect to the calculated orientation. In the resulting regions Haar-wavelets are applied where the responses of the horizontal wavelet are called d_x and the vertical d_y. The responses are weighted and accumulated to form the feature vector

$$\boldsymbol{v} = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|). \tag{2.1}$$

As there are 4×4 regions the whole feature vector has a dimension of $4 \times 4 \times 4 = 64$. According to the needed precision feature vectors are typically stored as 32 bit floating point or 8 bit fixed point values, leading to a vector byte size of 256 or 64. This corresponds to half the size needed for a standard SIFT feature.

Evaluation

A comparison of SURF against other feature detection and description algorithms is provided by Miksik and Mikolajczyk [46]. Additionally to the default SURF setup the SURF detector

is combined with other description algorithms. Results show that SURF's precision and recall is comparable to SIFT while significantly outperforming it in terms of run-time and efficiency.

2.1.4 BRIEF

Binary Robust Independent Elementary Features (BRIEF) were introduced by Calonder et al. [10]. BRIEF is based on binary strings as feature descriptors, which can be computed very efficiently using simple intensity difference tests. BRIEF does not incorporate feature detection. Therefore in the referenced publication, BRIEF is paired with the SURF feature detector and compared against SURF and U-SURF descriptor. Calonder et al. suggest to use BRIEF with a faster detector than SURF as to not diminish the run-time advantage gained by using their descriptor. They specifically suggest to use the detector described by Agrawal et al. [1].

Description

To create the descriptor vector for a given keypoint binary test are performed on the relevant image patch. The formula for test τ on patch $\boldsymbol{p}$ sized $S \times S$ is

$$\tau(\boldsymbol{p}; \boldsymbol{x}, \boldsymbol{y}) \coloneqq \begin{cases} 1 & \text{if } \boldsymbol{p}(\boldsymbol{x}) < \boldsymbol{p}(\boldsymbol{y}) \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

and $\boldsymbol{p}(\boldsymbol{x})$ specifies the pixel intensity of patch $\boldsymbol{p}$ in point $\boldsymbol{x}$. Patch $\boldsymbol{p}$ is smoothed prior to testing to decrease the influence of noise. To construct the feature vector the test is applied to a predefined number of sampling locations and every sample is represented by one bit in the result

$$f_{n_d}(\boldsymbol{p}) \coloneqq \sum_{1 \leq i \leq n_d} 2^{i-1} \tau(\boldsymbol{p}; \boldsymbol{x_i}, \boldsymbol{y_i}). \tag{2.3}$$

For different use cases different levels of precision are possible and the values $n_d = 128, 256$ and 512 are evaluated in [10]. A 512 bit descriptor fits in eight standard 64 bit variables (e.g. C standard integer type *uint64_t*) which makes it efficient to handle in programs. The different sizes are represented by the naming scheme BRIEF-k where k denotes the number of bytes required ($k = n_d/8$). The best sampling locations found follow a specific random distribution around the center of patch $\boldsymbol{p}$. For feature to feature distance calculations the hamming distance is suggested as an especially fast and efficient distance measure – in contrast to SIFT where typically the L2 norm $|\cdot|_2$ is calculated with floating point vectors. This useful property is valid for all binary features.

Evaluation

BRIEF is evaluated against SURF and U-SURF. Although BRIEF is not scale and orientation independent, it is robust to small changes in rotation [10]. Therefore these preconditions are reflected in the image test sets used. Results show that in this setting BRIEF is much faster than SURF while yielding equal or better recognition rates than SURF/U-SURF.

2.1.5 ORB

In [62] the authors Rublee et al. take on the problems of BRIEF [10]: the need of a scale and orientation independent detector and an accompanying descriptor. Given the performances of BRIEF and the corner detector FAST [61], Rublee et al. [62] based their work on these two algorithms. This is also implied by the name: Oriented FAST and Rotated BRIEF (ORB).

Detection

FAST detects corners but also tends to generate unwanted responses on edges and is not scale invariant [61, 62]. Therefore edge responses are filtered out by employing the Harris corner measure from [28]. Furthermore the Harris corner measure is used to sort the corners by there "cornerness" and only keep the N best. Multi-scaled detection is achieved by simply applying the detector to each level of a scale pyramid of the original image. For orientation measure the intensity centroid approach of Rosin [60] is used. This approach defines the moments of a patch, with the coordinate system centered at the corner, as

$$m_{pq} = \sum_{x,y} x^p y^q I(x, y) \tag{2.4}$$

and the position of the centroid as

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right) \tag{2.5}$$

which leads to the orientation of the corner

$$\theta = atan2(m_{10}, m_{01}). \tag{2.6}$$

Description

The ORB descriptor works on the same principle as the BRIEF descriptor with the addition that the sampling points (see figure 2.1b) are rotated accordingly to the calculated orientation. The rotated sample locations are pre-calculated for efficiency. Results show that the original sample locations from BRIEF lose discrimination when used in this orientation independent setup. Therefore Rublee et al. suggest new sampling locations, based on a new training algorithm. The resulting approach is named rBrief.

Evaluation

For evaluation ORB is compared against SIFT and SURF [62]. As expected ORB is significantly faster, one magnitude for SURF and at least two magnitudes for SIFT. In the feature matching tests conducted ORB is on par with SIFT and SURF for one image set and clearly outperforms both in the other.

2.1.6 BRISK

Another detection and description framework called Binary Robust Invariant Scalable Keypoints (BRISK) is presented by Leutenegger et al. [38]. Like ORB, BRISK features scale and orientation invariant keypoints and builds upon binary descriptor strings for efficiency in feature matching applications [38].

Detection

To achieve scale invariance a scale pyramid of the image is built. The FAST corner detector is applied to each level of the pyramid. Similar to SIFT the pyramid is seen as a 3d volume and maxima in respect to the corner measure of FAST are extracted. The keypoint positions are refined in sub-pixel space as well as in continuous scale space. First a 2d quadratic function is fitted into the three related image planes and each maximum evaluated. Second, for the three resulting points a 1d parabola is fitted along the scale axis. The result is the exact position of the local maximum in respect to the FAST corner measure.

Description

Following the principles of [10] a specific sampling pattern is introduced (see figure 2.1c). The image patch corresponding to the detected keypoint is smoothed with a Gaussian kernel to reduce the influence of noise. The parameters for the Gaussian are adapted to the calculated scale value of the keypoint. The sampling locations consist of point pairs used in binary tests. These point pairs (p_i, p_j) are split into two groups according to their distance $\|p_i - p_j\|$ and only the group $\mathcal{L}$ with longer distances are used for orientation estimation. This is done by analyzing the gradients of the image patch

$$\boldsymbol{g} = \begin{pmatrix} g_x \\ g_y \end{pmatrix} = \frac{1}{|\mathcal{L}|} \sum_{(p_i,p_j)\in\mathcal{L}} \boldsymbol{g}(p_i, p_j) \tag{2.7}$$

$$\alpha = atan2(g_y, g_x) \tag{2.8}$$

where $\boldsymbol{g}(p_i, p_j)$ is the gradient at position (p_i, p_j) in the smoothed image patch and α is the orientation of the keypoint. The descriptor on the other hand is built by using only the point

pairs with shorter distances to which rotation α is applied

$$(p_i^\alpha, p_j^\alpha) \in \mathcal{S}. \tag{2.9}$$

One major difference to how BRIEF composes the description vector (formula 2.2) is that sample points have varying scale values. Thus the resulting formula for the bit string is

$$b = \begin{cases} 1 & \text{if } I(\boldsymbol{p}_j^\alpha, \sigma_j) > I(\boldsymbol{p}_i^\alpha, \sigma_i) \\ 0 & \text{otherwise} \end{cases}, \ \forall (\boldsymbol{p}_j^\alpha, \boldsymbol{p}_i^\alpha) \in \mathcal{S} \tag{2.10}$$

where σ is the scale value of a sample point. Default bit strings have a length of 512 which equals the length of BRIEF-64.

Evaluation

For evaluation Leutenegger et al. compare BRISK against SURF and SIFT [38]. The detector stage is evaluated for four different image sets and compared against the SURF detector. On the average BRISK performs equally well as the SURF detector. Finally the combination of BRISK detector and descriptor is compared to SURF and SIFT by means of feature matching. Results vary for each image set but overall the matching performance of BRISK is comparable to those of SURF and SIFT while run-time is significantly lower (about 6.5 times faster for SURF and one magnitude for SIFT).

2.1.7 FREAK

Fast Retina Keypoints (FREAK) is another keypoint descriptor, proposed by Alahi et al. [2]. It is based on BRIEF, ORB and BRISK, which are all binary features. Alahi et al. observe that the binary comparisons to obtain the descriptors are similar to the way ganglion cells in the human retina work.

Description

The sampling pattern for the binary tests are obtained from a training stage similar to that used in [62] and are illustrated in figure 2.1d. A special property of FREAK is that the bits in the description vector follow a specific order, namely coarse to fine. This is achieved by the special layout of the sampling pattern and the fact that samples are obtained from different scales as in [38]. The full descriptor is 512 bits long and split into 128 bit groups for feature matching. Because of the coarse to fine ordering a possible match can be eliminated prematurely if the hamming distance of the first 128 bits is too high. Otherwise the remaining 128 bit pairs are evaluated successively until the full distance is calculated or a certain threshold is exceeded.

Evaluation

FREAK is evaluated against SIFT, SURF and BRISK in a feature matching setup. The BRISK keypoint detector [38] is used for all descriptors to allow a detector neutral comparison. FREAK outperforms the other descriptors in terms of recall and precision. Additionally it is faster than BRISK in extracting the descriptor and especially in the matching stage because of its coarse to fine structure.

2.2 Clustering

Following Pruscha [57] clustering is the task of generating a group structure for a set of objects. While this is a clear and precise definition it does not assume anything about the nature of the objects or the generated group structure. This is underlined by Estivill-Castro [25] who argue, that the notion of "cluster" cannot be precisely defined and that this is the reason for the large number of clustering algorithms. Therefore the only sensible way to discuss clustering is in the context of a given task. In the bags of visual words approach [18] local visual features are extracted from a set of images and the complete feature space is too complex to be used directly. Thus clustering is used as a means of gaining insight into the structure of the feature space in respect to one's images. This information should provide a way to simplify the extracted information to a lower but still sufficient level of complexity.

Regardless of the type of objects a similarity or distance measure is needed to generate a group structure. Furthermore for evaluating clustering algorithms a certain quality measure has to be defined.

Categorization of clustering algorithms is not uniform. Pruscha [57] classifies clustering algorithms into two basic categories: hierarchical and non-hierarchical. Hierarchical is in general more time consuming and the resulting hierarchical structure not necessarily needed for state of the art methods in duplicate-image retrieval. The hierarchical single-link clustering algorithm for example can be implemented with a complexity of $\mathcal{O}(n^2)$ [64] which renders it unfit for large data-sets. Consequently only non-hierarchical clustering algorithms, used in relevant state of the art methods, are discussed.

2.2.1 kMeans

kMeans belongs to the centroid based clustering algorithms which means that for a given data-set k centroids are generated, thus structuring the data-set into Voronoi cells. The standard kMeans algorithm was introduced by Lloyd [42] and can be described in three basic steps:

1. Initialize k centroids

2. Assign each object to its nearest centroid

3. Recalculate centroids by averaging all assigned objects, go to step 2

Typically used termination criteria are:

- centroid movement has ceased
- a specified quality has been reached
- a fixed number of iterations have passed

Given the popularity of kMeans algorithms various improvements have been proposed for the standard algorithm. They aim to make the algorithm faster by approximation [39, 54], deal with the problem of finding good starting points for the centroids [3] or automatically determine the number of centroids [52]. Variants of kMeans are still the method of choice for specific state of the art methods in near-duplicate image retrieval when generating visual words (see section 2.3.2) e.g. [9, 13, 40, 59, 68].

2.2.2 Expectation-Maximization

The Expectation-Maximization (EM) algorithm [21] can be used to generate a Gaussian Mixture Model (GMM). A GMM describes a data-set by a fixed number of Gaussian distributions. Following Reynolds [58] a GMM is a weighted sum of M Gaussian densities defined as

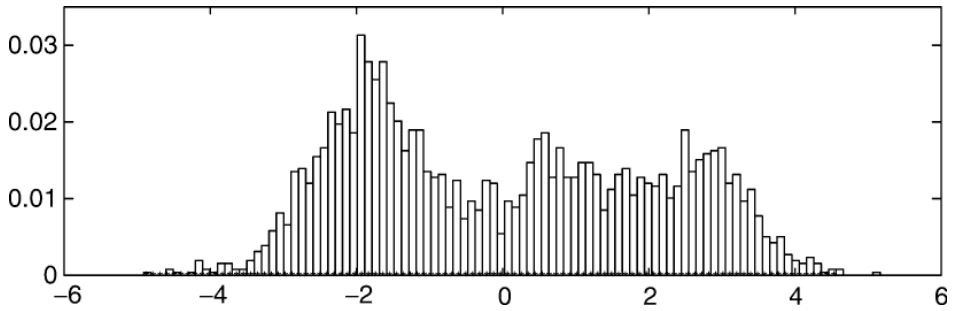
$$p(\boldsymbol{x}|\boldsymbol{\lambda}) = \sum_{i=1}^{M} w_i \; g(\boldsymbol{x}|\boldsymbol{\mu}_i, \Sigma_i) \tag{2.11}$$

where $\boldsymbol{x}$ is a data vector, w_i the weight for mixture i and $g(\boldsymbol{x}|\boldsymbol{\mu}_i, \Sigma_i)$ represents a single Gaussian component with mean vector $\boldsymbol{\mu}_i$ and covariance matrix Σ_i. The parameters of the whole GMM are bundled in $\boldsymbol{\lambda} = \{w_i, \boldsymbol{\mu}_i, \Sigma_i\}, i = 1, 2, \ldots, M$. The mixture weights have to fulfill the constraint $\sum_{i=1}^{M} w_i = 1$. $p(\boldsymbol{x}|\boldsymbol{\lambda})$ is the likelihood function which is a measure how well the model $\boldsymbol{\lambda}$ fits the data $\boldsymbol{x}$ and should therefore be maximized. As an example figure 2.2 shows a 1d histogram and a corresponding GMM with ten components.

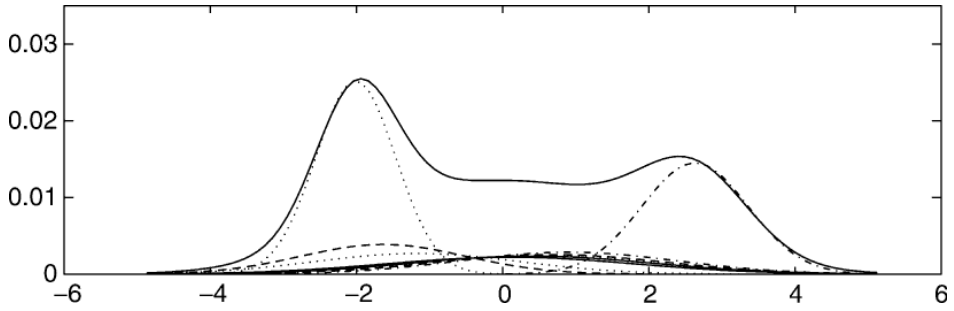
The EM algorithm is iterative and can be described in three basic steps:

1. Initialize model $\boldsymbol{\lambda}$.
2. **Expectation**
   Calculate the likelihood for each data point and distribution pair (Bayes theorem).
3. **Maximization**
   Estimate new model from previous model, data points and their likelihoods so that overall likelihood $p(\boldsymbol{x}|\boldsymbol{\lambda})$ increases.

Step 2 and 3 are iterated until the result is satisfying. Clearly a parallel can be drawn from the EM algorithm to the standard kMeans algorithm as it is in fact a specialized EM variant.



(a) Histogram



(b) Gaussian mixture density

Figure 2.2: A histogram and its corresponding Gaussian mixture model with ten components [58].

The main difference is that kMeans does a hard selection while a GMM generated by the EM algorithm provides a statistical model for the data which can be seen as soft selection. Consequently clustering quality can be derived directly from the likelihood function $p(\boldsymbol{x}|\boldsymbol{\lambda})$. Recently GMMs have been used with great success in duplicate-image retrieval or related fields [33, 53].

2.3 Near-Duplicate Image Retrieval

For an efficient image retrieval system a compact and discriminative image representation is crucial. A chosen image representation deeply affects the properties of a near-duplicate image retrieval system [33]. In the following three major strategies for representing images in near-duplicate image retrieval frameworks are discussed.

2.3.1 Local Features

One possible method to represent an image is directly through its local features. Clearly a drawback of this method is high memory usage and low efficiency. On the other hand it probably provides the best search accuracy as it discards no information after the local visual feature extraction.

Vitaladevuni et al. [70] use this approach when finding near-duplicate images which consist of scans of handwritten and machine typed documents. For each image in the database SIFT [43] features are extracted and stored directly in the database. Likewise features are

extracted from the query images. With these query features a nearest neighbour search is performed in the database and the documents with the best matching features are retrieved. The algorithm is compared to a bags of visual words approach and clearly outperforms it in terms of search accuracy.

Dong et al. [22] rely on direct feature comparison as well and the first stage of their approach is similar to the approach used by Vitaladevuni et al. [70]. The main difference is that not all visual features are used but only those who are discriminative enough. This is done by an entropy measure. For the retrieval stage results are interpreted as graph with each returned image being a node. An adapted graph cut algorithm is used to prune false positives and with the remaining images, query expansion is performed. They show that pruning visual features holding less information significantly improves the search accuracy of their proposed method. Additionally the graph cut combined with query expansion reduces the number of false positives. They compare their approach against BoW with tf-idf classification (see section 2.3.2) and achieve significantly higher recall percentage with the proposed method.

Another method based on nearest neighbour search is proposed by Bueno et al. [8]. First SIFT features are calculated for a "foreground" image set extended with "background" images. Several image transformations are applied to the foreground images and SIFT features are calculated for them as well. A match between a transformed foreground image and an original foreground image is defined as the 1-nearest SIFT feature match, when iterating over all foreground image SIFT features. These matches are grouped into correct and incorrect. For each group a distance histogram is formed from the corresponding nearest neighbour matches. Next a probability density function is fitted to each histogram. In the retrieval phase Bayesian decision theory is combined with the statistical models for the match distances. $P(X)$ is the probability that a match is correct. $P(D|X)$ is the probability that a distance is D if the match is correct and accordingly $P(D|\overline{X})$ for incorrect. This results in the likelihood ratio being

$$L_i = \frac{P(D_i|X) + \epsilon}{P(D_i|\overline{X}) + \epsilon} \tag{2.12}$$

where ϵ is a small constant to prevent division by zero. The probability that a query image is a near-duplicate of database image j is defined as

$$P_j(X|D) = \frac{P(D|X) \times P(X)}{P(D)} = \frac{P(D|X) \times P(X)}{P(D|X) \times P(X) + P(D|\overline{X}) \times P(\overline{X})} \tag{2.13}$$

and can be rewritten as

$$P_j(X|D_i) = \frac{L_i \times P(X)}{L_i \times P(X) + P(\overline{X})}. \tag{2.14}$$

For n matches with distances $D_1, D_2, \ldots, D_n$ the probability is given by

$$P_j(X|D_1 \cap D_2 \cap \cdots \cap D_n) = \frac{\prod_{i=1}^{n} L_i \times P(X)}{\prod_{i=1}^{n} L_i \times P(X) + P(\overline{X})}. \tag{2.15}$$

An exhaustive search of the whole database, in case a query image is not a near-duplicate form any image, is inefficient and prevented by the near-duplicate image retrieval system. To do so $P(Q)$ is defined, as the probability that an image matching the query Q is present in the database. Now if $P(\overline{Q})$ is above a certain threshold the query can be discarded right away. $P(\overline{Q})$ does not have to be highly precise as its only purpose is to increase search efficiency. The relevant probabilities are derived from the same training data, respectively $P(D|\overline{Q})$ is assumed as $P(D|\overline{X})$ and $P(D|Q)$ as $P(D)$. $P(\overline{Q}|D_1 \cap D_2 \cap \cdots \cap D_n)$ can be derived similar as equation 2.15. For a single query one feature after another is selected randomly from the query image and the nearest neighbour from the whole database is retrieved together with the corresponding database image j. A threshold $\mathcal{T}$ is defined and the first image j with $P_j(X|D_1 \cap D_2 \cap \cdots \cap D_n) > \mathcal{T}$ is retrieved. Concurrently $P(\overline{Q}|D_1 \cap D_2 \cap \cdots \cap D_n)$ is updated and if it exceeds a defined threshold the search terminates and no image is retrieved. Bueno et al. [8] compare their approach against a BoW setup and reach a precision of 99.1% in contrast to the BoW setup with 87.2%. This high precision has the consequence of accompanying higher storage, memory and run-time costs.

2.3.2 Bags of Visual Words

The bags of visual words approach (BoW) [18] is currently one of the most commonly used state of the art method [9, 68, 72]. It was introduced by Csurka et al. [18] for the task of visual categorization. The BoW approach is inspired by text retrieval methods where a document is described by the frequency of its contained keywords. For image retrieval local features are extracted and quantized to so called visual words. An image can then be described similar to a text document – by the frequency of its contained visual words. Because of this similar approach common document retrieval methods can be used to retrieve images. Of course the reduction from a set of local features to a histogram is significant but still preserves enough information for many applications. The size of the histogram is fixed meaning every image is described by a vector of the same length which allows for simple distance calculations.

The first step when using the BoW approach is to build the visual words (different names are used too, like visual-codebook, visual-centroids, etc.). The default approach is to extract local visual feature from a training-set. These features are than clustered with a predetermined number of resulting visual words. In general the number of visual words should be as high as acceptable in terms of efficiency, respectively storage and memory usage, as it directly affects search accuracy. Often standard kMeans or a variant of it is the method of choice [13, 18, 59].

The next step is to perform vector quantization to map the feature vectors to the previously obtained visual words. Without a special search structure this requires k distance calculations for every feature, which is time consuming. Note that the same problem is present in clustering algorithms and there exist a number of approaches to counter it. Kanungo et al. [34] approach this by building a kd-tree for the data points when clustering. Arya

and Mount [4] present three vector quantization algorithms and show that significant speed gains can be achieved if it is not required to find the true nearest neighbour. A collection of implemented state of the art algorithms can be found in the Fast Library for Approximate Nearest Neighbours[2] (FLANN) [48]. The results of the vector quantization are used to build the bag (histogram) of visual words. Figure 2.3 shows a schema for the complete process from feature extraction to bags of visual words. The upper half shows the three steps:

1. Feature extraction
2. Calculation of visual words
3. Vector quantization: Assignment of features to visual words

The lower half illustrates the processing of images into bags of visual words. These bags of

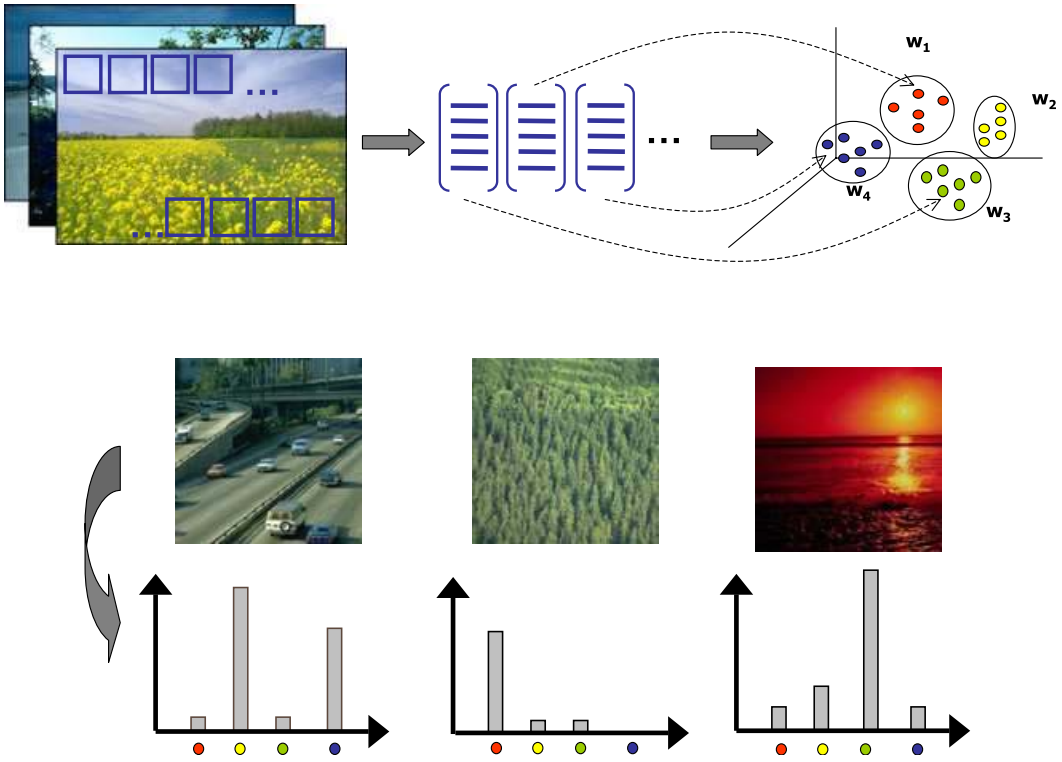


Figure 2.3: The bag of visual words approach [18]: Feature extraction, clustering and vector quantization, bags (histograms) of visual words (figure from [7]).

visual words can further be used for image retrieval. Depending on the specific classification method bags of visual words are normalized. A common classification method is to calculate inverse document frequencies (idf), as in text document retrieval, and use tf-idf weighting for scoring documents [54]:

$$idf(t, D) = log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{2.16}$$

$$tfidf(t, d, D) = tf(t, d) \cdot idf(t, D) \tag{2.17}$$

[2] `http://www.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN`, accessed August 21, 2013

$|D|$ is the number of documents or images, $|\{d \in D : t \in d\}|$ counts the documents $d \in D$ which fulfill $t \in d$. $tf(t, d)$ is the frequency of term t in document d. The result is a weighted term frequency for document d. Terms appearing less often are weighted higher than very common terms to increase their influence

Given the overall success of the BoW approach in different fields several improvements to account for particular tasks have been proposed. In the following specific methods in the field of duplicate-image retrieval are discussed.

Chen et al. propose spatial visual phrases (SVP) [13] to improve the generic BoW approach. As starting point their method follows the basic process up to the mapping of visual features to visual words. From there on they build SVPs by bundling commonly co-occurring visual word pairs with a particular spatial relation. Different visual word pairs as well as different spatial relations result in different SVPs. An image is represented as a vector where each component represents the frequency of a SVP or to phrase it in the context of BoW, a bag of spatial-visual-phrases. They propose an adapted classification method to match the properties of this representation. They evaluate their approach on two image data-sets and compare it against different classification methods for SVPs.

In [72], Wang et al. follow a similar approach. The main difference is that not only phrases for pairs of visual words are generated but also phrases for larger visual word groups (up to 20 spatially related visual words are considered). These more complex phrases are only useful for ranking very similar images as they impose high spatial constraints. Therefore they are only used for ranking the top 10 results.

Representing images as BoW introduces a quantization error when mapping visual features to visual words. Cai et al. [9] try to reduce this error by simply removing every visual feature that is not within a certain distance to its nearest visual word. While at first glance this seems to discard a lot of visual features and therefore information, they underline their assumption by theoretical as well as practical results. Their experiments show that search accuracy increases which is explained by the lower overall quantization error. Furthermore efficiency increases as well because the reduction of visual words leads to sparser BoW which reduces storage and memory requirements.

Hashing

Efficiency is an important property of near-duplicate image retrieval systems. Hashing functions are suitable to improve efficiency in different steps of the BoW approach.

As in [13, 72], visual feature bundling is used by Romberg et al. [59], but in contrast in this approach visual features are hashed before bundling. Then bundling is done by combining a central visual features with all visual features in its local neighbourhood. The resulting feature is a tuple consisting of the hashed central visual feature and a hash value calculated from the hash values of all bundled visual features.

Clustering visual features to visual words is time consuming. To counter this drawback

of the BoW approach Ling et al. [41] apply a hashing function to every visual feature of an image. The result is a 32 bit vector which acts directly as visual word. This means that not only the clustering stage is skipped but also the vector quantization from visual feature to visual word is replaced with a faster step, thus providing a great speed advantage.

Chum and Matas [14] take yet another approach as they apply a min-Hash directly to bags of visual words. The main idea is explained in greater detail in [15]. An image can be represented either by a standard bag of visual words or a reduced version where only the presence of visual words $X_w \in \mathscr{V}$ is stored but not its frequency. For simplicity the approach is described only for the reduced version. An image is represented by a set $\mathscr{A}_i$ containing visual words $X_w \in \mathscr{A}_i$. The similarity measure between two images is defined as

$$sim_s(\mathscr{A}_1, \mathscr{A}_2) = \frac{|\mathscr{A}_1 \cap \mathscr{A}_2|}{|\mathscr{A}_1 \cup \mathscr{A}_2|}. \tag{2.18}$$

Now N independent random hashing functions $f_j : \mathscr{V} \to \mathbb{R}$ are defined which assign a real number to each visual word. This leads to the min-Hash function for an image representation under hashing function f_j

$$m(\mathscr{A}_i, f_j) = \underset{X \in \mathscr{A}_i}{\arg\min}\, f_j(X) \tag{2.19}$$

that returns the visual word X of set $\mathscr{A}_i$ with lowest hash value. It can be shown that the probability of $P(m(\mathscr{A}_1, f_j) = m(\mathscr{A}_2, f_j))$ matches $sim_s(\mathscr{A}_1, \mathscr{A}_2)$. For retrieval a set of hash functions $F = (f_1, f_2, \ldots, f_n)$ is applied to an image representation $\mathscr{A}_1$ to get a sketch

$$S_F(\mathscr{A}_1) = (m(\mathscr{A}_1, f_1), m(\mathscr{A}_1, f_2), \ldots, m(\mathscr{A}_1, f_n)) \tag{2.20}$$

where n is the sketch size. As hash functions are independent the probability that two image sketches are equal is $sim_s(\mathscr{A}_1, \mathscr{A}_2)^n$. The probability, that two images have at least h equal sketches out of k, $P(\mathscr{A}_1 \overset{h}{\sim} \mathscr{A}_2)$ is given by

$$P(\mathscr{A}_1 \overset{h}{\sim} \mathscr{A}_2) = \sum_{i=h}^{k} \binom{k}{i} p^{in}(1 - p^n)^{k-i},\ p = sim_s(\mathscr{A}_1, \mathscr{A}_2). \tag{2.21}$$

By requiring two images to have at least h identical sketches a minimal probability for near-duplicate image detection is specified and only images satisfying this criterion are considered for an exact calculation of $sim_s(\mathscr{A}_1, \mathscr{A}_2)$. Concrete values suggested by Chum et al. [15], are:

- Count of visual words $|\mathscr{V}| = 64k$
- Sketch size $n = 3$
- Number of sketches used for classification $k = 64$
- Count of hash functions $N = nk = 192$

- Required equal sketches $h = 1$

One image is therefore represented by $k = 64$ tuples of size $n = 3$ in contrast to a non-sparse bag which would have a size of $|\mathscr{V}| = 64k$. Because the initial calculation of all sketches would still be demanding for large image databases, Chum and Matas [14] propose a method where the database is populated gradually. Sketches are allowed to have "NDef" values in them which are evaluated on demand. E.g. two sketches $\mathscr{A}_1 = (1, \mathit{NDef})$ and $\mathscr{A}_2 = (1, \mathit{NDef})$, with sketch size $n = 2$, are a possible match. Therefore both missing hash values are calculated and the database updated accordingly. Additionally a database structure with inverted sketch-lists to further increase search efficiency is suggested.

The approach proposed by Tong et al. [68] is not a BoW approach strictly speaking. It is discussed here as the idea behind the representation of images is very similar. For each image $\mathcal{I}$ local visual features are calculated and it is assumed that these features $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ are random samples from a particular probability distribution $p(\boldsymbol{x}|\mathcal{I})$. Henceforth images are represented by this probability distribution. The connection to the BoW approach is obvious as a bag of visual words can be interpreted as an approximation to a visual feature probability density function. A significant difference is that the probability function for this method is based on the features of the image while BoW can be seen as based on a global probability function. The main contributions of [68] is an algorithm to efficiently estimate the probability distribution function for an image from its features and an accompanying compact description. The description consists of a mixture model of kernel functions $\kappa(\boldsymbol{x}, \boldsymbol{c})$

$$I(z) = \begin{cases} 1 \text{ if true} \\ 0 \text{ otherwise} \end{cases} \tag{2.22}$$

$$\kappa(\boldsymbol{x}, \boldsymbol{c}) \propto I(|\boldsymbol{x} - \boldsymbol{c}|_2 \leq \rho),\ \rho > 0 \tag{2.23}$$

$$p(\boldsymbol{x}|\mathcal{I}) = \sum_{i=1}^{N} \alpha_i \kappa(\boldsymbol{x}, \boldsymbol{c}_i) \tag{2.24}$$

where $\boldsymbol{c}_i$ is the center of kernel i. They show that a value of $N = 1,000$ results in sufficient representations for their experiments which is a significant reduction of dimensionality when compared to BoW approaches with a typical number of visual words above 10,000 [18]. To compare a query image against a database image its local features are extracted and interpreted as a sample of the database image's visual feature probability distribution. Then the likelihood of this sample is evaluated which results in the probability that the query image is a near-duplicate of the database image. The log likelihood of query image $\mathcal{Q}$ with features $\{\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_m\}$ in respect to a particular database image $\mathcal{I}_i$ is defined as

$$log\ p(\mathcal{Q}|\mathcal{I}_i) = \sum_{k=1}^{m} \left(\sum_{j=1}^{N} \alpha_j^i \kappa(\boldsymbol{q}_k, \boldsymbol{c}_j) \right) \tag{2.25}$$

where $\boldsymbol{\alpha}_j^i = \{\alpha_1^i, \alpha_2^i, \ldots, \alpha_N^i\}$ and $\boldsymbol{c}_j$ are the estimated values for image $\mathcal{I}_i$. To prevent a full naive scan of the database for every query an approximate similarity measure is given as well and equation 2.25 is only applied to a restricted candidate list. They compare their approach against BoW with three different setups for generation of the visual words and show that their approach needs significantly less pre-processing (training) time. Retrieval time however is higher (below factor 2). Search accuracy is about the same as BoW with visual words constructed by approximate kMeans [54].

2.3.3 Fisher Vectors

In [33], Jégou et al. propose to represent images as Fisher Vectors (FV). It follows the same idea as the approach in [68], by assuming that the local visual features $\{x_t, t = 1 \ldots T\}$ of an image $\mathcal{I}$ are samples from a random variable X and follow a particular distribution. Instead of an image specific distribution a global image-independent distribution is assumed. As representation a Gaussian Mixture Model (GMM) is used. λ contains all parameters describing the GMM. Now the image $\mathcal{I}$ can be represented by the vector obtained by

$$G_\lambda(\mathcal{I}) = \frac{1}{T} \sum_{t=1}^{T} \nabla_\lambda \, log \, u_\lambda(x_t) \tag{2.26}$$

$$= \frac{1}{T} \sum_{t=1}^{T} \nabla_\lambda \, log \, \left(\sum_{j=1}^{N} w_j \mathcal{N}(x_t | \mu_j, \Sigma_j) \right). \tag{2.27}$$

G_λ is the gradient of the log-likelihood function in respect to parameter vector λ. The resulting vector has therefore the same dimension as the number of Gaussian components. It points in the direction in which the model λ has to be moved to better fit image $\mathcal{I}$. One of its properties is that it transforms a variable number of feature vectors into a single fixed length vector describing the image. To compare two image $\mathcal{I}, \mathcal{J}$ the Fisher kernel is used

$$K(\mathcal{I}, \mathcal{J}) = G_\lambda(\mathcal{I})^T F_\lambda^{-1} G_\lambda(\mathcal{J}) \tag{2.28}$$

$$F_\lambda = E_{X \sim u_\lambda} \left[\left(\nabla_\lambda \, log \, u_\lambda(X) \right) \left(\nabla_\lambda \, log \, u_\lambda(X) \right)^T \right] \tag{2.29}$$

where F_λ is the Fisher information matrix which acts as whitening transform. In [53] it is shown that vectors resulting from G_λ can be normalized beforehand and the Fisher kernel can be rewritten as dot product

$$K(\mathcal{I}, \mathcal{J}) = \mathcal{G}_\lambda(\mathcal{I})^T \mathcal{G}_\lambda(\mathcal{J}). \tag{2.30}$$

where $\mathcal{G}_\lambda$ incorporates said transform. The vector $\mathcal{G}_\lambda(\mathcal{I})$ is called the Fisher vector (FV) of image $\mathcal{I}$. The dimension of such a Fisher vector is $2Nd + N$ where d is the dimensionality of the visual features and N the number of Gaussian components. It is however sufficient to consider the gradient only in respect to the mean of the Gaussian components [33] and

thus reducing the Fisher vector dimension to Nd. With this reduction Fisher vectors are approximated [53] by

$$\mathcal{G}_\lambda(\mathcal{I}) = \frac{1}{T}\sum_{i=1}^{N}\left(\frac{1}{\sqrt{w_i}}\sum_{t=1}^{T}\frac{w_i\,\mathcal{N}(x_t|\mu_i,\Sigma_i)}{\sum_{j=1}^{N} w_j\,\mathcal{N}(x_t|\mu_j,\Sigma_j)}\frac{x_t-\mu_i}{\sigma_i}\right). \tag{2.31}$$

The Gaussian components can be seen as the visual words of the FV approach. When compared to a BoW setup with m visual words, only m/d visual words are needed to get image descriptions of same dimensionality. Equation 2.31 further shows that for each local visual feature its relations to all visual words are encoded into the vector. Therefore FV provide a richer description than BoW. Jégou et al. [33] compare their FV setup against a standard BoW setup and show that FV can achieve competitive results while reducing image description dimensionality significantly.

2.3.4 Histogram Distance

The bags of visual words method as well as the fisher vector method produces histograms describing an image by the frequencies of its visual words. To calculate the distance or similarity of two images different measures can be applied. For two histograms $\mathscr{A} = (a_1, a_2, \ldots, a_n)$ and $\mathscr{B} = (b_1, b_2, \ldots, b_n)$ the manhattan distance

$$d(\mathscr{A},\mathscr{B}) = \sum_{i=1}^{n}|a_i - b_i| \tag{2.32}$$

as well as the euclidean distance

$$d(\mathscr{A},\mathscr{B}) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2} \tag{2.33}$$

can be used [51]. If the histograms are normalized with the L_2 norm ($\|\mathscr{A}\| = \|\mathscr{B}\| = 1$) the dot product results in a similarity measure with values in $[0, 1]$

$$d(\mathscr{A},\mathscr{B}) = \mathscr{A}\cdot\mathscr{B} = \sum_{i=1}^{n} a_i b_i \tag{2.34}$$

This is used to rank images in [33]. Another distance for histograms is the χ^2-distance as described by Xu et al. [76]

$$d(\mathscr{A},\mathscr{B}) = \frac{1}{2}\sum_{i=1}^{n}\frac{(a_i - b_i)^2}{a_i + b_i}. \tag{2.35}$$

The χ^2-distance is derived from the identically named statistical test. It does not only take the difference of two bins into account but also their accumulated sizes and thus adjusts the

influence on the overall distance. E.g. the difference of two bins weighs stronger than the same difference for two bins with higher values. An even more advanced distance measure is the earth mover's distance (EMD), with which images are compared in [63]:

> "Intuitively, given two distributions, one can be seen as a mass of earth properly spread in space, the other as a collection of holes in that same space. We can always assume that there is at least as much earth as needed to fill all the holes to capacity by switching what we call earth and what we call holes if necessary. Then, the EMD measures the least amount of work needed to fill the holes with earth".

Formally the overall costs can be written as

$$d(\mathscr{A}, \mathscr{B}) = \sum_{a_i} \sum_{b_j} c_{ij} f_{ij} \tag{2.36}$$

where c_{ij} is the cost of moving a "unit of supply" from bin i to bin j and f_{ij} specifies the flow so that the distance or "work" is minimal under the following constraints:

$$f_{ij} \geq 0,\ a_i \in \mathscr{A}, b_i \in \mathscr{B} \tag{2.37}$$

$$\sum_{a_i} f_{ij} = b_j,\ b_j \in \mathscr{B} \tag{2.38}$$

$$\sum_{b_j} f_{ij} \leq a_i,\ a_i \in \mathscr{A} \tag{2.39}$$

The three constraints can be explained as:

- Only positive flows are possible.
- All holes have to be filled completely.
- The flow of one bin can not exceed its value.

Pele and Werman [50] provide an efficient implementation[3] and apply it to the problem of SIFT feature matching [49].

[3] `http://www.seas.upenn.edu/~ofirpele/FastEMD/code`, accessed August 21, 2013

CHAPTER 3

Methodology

This chapter describes the methodology used in this thesis. Different methods for comparison of near-duplicate image retrieval systems are considered. The specific image sets used for evaluation are discussed. And finally tools, programming languages and statistics used in this thesis are described.

3.1 Image Sets and Transformations

Beside efficiency, search accuracy is one of the most important properties of near-duplicate image retrieval systems. A chosen image set influences the efficiency through the average number of features per image. However by purposefully setting the specific parameters of the components of the near-duplicate image retrieval system the average number of features per image can be controlled. On the other hand one image set can be more challenging than another because of its specific visual content and its accompanying transformations. This influences the search accuracy and therefore a near-duplicate image retrieval system has to be evaluated with image sets and transformations that reflect their specific, intended tasks.

3.1.1 Inria Copydays & Flickr1M

The Inria Copydays image set[1] is especially targeted at copy detection (see figure 3.1 for examples). It consists of 157 images which are transformed into query images. The applied transformations mimic copying of the image under various circumstance, hence copy detection. The transformations match those covered by near-duplicate image retrieval and the image set is used in various state of the art papers, e.g. [11, 33, 75]. The images are high quality with varying motifs and have an average resolution of 4 mega pixel, with a standard deviation of 1.4 mega pixel. Transformed versions of the original are also provided. The image transformations are:

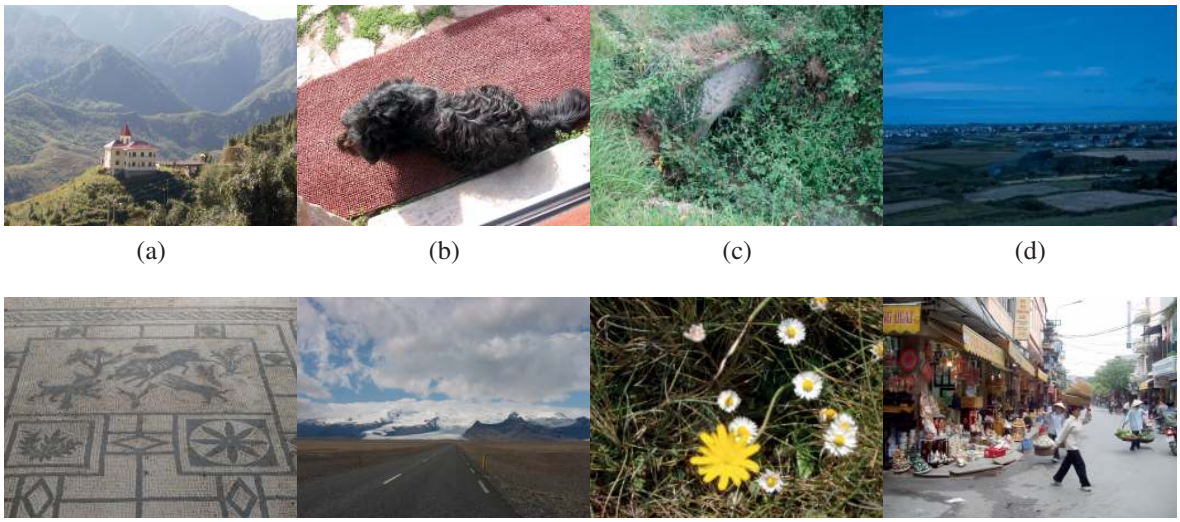
Cropping 10% – 80% of the image surface is removed.

[1] `http://lear.inrialpes.fr/~jegou/data.php#holidays`, accessed August 21, 2013

Scaling & Compressing Image is scaled to $1/16$ pixels and different levels of jpeg compression are applied (75, 50, 30, 20, 15, 10, 8, 5, 3).

Strongly Attacked Images under heavy transformation, e.g. print and scan, blur, paint, etc.

Figure 3.2 shows examples for jpeg compressed and strong attacked images. To increase the image database size to meaningful levels, an additional image set called Flickr1M is provided with the Inria Copydays image set. It consists of 1,000,000 images randomly downloaded from Flickr. A random subset of 5,000 images, used in section 4.4, shows an average resolution of 2.5 mega pixel, with a standard deviation of 3 mega pixel. The images are added to the Copydays images as distractor or background images.



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

(e) (f) (g) (h)

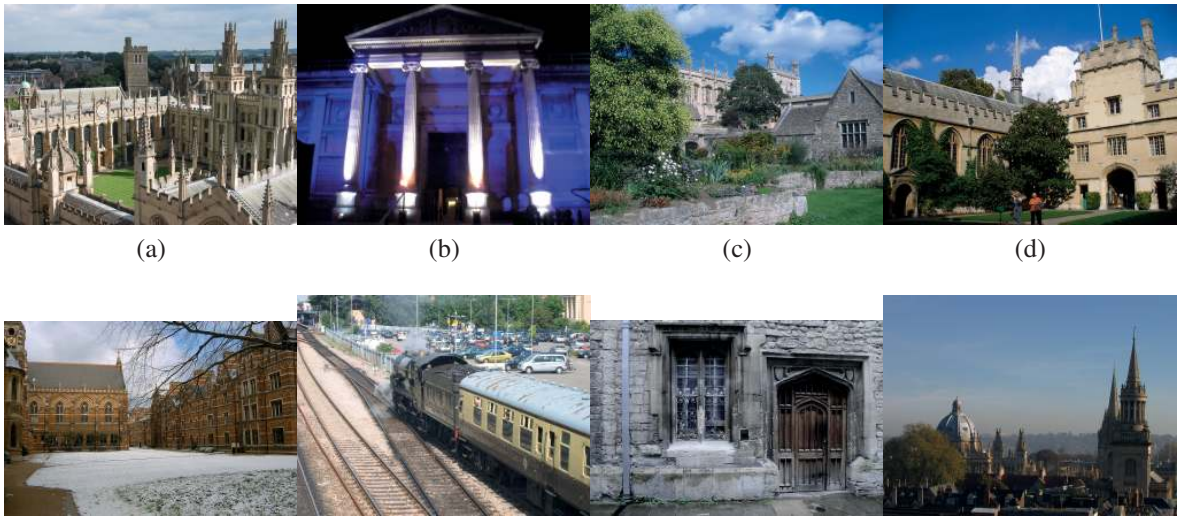
Figure 3.1: Eight random images from the Inria Copydays image set.

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

Figure 3.2: (a) and (c) show strong attacked and (b) and (d) jpg compressed images from the Inria Copydays image set. Corresponding originals are shown in figure 3.1.

3.1.2 The Oxford Buildings Dataset

The Oxford Buildings Dataset[2] consists of 5,062 images collected by searching Flickr[3] for images showing particular Oxford landmarks. The main motifs are buildings but the image set is not restricted to those (see figure 3.3 for examples). The images have an average resolution of 0.8 mega pixel, with a standard deviation of 0.06 mega pixel. While the image set is not intended for near-duplicate image retrieval it is used in related works, e.g. [9, 68, 72]. Consequently no transformed images are provided or image transformations suggested. Instead a set of query images for specific landmarks with meta-information is provided. The meta-information contains regions of interest which are intended to be used in a training stage. Figure 3.4 shows some of the query images combined with their corresponding regions of interest. The query images are used for an image retrieval challenge where all images of corresponding Oxford landmarks should be retrieved.



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

(e) (f) (g) (h)

Figure 3.3: Eight random images from the The Oxford Buildings Dataset.

Figure 3.4: Query images from the Oxford Buildings Dataset with regions of interest.

[2] `http://www.robots.ox.ac.uk/~vgg/data/oxbuildings`, accessed August 21, 2013
[3] `http://www.flickr.com`, accessed August 21, 2013

3.1.3 APA-IT Press Images

The APA-IT[4] image set consists of approximately 1,000,000 press images. 10,000 images, selected randomly from the whole image set, have an average resolution of 6.5 mega pixel, with a standard deviation of 2.8 mega pixel. The image set represents a specific use-case discussed in this thesis. Press images are sold to newspapers, magazines, etc. and in turn often published online. Copying these images and publishing them without proper licensing is simple. The images can be transformed beforehand to match specific visual preferences or disguise copyright infringements. Identifying illegal copies is therefore not trivial and a typical problem of near-duplicate image retrieval.

3.1.4 Ancient Coins

The automatic classification of ancient coins is a common problem in the scientific field of numismatic[5]. An image of a coin belonging to a specific type or class can be seen as original image and an image of a coin belonging to the same type or class as near-duplicate. Thus the problem of classification of ancient coins can be interpreted as near-duplicate image retrieval problem.

Figure 3.5 shows images of three ancient coins belonging to the same class. The images belong to an image set[6] that features 180 images with 60 class (three images per class). To set

(a) (b) (c)

Figure 3.5: Three coins of the same class from [78].

up a near-duplicate image retrieval challenge each image can be used as query image while the remaining images form the database of originals. The correct retrieval result would be the other two coin images which share the same class.

[4] `http://www.apa-it.at`, accessed August 21, 2013

[5] `http://www.caa.tuwien.ac.at/cvl/research/ilac`, accessed August 21, 2013

[6] `http://www.caa.tuwien.ac.at/cvl/research/ilac/coindata`, accessed August 21, 2013

3.2 Performance Metrics for Clustering

Clustering results in the partition of a data-set in groups. Evaluating such a partition requires a measure of quality.

3.2.1 Within Cluster Sum of Squares

The kMeans algorithm [42] aims to minimize the within cluster sum of squares (WCSS). The WCSS is defined as follows. For a set of objects $(\boldsymbol{o_1}, \boldsymbol{o_2}, \ldots, \boldsymbol{o_n})$ with centroids $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)$ and defined norm $\|\cdot\|$ the goal is to minimize the WCSS

$$\arg\max_{C} \sum_{i=1}^{k} \sum_{\boldsymbol{o}_j \in G_i} \|\boldsymbol{c}_i - \boldsymbol{o}_j\|^2 \tag{3.1}$$

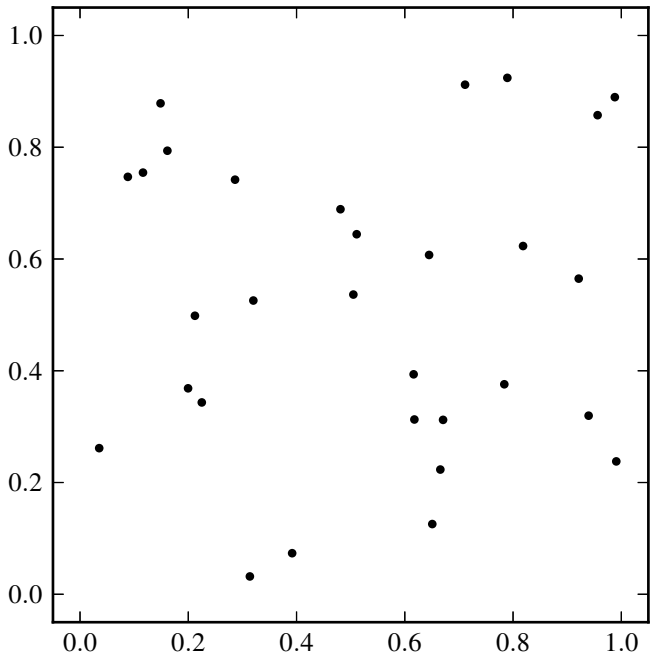
where G_i is the group corresponding to centroid $\boldsymbol{c}_i$. Figure 3.6 shows a visualization of the partitioning of data points by cluster centers and illustrates the accompanying WCSS. The WCSS can be used to measure the quality of centroid based algorithms. A minimal WCSS is achieved by spreading the data points evenly over the cluster centers. However this is not always the intended goal of clustering [25]. Another drawback of the WCSS is that it can not be used to compare partitions with different numbers of clusters. Therefore it can not be used to determine if a higher or lower number of clusters would represent the data better.

3.2.2 Dunn Index

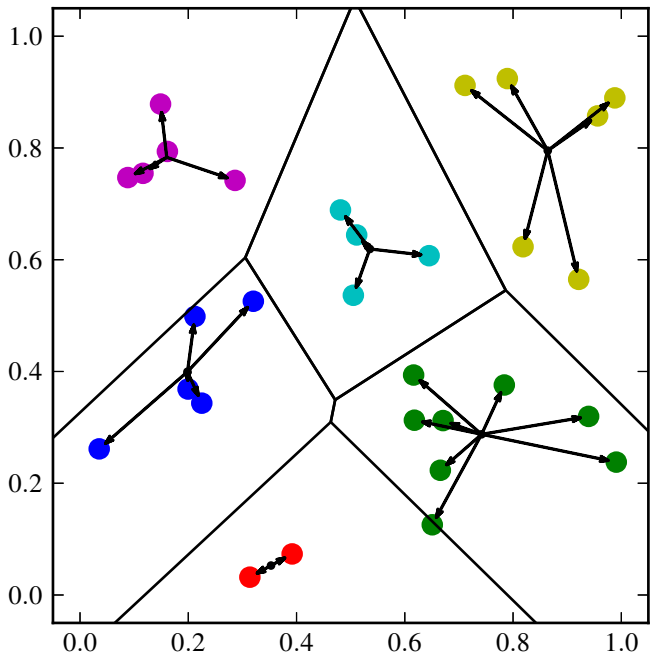
Another clustering algorithm performance measure is the Dunn Index [24]:

$$D = \min_{i=1,\ldots,N} \left(\min_{j=i+1,\ldots,N} \left(\frac{d(c_i, c_j)}{\max_{k=1,\ldots,N} d'(c_k)} \right) \right) \tag{3.2}$$

$d(c_i, c_j)$ measures the distance between cluster i and j. It is a measure for the inter cluster distance of the partition. $d'(c_k)$ represents the intra cluster distance. In other words the distance of the two nearest clusters is normalized by the intra distance of the "largest" cluster. The goal is to find a partition with high inter cluster distance and low intra cluster distance. Thus a clustering algorithm should produce partitions with a high Dunn index. In contrast to the WCSS partitions of different size can be compared, as the Dunn index takes the number of cluster into consideration. This is done indirectly, as the size of the largest cluster increases with decreasing number of clusters.



(a) Uniformly distributed 2d random data points.



(b) Partition of data points by cluster centers.

Figure 3.6: Voronoi diagram showing the partition of data points by cluster centers. The sum of the quadratic lengths of all arrows corresponds to the within cluster sum of squares (WCSS).

3.2.3 Davies–Bouldin Index

Davies and Bouldin [19] suggest the Davie–Bouldin index (DB) to quantify the quality of a data partition. It is defined as follows:

$$DB = \frac{1}{N} \sum_{i=1}^{N} \max_{j:i \neq j} \frac{S_i + S_j}{M_{ij}} \tag{3.3}$$

S_i, S_j stand for the dispersions of clusters i, j and M_{ij} represents the distance between cluster i and j. The DB measure should be minimized. Simplified this means that compact clusters can be near to each other, while widespread clusters should be as far from each other as possible. However only the maximum values are taken into consideration. As the number of cluster centers is taken into account, the Davie–Bouldin index can be used to compare partitions with different number of clusters.

3.3 Performance Metrics for NDIR

There exist different metrics for evaluating the performance of an image retrieval system. The most important are described in the following.

3.3.1 Precision and Recall

Precision and recall provide a measure for the performance of a single query only. To describe a retrieval system a specific number of queries is executed. The precision and recall for every query is calculated and plotted in a precision-recall graph with the precision on the vertical axis and the recall on the horizontal axis. Wang et al. [72] use precision-recall graphs in the evaluation of their retrieval system based on the Oxford Buildings Dataset. One drawback of precision and recall is, that they do not consider the order or rank of the retrieved documents.

3.3.2 Receiver Operating Characteristic

Receiver Operating Characteristic (ROC) curves are related to precision-recall curves as described in detail by Davis and Goadrich [20]. The principle is the same as for precision-recall curves, but instead of precision and recall the false positive and true positive rate are calculated for every query. To generate the curve the false positive rate is plotted on the vertical axis and the true positive rate on the horizontal axis. Zheng et al. [81] use ROC curves and the Inria Copydays image set in their work.

3.3.3 Cumulative Matching Characteristics

Cumulative Matching Characteristics (CMC) are described in [47]. A set of images $\mathcal{P}$ is used as queries for a database with images $\mathcal{Q}$. For each query $p_i \in \mathcal{P}$ a sorted list of database images $q_j \in \mathcal{Q}$ is calculated, with the best matching images on top. The value R_n determines the correct images in the top n of each list. The CMC curve is now generated by plotting the cumulative match score $\frac{R_n}{\|\mathcal{P}\|}$ on the vertical axis and n on the horizontal axis. Tong et al. [68] use CMC curves in their work and the relationship between CMC and ROC curves is discussed in [6].

3.3.4 Mean Average Precision

A suitable measure, which is commonly used in information retrieval, is the Mean Average Precision (maP) [71]. It provides a measure of the overall performance of an information retrieval system and incorporates precision as well as recall. It is calculated over a fixed number of queries where each query consists of documents ranked by their relevance. The average precision is defined as

$$AveP = \frac{\sum_{k=1}^{n}(P(k) \times rel(k))}{|\{relevant\ documents\}|} \tag{3.4}$$

where n is number of retrieved documents. $P(k)$ is the precision when assuming that only the first k documents are received. $rel(k)$ is 1 if the k-th document is relevant and 0 otherwise. $|\{relevant\ documents\}|$ is the number of relevant documents in the whole database and not in the query and thus adds information regarding recall. In contrast to plain precision and recall the rank of each relevant document is considered through $P(k)$. The mean average precision for a given number of queries Q is defined as

$$maP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q}. \tag{3.5}$$

For the experiments in section 4.4 only one relevant image is in the database for each query. Thus the average precision becomes

$$AveP = \begin{cases} \frac{1}{k} \text{ if the k}^{th} \text{ image is relevant} \\ 0 \text{ otherwise} \end{cases}. \tag{3.6}$$

If the relevant image is not among the retrieved image its rank can be seen as $k = +\infty$ and thus $AveP = \frac{1}{+\infty} = 0$. This leads to a mean average precision of

$$maP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} = \frac{\sum_{q=1}^{Q} \frac{1}{k_q}}{Q} = H(k_1, k_2, \ldots, k_q)^{-1}, \tag{3.7}$$

where $H(k_1, k_2, \ldots, k_q)$ is the harmonic mean of the ranks of the retrieved images. The harmonic mean is typically used to calculate the average of rates. In case of image retrieval it provides a sensible way of building the average of the image ranks. E.g. for three queries with image ranks 1, 2 and 1,000 the mean rank is $\frac{1+2+1000}{3} \approx 334$, which is rather high. Practically however one could say that rank 1 and 2 are "successful" queries, while a rank of 1,000 means the image was not found. The harmonic mean is $\frac{3}{\frac{1}{1}+\frac{1}{2}+\frac{1}{1000}} \approx 2$ and the $maP \approx 50\%$. Consequently this means that the maP is not distorted by high ranks. It has little influence if a rank is above 100 ($\frac{3}{\frac{1}{1}+\frac{1}{2}+\frac{1}{100}} \approx 2$) or above 1,000 as both values are mapped to a precision value smaller than 1%. Similar both image ranks can be interpreted simply as image not found.

3.4 Statistics

To evaluate and visualize the results of experiments where algorithms with randomized components are present specific statistical methods are considered.

3.4.1 Student's t-Test

The variant of the Student's t-Test [82] for independent two-sample experiments, e.g. two series of results with different parameters or different algorithms, is of particular interest:

$$t = \frac{\overline{X}_1 - \overline{X}_2}{S_{\overline{X}_1\overline{X}_2} \cdot \sqrt{\frac{2}{n}}} \tag{3.8}$$

$$S_{\overline{X}_1\overline{X}_2} = \sqrt{\frac{1}{2}\left(S_{X_1}^2 + S_{X_2}^2\right)} \tag{3.9}$$

Where S_{X_1} and S_{X_2} are the variances of sample 1 and 2. The degrees of freedom are calculates as $2n - 2$. Requirements for the application of the Student's t-Test are that:

1. The sample sizes are equal.
2. The two distributions of the test series have the same variance.

If these requirements are fulfilled the t value together with the degrees of freedom can be used determine a probability, that two samples have a significantly different mean value. If the samples represent the results of two different algorithms a decision can be made which algorithm performs better, if the test yields a confidence high enough.

3.4.2 Welchs's t-Test

In contrast to the Student's t-test, the Welch's t-test [74] is intended for use with two samples with different variance and defined as:

$$t = \frac{\overline{X}_1 - \overline{X}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}} \tag{3.10}$$

$$\nu = \frac{\left(\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}\right)^2}{\frac{s_1^4}{N_1^2(N_1-1)} + \frac{s_2^4}{N_2^2(N_2-1)}} \tag{3.11}$$

Where $\overline{X}_i$ is the mean, s_i^2 the variance and N_i the size of sample i. ν represents the degrees of freedom. t and ν are used in a two tailed test to estimate if the mean values of the two samples are significantly different. The null hypothesis is:

$$H_0 : \overline{X}_1 = \overline{X}_2 \tag{3.12}$$

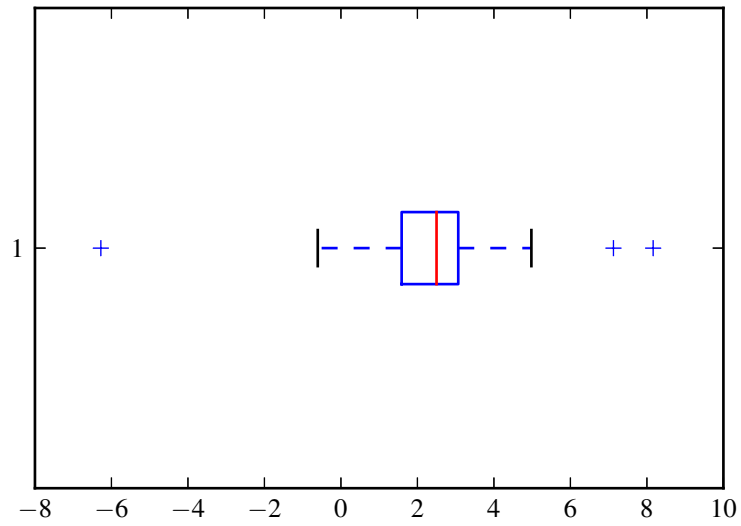
The result is a probability value p that the sample means are equal (H_0 is true). Based on a significant difference, e.g. p is smaller than 1%, one can assume that an algorithm or parameter set is better (or worse) than another.

3.4.3 Box Plot

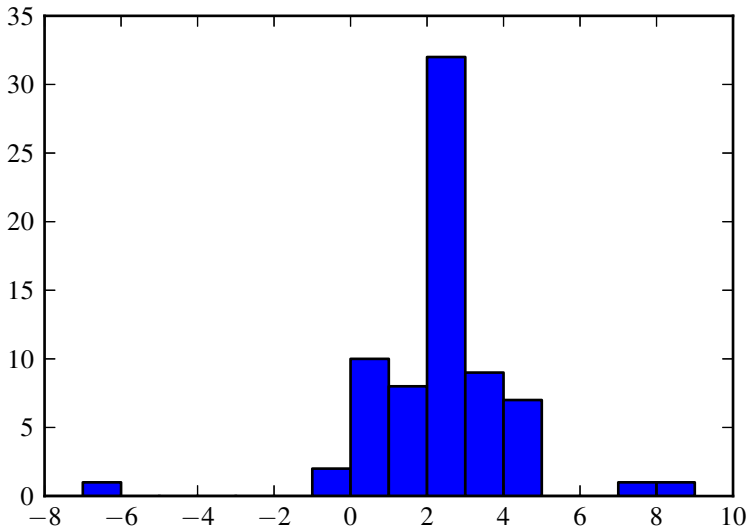
To visualize the results of experiments as described above, box plots are used. An example with corresponding histogram can be seen in figure 3.7. The histogram shows the number of occurrences of samples belonging into the respective histogram bin, where each bin represents a numerical interval. The histogram provides a visual representation of the distribution of the samples over these numerical intervals. The box plot provides an alternative and more compact visualization of this distribution. The vertical line in the middle of the box specifies the mean of all samples (second quartile). From this line the box extends left to the first and right to the third quartile. The meaning of the whiskers follows the definition of McGill et al. [44]. The whiskers of each box extend at most 1.5 inter-quartile range and the additional crosses represent samples lying outside of this range.

3.5 Data Storage

One important component of every content-based image retrieval and thus near-duplicate image retrieval system is the database. A large amount of images has to be stored, as well as the corresponding image meta data. Image meta data includes the image representation used for retrieval. For efficiency image representations should be as small as possible in terms of data storage size. The read to write ratio can be assumed as high: An image is written into the database once but queried often. The exact value depends on the specific use case.



(a) Box plot



(b) Histogram

Figure 3.7: Illustration of an exemplary box plot and corresponding histogram.

Furthermore near-duplicate image retrieval systems are required to handle parallel requests. The requirements for a data storage are therefore to efficiently handle the parallel access of small data records. In this thesis three basic types of data stores are considered:

- Relational Database
- File-system
- NoSQL Data Store

The model for relational databases is formulated by Codd [16]. Data is organized in tables and data records are identified through unique ids. Complex queries can be formulated to access and combine data records across different tables. However this strength of relational

databases is not needed for near-duplicate image retrieval systems but comes with a cost in terms of run-time [12].

Using a file-system for storing images and image related data is straight forward. Filenames act as unique keys and the data is the file's content. This approach would limit file-system access to a single system without further measures, e.g. a clustered file-system like the Network File System (NFS)[7]. Additionally without defining a custom data format every component of a data record would have to be stored in a separate file. This would result in overhead in terms of storage space.

The third option are NoSQL Data Stores [12] which have become popular in recent years. Data is represented in flat structures. Typical models are tuples, documents, extensible records or objects. According to [12] key-features of NoSQL systems are:

> "
>
> 1. the ability to horizontally scale "simple operation" throughput over many servers,
> 2. the ability to replicate and to distribute (partition) data over many servers,
> 3. a simple call level interface or protocol (in contrast to a SQL binding),
> 4. a weaker concurrency model than the ACID transactions of most relational (SQL) database systems,
> 5. efficient use of distributed indexes and RAM for data storage, and
> 6. the ability to dynamically add new attributes to data records.
>
> "

These key-features are essential for large scale near-duplicate image retrieval systems and therefore NoSQL is the system of choice in this thesis. The concrete implementation chosen is MongoDB[8]. This choice reflects a personal preference of the author for MongoDB because of its application programming interface. MongoDB is a document orientated storage where documents are stored in collections. Collections in turn are stored in databases. A document is stored as Binary JSON[9]. Efficient queries can be issued per collection. Therefore indices for specific document fields can be generated. Among the high-level features of MongoDB are horizontal scaling (sharding) and automatic replication.

3.6 Programming Languages and Libraries

A near-duplicate image retrieval system requires high efficiency. Furthermore to fulfil the requirements in terms of parallel requests and scalability it is typically implemented as distributed system.

[7] `http://tools.ietf.org/html/rfc3530`, accessed August 21, 2013
[8] `http://www.mongodb.org/`, accessed August 21, 2013
[9] `http://bsonspec.org/`, accessed August 21, 2013

To reflect this the C++ programming language[10] (GNU implementation[11]) is used for components responsible for computational tasks. C++ is a compiled, statically typed, "multi-paradigm" language [67] and is used because of its high run-time efficiency and its library ecosystem. Most important the Open Source Computer Vision (OpenCV)[12] library which provides implementation for all visual features described in section 2.1. To allow for vertical scaling the Open Multi-Processing (OpenMP)[13] library, which can be seen as an extension of C++, is used. Hardware resources provided by graphical processing units (GPU) can be utilized by the proposed near-duplicate image retrieval system. To achieve this the Open Computing Language (OpenCL)[14] is used. More specifically the implementations provided by NVIDIA[15] and AMD[16] which can be interfaced directly from C++.

For implementation of the distributed system components the Python programming language (CPython implementation)[17] is used. Python is a high level language featuring dynamic typing and automatic memory management. Its extensive standard library allows rapid development. For horizontal scaling especially the "multiprocessing" module is used. It provides a straight forward implementation of the Queue pattern for sharing workload across computation nodes in a TCP/IP network. The Numpy[18] module is used for non-time critical scientific calculations and evaluations. The matplotlib[19] module is used for scientific plotting in this thesis.

[10] `http://www.open-std.org/JTC1/SC22/WG21/`, accessed August 21, 2013
[11] `http://gcc.gnu.org/projects/cxx0x.html`, accessed August 21, 2013
[12] `http://opencv.willowgarage.com`, accessed August 21, 2013
[13] `http://openmp.org`, accessed August 21, 2013
[14] `http://www.khronos.org/opencl/`, accessed August 21, 2013
[15] `https://developer.nvidia.com/opencl`, accessed August 21, 2013
[16] `http://developer.amd.com/resources/heterogeneous-computing/opencl-zone`, accessed August 21, 2013
[17] `http://www.python.org/`, accessed August 21, 2013
[18] `http://www.numpy.org`, accessed August 21, 2013
[19] `http://matplotlib.org`, accessed August 21, 2013

CHAPTER 4

Results

In the following chapter the developed implementations and practical results are explained, discussed and analyzed. **First** the topic of large scale clustering of binary local visual features is discussed. Therefore a fixed set of binary features is clustered with different algorithms and varying parameters. **Second** nearest neighbour search in high dimensional binary feature space is addressed briefly. **Finally** the implemented bags of visual words approach is explained and evaluated. The mean average precision is calculated for queries over specific image sets. Comparisons are drawn to state of the art methods in terms of search accuracy and overall efficiency.

4.1 Visual Words

Clustering is an important and time consuming stage of the bags of visual words [18] method. In the following two clustering algorithms are adapted for binary local visual features and compared against each other. As quality measure the within cluster sum of squares is used. This decision is based on the nature of the two compared algorithms. Additionally the number of centers is selected manually and does not need to be reflected in the quality measure.

4.1.1 kShifts

kShifts is a fast and efficient clustering algorithm introduced in [55, 56] and is suitable for clustering large data-sets (e.g. 45 GB of 960 dimensional spatio temporal descriptors to 10,000 clusters in 20 hours using two X5560@2.8GHz processors with 4 cores each [66]).

The input is a set of given data points $S = (\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_n)$ where $\boldsymbol{s}_i \in \mathbb{R}^h$. Out of S a multiset P is generated that contains l random permutations of S, thus $P = (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m)$ and $m = n \cdot l$. This multiset is then processed in linear order (algorithm 1). The results are cluster centers $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)$ where $\boldsymbol{c}_i \in \mathbb{R}^h$. The number of cluster centers k is predetermined. Algorithm 1 shows the kShifts algorithm. Three different weighting

Input:
permutation of data points $P = (\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m)$
number of cluster centers k with $k \leq m$
distance metric $d(\boldsymbol{p}_i, \boldsymbol{c}_j)$
Result:
cluster centers $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)$
Function:
$nearestCenterIndex(\boldsymbol{p}_i, C) = \arg\min_j d(\boldsymbol{p}_i, \boldsymbol{c}_j), \boldsymbol{c}_j \in C$
$weightCenter(i, m) \in [0, 1]$
Algorithm:
initialize C
for $i = 1, 2, \ldots, m$ **do**
$j = nearestCenterIndex(\boldsymbol{p}_i, C)$
$\boldsymbol{c}_j = (1.0 - weightCenter(i, m)) * \boldsymbol{p}_i + weightCenter(i, m) * \boldsymbol{c}_j$
end

Algorithm 1: The kShifts algorithm.

functions are evaluated. The original weighting function as defined in [55]

$$f(i, m) = 1 + \frac{a}{i \cdot m} \tag{4.1}$$

$$weightCenter(i, m) = \frac{f(i, m, a) - 1}{f(i, m, a)} \tag{4.2}$$

where a is an additional parameter, a linear weighting function

$$weightCenter(i, m) = \frac{i + \frac{a \cdot m}{1-a}}{m + \frac{a \cdot m}{1-a}} \tag{4.3}$$

where $a \in [0, 1]$ is the starting offset and a quadratic weighting function

$$weightCenter(i, m) = \left(1 - \left(1 - \frac{i}{m}\right)^2\right)(1 - a) + a \tag{4.4}$$

with $a \in [0, 1]$ again the starting offset in percent.

Given the performance and run-time of binary visual features there is no reason that speaks against using such features in a bags of visual words setup. However as these feature vectors are binary the L2 norm $|\cdot|_2$ is not directly applicable. Clustering algorithms exist which work directly with binary strings but are not suitable in this context. Hierarchical single-link clustering can be implemented in $\mathcal{O}(n^2)$ at best [64] which is inefficient for large data-sets. More important link based algorithms are able to form non-globular clusters, e.g. Jarvis and Patrick clustering [31]. Figure 4.1 illustrated the difference between globular and non-globular clusters. Non-globular cluster have a shape that does not allow a description, with acceptable accuracy, by a center point and accompanying maximum distance. Furthermore, in the worst case, non-globular clusters can be entangled. These properties render an

efficient feature to cluster assignment unfeasible.

(a) Globular clusters. (b) Non-globular clusters.

Figure 4.1: Illustration of different clusters from [31].

In the following the kShifts algorithm is adapted to binary feature vectors (see algorithm 2). Finding the nearest center for a data point is done with the L1 norm $|\boldsymbol{p} - \boldsymbol{c}|_1$. The L1 norm equals the Hamming distance for binary feature vectors and can be implemented very efficiently. Furthermore the L1 norm $|\cdot|_1$ matches the L2 norm $|\cdot|_2$ in this case:

$$|\boldsymbol{p} - \boldsymbol{c}|_2 = \sum_{i=1}^{n} (p_i - c_i)^2 = \sum_{i=1}^{n} |p_i - c_i| = |\boldsymbol{p} - \boldsymbol{c}|_1,\ p_i, c_i \in \{0, 1\} \tag{4.5}$$

Updating the centers has to be performed in floating point. Therefore the data point is convert to a floating point vector, e.g. $\boldsymbol{v}_b = (1, 0, 1, 0, 0, 1)$ to $\boldsymbol{v}_f = (1.0, 0.0, 1.0, 0.0, 0.0, 1.0)$. For the centers a binary version and a floating point version is kept. After updating the center the floating point version is converted to binary by rounding each value.

Implementation

The most important aspect of the kShifts clustering algorithm is efficiency. Therefore the implementation aims to be as efficient as possible. The most time consuming part of the algorithm is to find the current nearest center for a data point. The L1 $|\cdot|_1$ measure is used to determine the nearest center. For a data point $\boldsymbol{p}$ and a center $\boldsymbol{c}$, distance $d(\boldsymbol{p}, \boldsymbol{c})$ is defined as

$$d(\boldsymbol{p}, \boldsymbol{c}) = |\boldsymbol{p} - \boldsymbol{c}|_1 \tag{4.6}$$

$$= \sum_{i=1}^{n} |p_i - c_i|,\ p_i \in \boldsymbol{p}, c_i \in \boldsymbol{c}. \tag{4.7}$$

For binary vectors this equals the hamming distance

$$d(\boldsymbol{p}, \boldsymbol{c}) = \sum_{i=1}^{n} xor(p_i, c_i), p_i \in \boldsymbol{p}, c_i \in \boldsymbol{c}. \tag{4.8}$$

On modern central processing units (CPU) with x86 architecture this can be calculated very efficiently using intrinsics[1]. Two intrinsic functions are needed:

_mm_xor_si128 Is part of the Streaming SIMD Extensions 2 (SSE2), where SIMD stands for "single instruction multiple data". The function evaluates the xor value for four pairs of 32 bit integers in one instruction. This equals a binary vector with a size of 128 bit. Thus for ORB [62] or BRIEF-32 [10] feature vectors with a size of 256 bits two instructions would be necessary for the xor calculation.

_mm_popcnt_u64 The sum in the hamming distance (equation 4.8) equals counting the bits with value one after the xor operation. This operation is called "popcount". The equation becomes

$$d(\boldsymbol{p}, \boldsymbol{c}) = popcount(xor(p, c)) \tag{4.9}$$

where $xor(\cdot)$ is a bit-wise operation. On modern CPUs a dedicated instruction is available. *_mm_popcnt_u64* calculates the popcount for a 64 bit integer. For the example above this equals four popcount calls plus calculating the sum of all four popcount values.

In the binary adapted version of kShifts, algorithm 2, a conversion from binary to floating point vector and vice versa is performed. To efficiently convert from binary to floating point vector the binary vector is split into groups of four bit. Each four bit group is interpreted as a number between 0 and $15 = 2^4 - 1$. Every number represents a case in a *switch*-statement. In each case a vector of four floats has to be written to memory

$$\boldsymbol{v} = (v_1, v_2, v_3, v_4),\ v_i \in \{0.0, 1.0\}. \tag{4.10}$$

The reason for the group size of four bits is explained by the fact that four floating point values can be written by a single SSE instruction:

_mm_store_ps The instruction writes four floating point values to a given memory address.

The reverse conversion, from floating point to binary vector, requires rounding of the floating point values. This operation can also be executed in groups of four:

_mm_round_ps The instruction rounds four floating point values.

The results are than converted to four 8 bit integers:

_mm_cvtps_pi8 Four 32 bit floating point values are converted to four 8 bit integers, where each is stored in the lower byte of a 16 bit integer.

[1] `http://software.intel.com/en-us/articles/intel-intrinsics-guide`, accessed August 21, 2013

Input:
binary features $B = (\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_m)$
number of cluster centers k with $k \leq m$
distance metric $d(\boldsymbol{b}_i, \boldsymbol{d}_j)$
Result:
binary cluster centers $D = (\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k)$
Data:
cluster centers $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)$
Function:
$nearestCenterIndex(\boldsymbol{b}_i, D) = \arg\min_j d(\boldsymbol{b}_i, \boldsymbol{d}_j), \boldsymbol{d}_j \in D$

$$toFloat(\boldsymbol{d}) = (x_1, x_2, \ldots, x_n), x_i = \begin{cases} 1.0 & \text{if } y_i = 1 \\ 0.0 & \text{otherwise} \end{cases}, y_i \in \boldsymbol{d}$$

$$toBinary(\boldsymbol{c}) = (y_1, y_2, \ldots, y_n), y_i = \begin{cases} 1 & \text{if } x_i \geq 0.5 \\ 0 & \text{otherwise} \end{cases}, x_i \in \boldsymbol{c}$$

$weightCenter(i, m) \in [0, 1]$
Algorithm:
initialize C
for $i = 1, 2, \ldots, k$ **do**
$\boldsymbol{d}_i = toBinary(\boldsymbol{c}_i)$
end
for $i = 1, 2, \ldots, m$ **do**
$j = nearestCenterIndex(\boldsymbol{b}_i, D)$
$\boldsymbol{c}_j = (1.0 - weightCenter(i, m)) * toFloat(\boldsymbol{b}_i) + weightCenter(i, m) * \boldsymbol{c}_j$
$\boldsymbol{d}_j = toBinary(\boldsymbol{c}_j)$
end

Algorithm 2: The kShifts algorithm adapted for binary features.

The result is a 64 bit variable with again 16 possible values. A *switch*-statement in a loop is used to efficiently concatenate 4 bit groups to the complete binary feature vector.

Another important factor is that the addresses of all memory locations are 16 byte aligned. This allows to use read and write instructions for aligned memory addresses which are faster than the matching instructions for unaligned memory addresses.

To utilize all CPU cores in a system OpenMP is used. Centers are distributed over all computational units and every unit determines the nearest center from its assigned centers for a given data point. This happens in parallel. Then the nearest center of these remaining centers is selected.

For calculating the updated center OpenMP is used as well. In this case the vector's dimensions are split and distributed over computational units. E.g. for a 256 dimensional vector each core of a four core CPU calculates the values of 64 dimensions.

Thus kShifts clustering is implemented in a highly concurrent fashion. The adaptions for binary feature vectors are supported by special CPU instructions which allow for a very efficient implementation.

Evaluation

For the adapted kShifts version a series of evaluation experiments are conducted. As test data orb features are extracted from the Oxford Buildings Dataset. The maximum number of features is restricted to 1,000. The OpenCV 2.4[2] implementation is used with a two level pyramid and default options otherwise. From 5,052 images, 4,939,099 features with a size of 256 bit, or 32 byte, are extracted. This equals approximately 151 megabyte of data. The set of features is shuffled once. According to the transition of the weighting functions feature vectors with a low index influence the centers more than those with a high index. Without randomization data points are processed in image specific order. This could lead to biased results and even more so if images are sorted by visual appearance (e.g. photos of a shooting can be expected to be visually similar). For the following experiments only one iteration of kShifts is executed and thus every feature vector touched only once. Because the start points for the centers are randomized, 30 runs are executed for every configuration.

The first experiment aims at the decision which weighting function to choose. A fixed data point order is used for each run to keep results more stable. Therefore only the starting positions of the centers are randomized. A uniform distribution is used for the values of the floating point version of the centers with values in $[0, 1]$. For the generation of the random values the C++ implementation defined default "engine" is used. The random floating point centers are than converted to binary centers as can be seen in algorithm 2. As quality measure the WCSS is used on the binary version of the feature vectors and centers with the hamming distance

$$WCSS = \sum_{i=1}^{k} \sum_{\boldsymbol{v}_j \in G_i} |\boldsymbol{c}_i - \boldsymbol{v}_j|_2 \tag{4.11}$$

$$= \sum_{i=1}^{k} \sum_{\boldsymbol{v}_j \in G_i} popcount(xor(\boldsymbol{c_i}, \boldsymbol{v_j})) \tag{4.12}$$

where G_i is the cluster represented by center $\boldsymbol{c}_i$. Four different values are chosen for a:

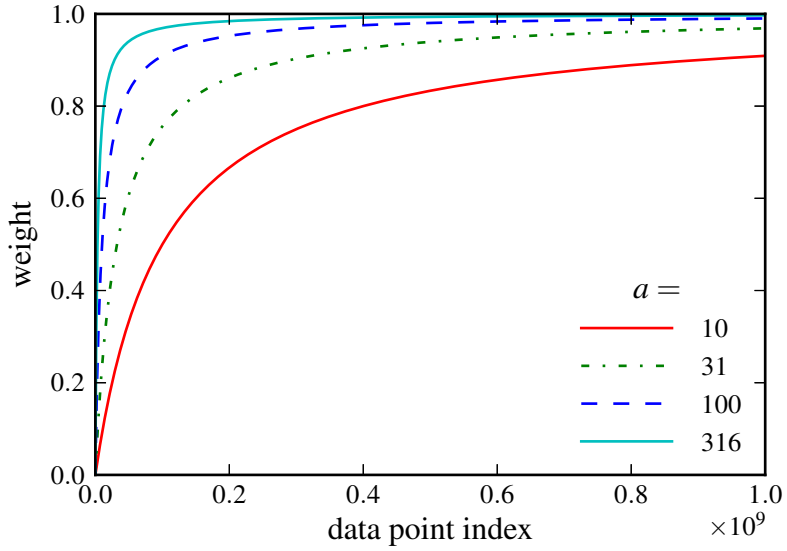
Original 10, 31, 100 and 316

Linear and quadratic 5, 30, 55 and 80

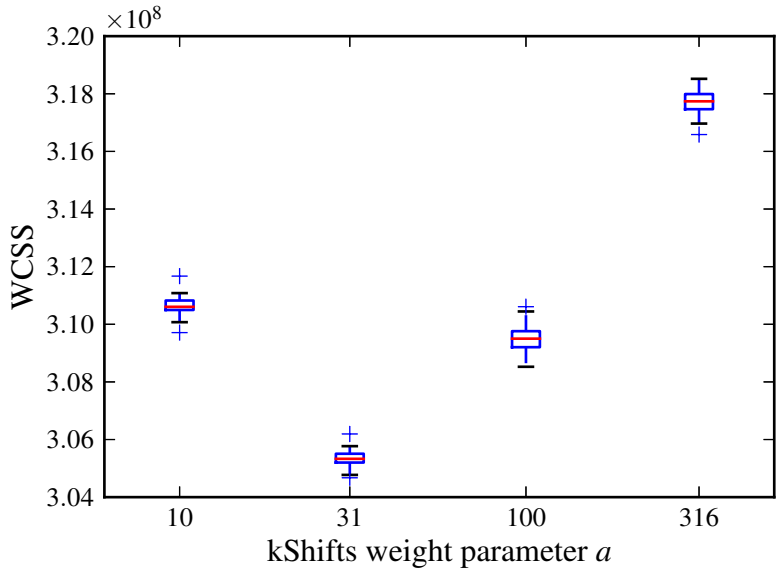
Figures 4.2, 4.3 and 4.4 show the weighting functions and corresponding box plots for the WCSS over each value.

To verify the significance of the results, Welch's t-test is used. Table 4.1 compares the results of different weighting functions among themselves. For the original and linear weighting functions the results show that $a = 31$ and $a = 55\%$ are best. The results of 30 runs with quadratic weighting functions do not lead to a definite decision (probability of equal mean values smaller than 1%). The weight parameter a of 31 is used because it is between

[2] `http://opencv.willowgarage.com`, accessed August 21, 2013

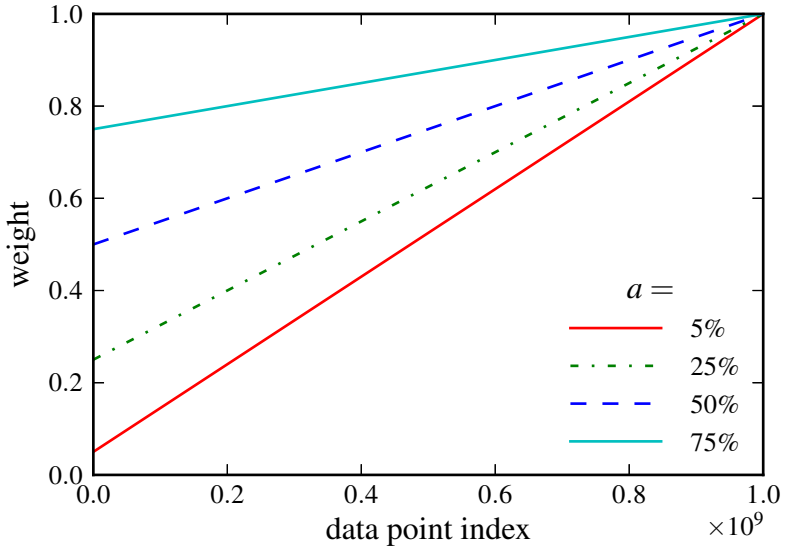


(a) Original weighting

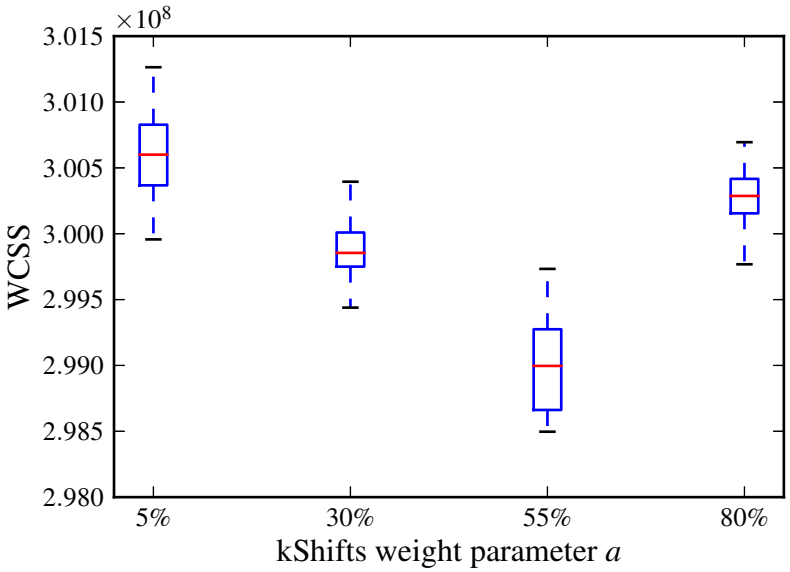


(b) Box plots

Figure 4.2: Original weighting: WCSS over kShifts weight parameter a. 30 runs with fixed data point order and random start points.

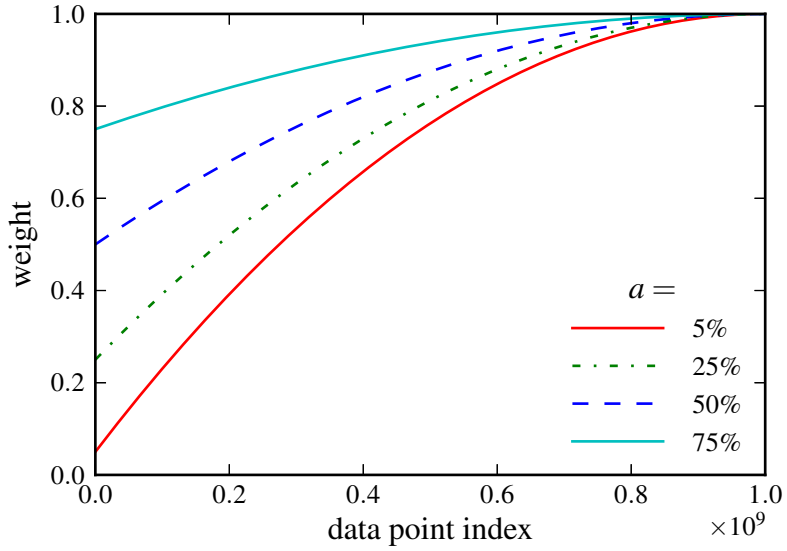


(a) Linear weighting

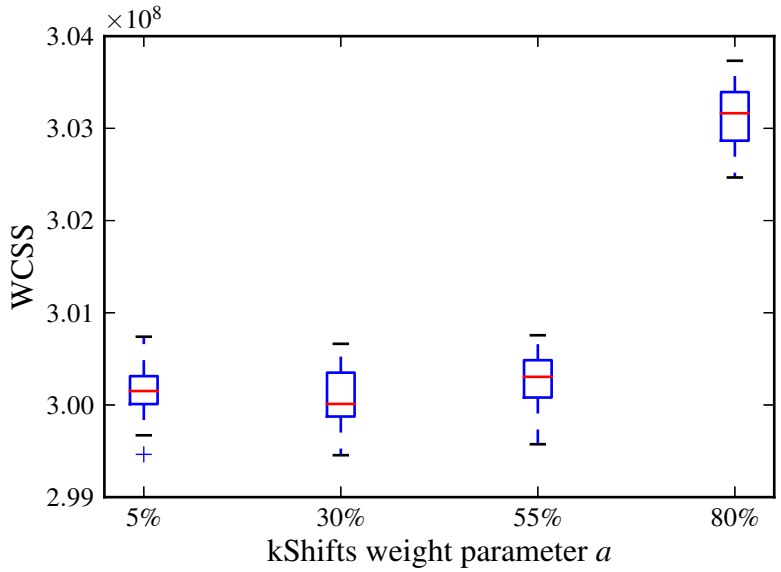


(b) Box plots

Figure 4.3: Linear weighting: WCSS over kShifts weight parameter a. 30 runs with fixed data point order and random start points.



(a) Quadratic weighting



(b) Box plots

Figure 4.4: Quadratic weighting: WCSS over kShifts weight parameter a. 30 runs with fixed data point order and random start points.

the other two equally good performing parameter values (5 and 55). Table 4.2 compares the best weighting functions against each other. With a high probability the linear weighting function, with $a = 55\%$, gives the best results. Therefore the remaining kShifts experiments are executed with this configuration.

Table 4.1: Welch's t-tests for different kShifts weighting functions. The values represent the probability of null hypothesis of equal WCSS mean values, $H_0 : \overline{X}_i = \overline{X}_j,\ i, j \in \{a_1, a_2, \ldots, a_n\}$.

a	10	31	100	316
10	100.0%	0.0%	0.0%	0.0%
31		100.0%	0.0%	0.0%
100			100.0%	0.0%
316				100.0%

(a) Original weighting

a	5%	30%	55%	80%
5%	100.0%	0.0%	0.0%	0.0%
30%		100.0%	0.0%	0.0%
55%			100.0%	0.0%
80%				100.0%

(b) Linear weighting

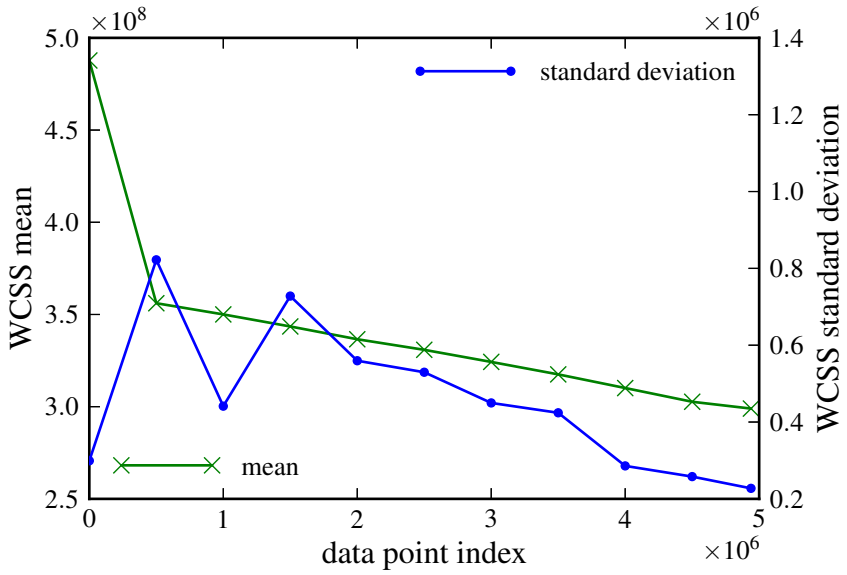
a	5%	30%	55%	80%
5%	100.0%	26.1%	15.5%	0.0%
30%		100.0%	1.9%	0.0%
55%			100.0%	0.0%
80%				100.0%

(c) Quadratic weighting

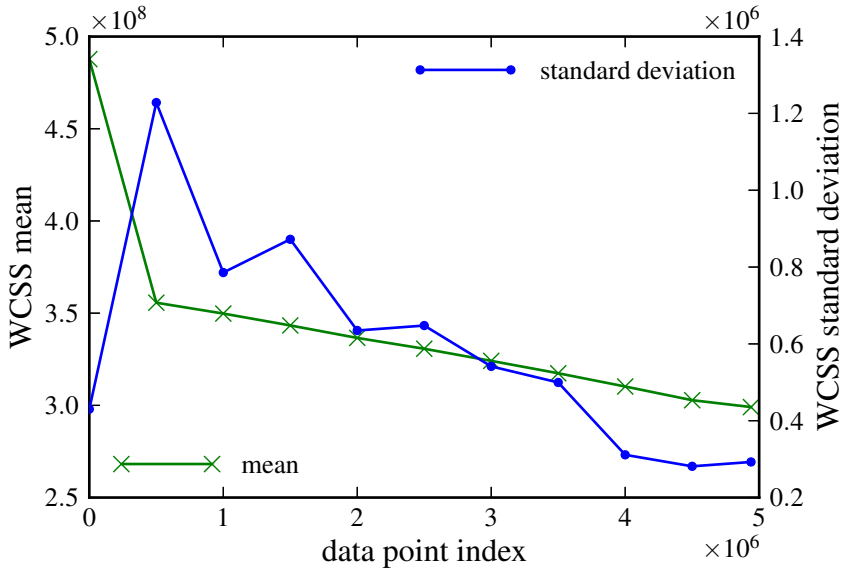
Table 4.2: Welch's t-tests for the best weighting functions. The values represent the probability of null hypothesis of equal WCSS mean values, $H_0 : \overline{X}_i = \overline{X}_j,\ i, j \in \{original_{31}, linear_{55\%}, quadratic_{30\%}\}$.

	$original_{31}$	$linear_{55\%}$	$quadratic_{30\%}$
$original_{31}$	100.0%	0.0%	0.0%
$linear_{55\%}$		100.0%	0.0%
$quadratic_{30\%}$			100.0%
$\overline{X}$	3.053e+08	2.990e+08	3.001e+08

The second experiment is conducted to evaluate the influence of data point order on the WCSS. Therefore the WCSS is calculated after a specified number of data points. Figure 4.5 shows the mean and standard deviation across the 30 runs. As expected the results with randomized data order show an increased standard deviation. Even so the final mean values are very similar which shows that the kShifts algorithm produces stable results for shuffled data points.



(a) Fixed data point order



(b) Randomized data point order

Figure 4.5: kShifts: Mean and standard deviation of WCSS over data point index. For (a) the data point order is fixed, while for (b) for every run a randomized data point order is generated.

4.1.2 kMeans

The standard kMeans algorithm is applied to floating point feature vectors. An adaption for binary feature vectors is proposed in the following.

Implementation

The implementation (algorithm 3) follows that of kShifts. The conversion from binary to integer feature vector and distance calculation are analogous to the floating point conversion and distance calculation in binary kShifts. Basically assigned binary feature vectors are converted to integer vectors and summed. The rounded mean is built by looking at each component of the summed integer vector. If a component is higher than or equal to half the number of assignments of this center, the resulting component in the binary vector is 1 and 0 otherwise. This is the same as calculating the mean in floating point and rounding it followed by a floating point to binary conversion.

Evaluation

In contrast to the kShifts algorithm the data point order is of no concern for the kMeans algorithm. For the evaluation of the binary kShifts algorithm the same data and quality measure is used as for the kShifts algorithm. Again 30 runs are performed. The algorithm is set to terminate if no center changed after a complete iteration. The maximum number of iterations is set to 100, but never reached. Figure 4.6 shows the mean and standard deviation of the WCSS after each iteration. Only one run reached above 37 iterations and therefore the standard deviation drops to 0 from 38 iterations on.

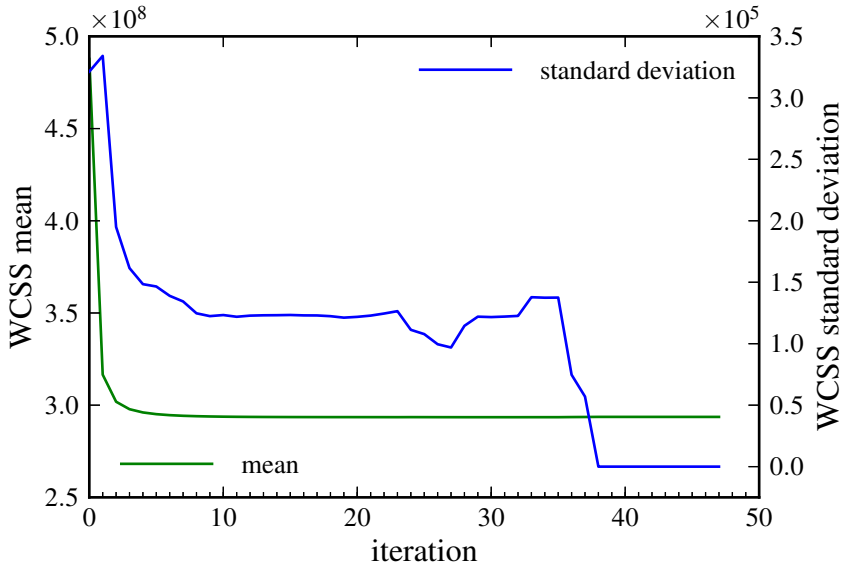


Figure 4.6: kMeans: Mean and standard deviation of the WCSS over iterations. 30 runs with random start points were performed.

Input:
binary features $B = (\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_m)$
number of cluster centers k with $k \leq m$
distance metric $d(\boldsymbol{b}_i, \boldsymbol{d}_j)$
number of iterations $iter$
Result:
binary cluster centers $D = (\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k)$
Data:
integer cluster centers $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_k)$
cluster center counts $I = (c_1, c_2, \cdots, c_k),\ c_i \in \mathbb{N}$
Function:
$nearestCenterIndex(\boldsymbol{b}_i, D) = \arg\min_j d(\boldsymbol{b}_i, \boldsymbol{d}_j), \boldsymbol{d}_j \in D$
$toInt(\boldsymbol{d}) = (x_1, x_2, \ldots, x_n),\ x_i = \begin{cases} 1 & \text{if } y_i = 1 \\ 0 & \text{otherwise} \end{cases}, y_i \in \boldsymbol{d}$
Algorithm:
initialize D
for $i = 1, 2, \ldots, iter$ **do**
  for $j = 1, 2, \ldots, k$ **do**
    $\boldsymbol{c}_j = \mathbf{0}$
  end
  for $j = 1, 2, \ldots, m$ **do**
    $l = nearestCenterIndex(\boldsymbol{b}_j, D)$
    $\boldsymbol{c}_l = \boldsymbol{c}_l + toInt(\boldsymbol{b}_j)$
    $c_l = c_l + 1$
  end
  for $j = 1, 2, \ldots, k$ **do**
    $\boldsymbol{d}_j = (x_1, x_2, \ldots, x_n),\ x_i = \begin{cases} 1 & \text{if } y_i >= \frac{c_j}{2} \\ 0 & \text{otherwise} \end{cases}, y_i \in \boldsymbol{c}_j$
  end
end

Algorithm 3: The basic kMeans algorithm adapted for binary features.

4.1.3 Comparison

Finally the kShifts and kMeans algorithm are compared in terms of efficiency and cluster quality. Figure 4.7 shows box plots, comparing the WCSS for kMeans, after a specific number of runs (2, 3 and 18), with the WCSS for kShifts with linear weighting function and weight parameter $a = 55\%$ and randomized data order. The value of 18 iterations is used because all 30 runs reach at least the 18$^{\text{th}}$ iteration. Additionally table 4.3 shows the significance of the tests accordingly to Welch's t-test. The results show that one iteration of kShifts, touching each data point once, reaches a WCSS between two and three kMeans iterations. The complexity of kShifts and kMeans is equal as for each algorithm the nearest center for each data point has to be found during one iteration. Practically one iteration of binary kMeans takes less time than one iteration of kShifts because calculating the mean of the accumulated centers is computationally less expensive than updating the centers in

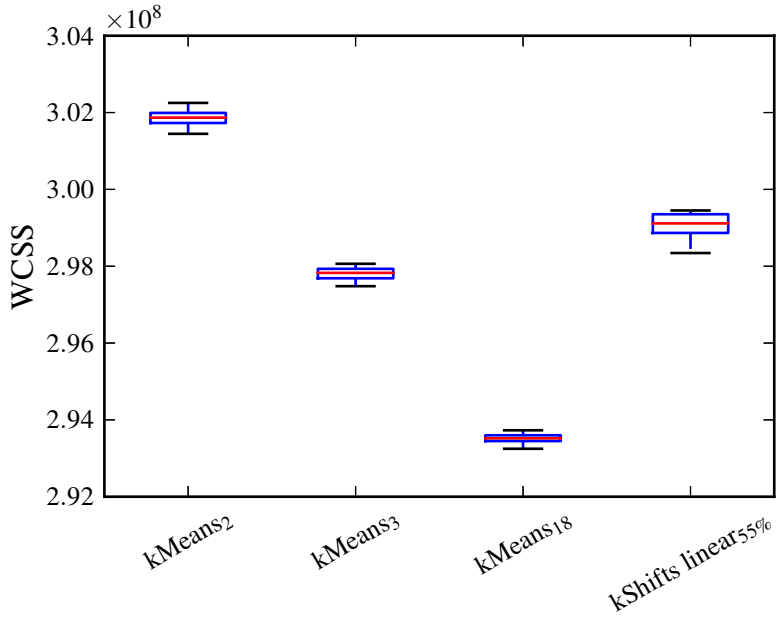


Figure 4.7: Comparison of the best kShifts configuration (linear$_{55\%}$) and binary kMeans with 2, 3 and 18 iterations.

kShifts. kShifts requires the conversion of a center to floating point and back after the update, while no conversion of the center is necessary for binary kMeans. Additionally kShifts requires the shuffling of data points which kMeans does not. An efficient way of shuffling would be to group the features by image (which is the form in which they are obtained during feature extraction) and shuffle only these groups. This would result in only minor overhead. Figure 4.5 shows that for kShifts the number of visited data points can be reduced to, e.g. 60%, with a predictable loss in clustering quality. Similar multiple kMeans iterations can be executed on a reduced data point set, but this is not evaluated in this thesis. The final decision, according to the presented results, is between at least three kMeans or one kShifts iteration and depends on the specific setup in terms of hardware and runtime versus cluster quality considerations.

Table 4.3: Welch's t-tests for kShifts and kMeans with different parameters. The values represent the probability of null hypothesis of equal WCSS mean values, $H_0 : \overline{X}_i = \overline{X}_j,\ i, j \in \{kMeans_2,\ kMeans_3,\ kMeans_{18},\ kShifts\ linear_{55\%}\}$.

	kMeans$_2$	kMeans$_3$	kMeans$_{18}$	kShifts linear$_{55\%}$
kMeans$_2$	100.0%	0.0%	0.0%	0.0%
kMeans$_3$		100.0%	0.0%	0.0%
kMeans$_{18}$			100.0%	0.0%
kShifts linear$_{55\%}$				100.0%

4.2 Nearest-Neighbour Search

Finding the nearest-neighbour of a vector in high dimensional spaces is a long standing problem of computer science. Muja and Lowe [48] state:

> "For high-dimensional spaces, there are often no known algorithms for nearest neighbor search that are more efficient than simple linear search. As linear search is too costly for many applications, this has generated an interest in algorithms that perform approximate nearest neighbor search, in which non-optimal neighbors are sometimes returned. Such approximate algorithms can be orders of magnitude faster than exact search, while still providing near-optimal accuracy".

An efficient nearest-neighbour search is critical for two stages of the bags of visual words approach. First for the clustering and second for the generation of the bags of visual words. In both stages data points (or vectors) have to be assigned to their nearest centroid. For the kShifts clustering algorithm one centroid is updated after every data point which renders building a special search structure infeasible. The kMeans algorithm on the other hand updates the centroids only after a complete iteration over all data points. The problem however is that space partitioning algorithms (kd-Trees) like FLANN [48] are aimed at floating point vectors and not binary features. A simple conversion would lead to features with even higher dimensionality (typically 256 or 512) than standard sift features (128). Furthermore figure 4.8 shows the speedup of FLANN over linear search for high dimensional random vectors. These results show that FLANN degrades to linear search if dimensionality is high

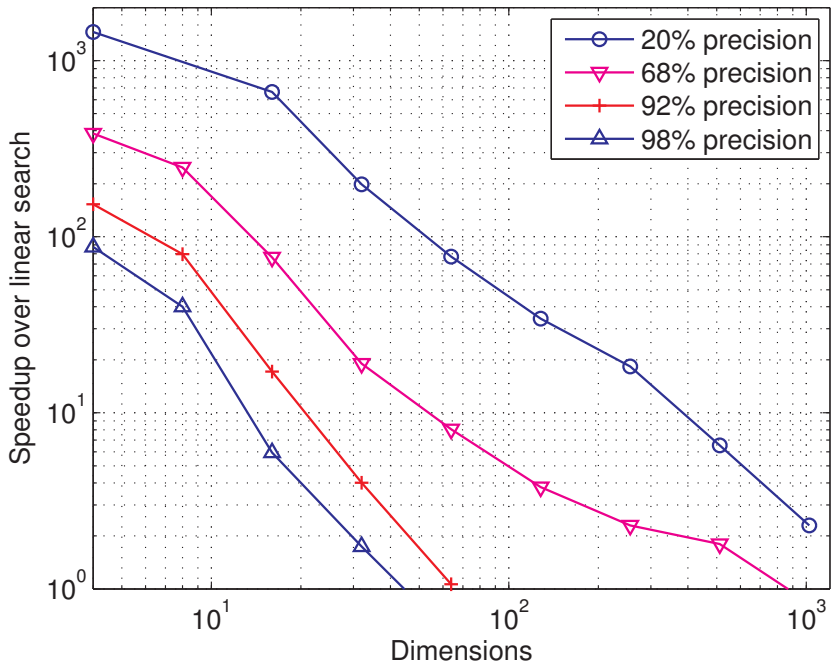


Figure 4.8: Speedup of FLANN [48] over linear search for high dimensional random vectors.

and a certain accuracy is required. The FLANN algorithm becomes more efficient as the correlation of vector increases. This is however in contradiction to visual features which aim to be as decorrelated as possible to be highly discriminative [62].

Because of these reasons linear search is used in this thesis. The CPU implementation is straight forward, using OpenMP for parallelization. To additionally utilize GPU resources a nearest-neighbour search, implemented in OpenCL, is used as well. The source code is listed in appendix A.1.

4.3 Retrieval System

Figure 4.9 shows the implemented retrieval system. The server accepts requests by the user. These request are typical management tasks like adding and removing images, training the database, etc. Additionally the server can be queried with near-duplicate images and returns corresponding originals. The server stores requests and images in a database and provides the workers with work packages (e.g. image id and task) and corresponding configuration. The worker retrieve their needed data (e.g. image data) from the database and store the finally processed results in the database. Sharding can be used to distribute the workload over several databases when the read to write ratio is high, as is the case for querying. After all workers have finished their tasks, the server collects the processed data from the database and returns the result. Heterogeneous workers (e.g. with and without additional GPU) can log on and off as needed and the system can be extended without further setup costs.

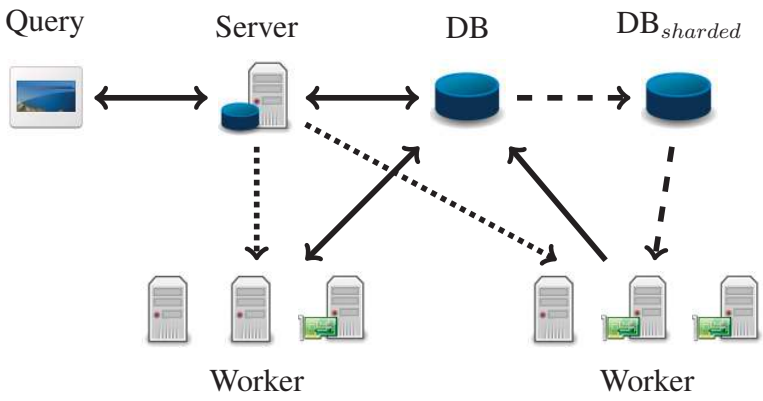


Figure 4.9: The implemented retrieval system.

4.4 Bags of Visual Words

For the first series of experiments a subset of the Copydays image set is used. The Copydays images are complemented with 5,000 images randomly selected from the Flickr1M image set. These images have an average resolution of 2.5 mega pixel, with a standard deviation of 3 mega pixel. The resulting image set is called Copydays + Flickr5k and is used as

originals image set. To build the query image set, two image transformations are applied to the Copydays images:

Cropping The size of the image is reduced, while aspect ratio and image center remain fixed.

Rotation The image is rotated around its center and then cropped so that the aspect ratio remains fixed. Figure 4.10 shows the image size as function of the rotation angle, when images with an aspect ratio of 4:3 are assumed.

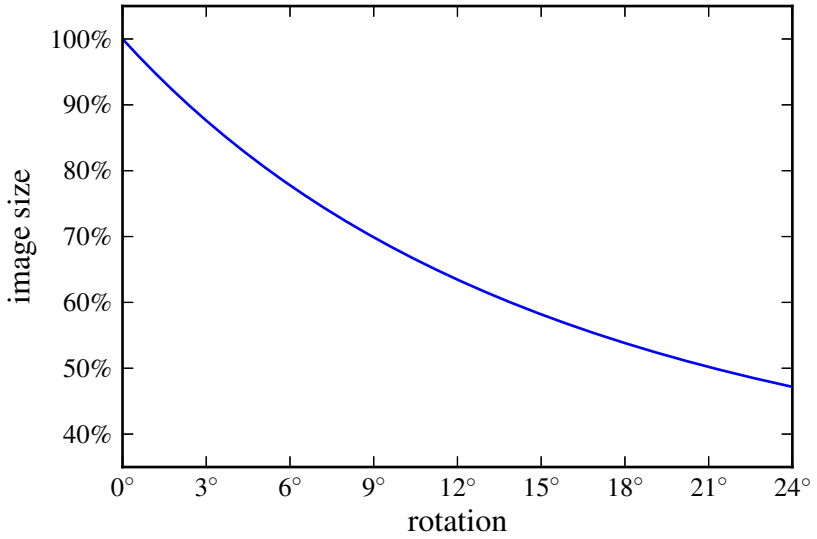


Figure 4.10: The image size as function of the image rotation angle in degree, when the image is cropped so that the aspect ratio remains the same after rotation.

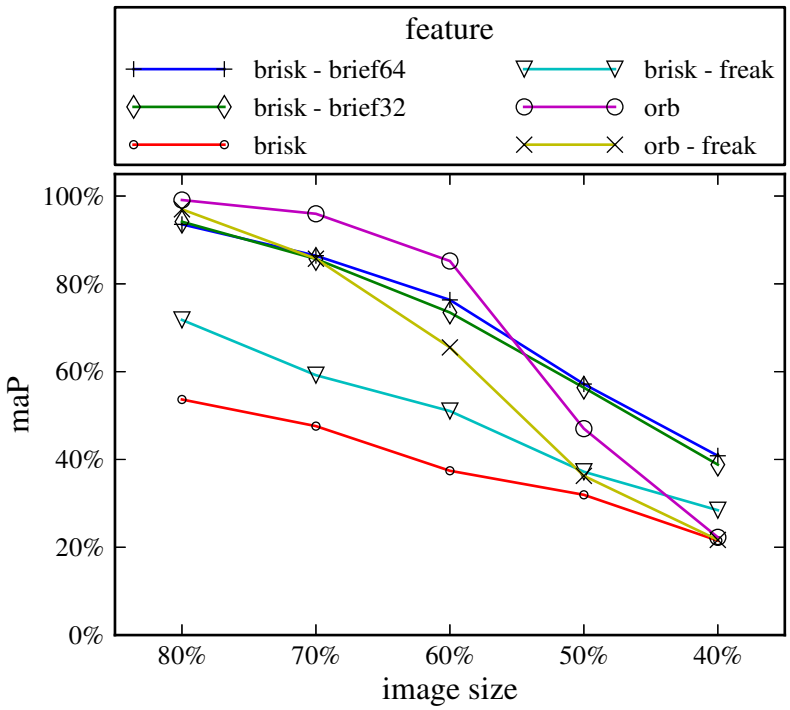
Before processing all images are scaled down so that the maximum dimension is 1024 pixel, while the aspect ration remains fixed. An exception to this are the experiments where the influence of the image resolution is examined (figure 4.13). If in the following two names are given as feature description, the first is the feature detector and the second the feature descriptor.

Bags of visual words are stored as normalized, sparse vectors. Tf-idf weighting is applied as proposed by Philbin et al. [54]. The similarity of two images is obtained by calculating the dot product of the two corresponding bags of visual words.

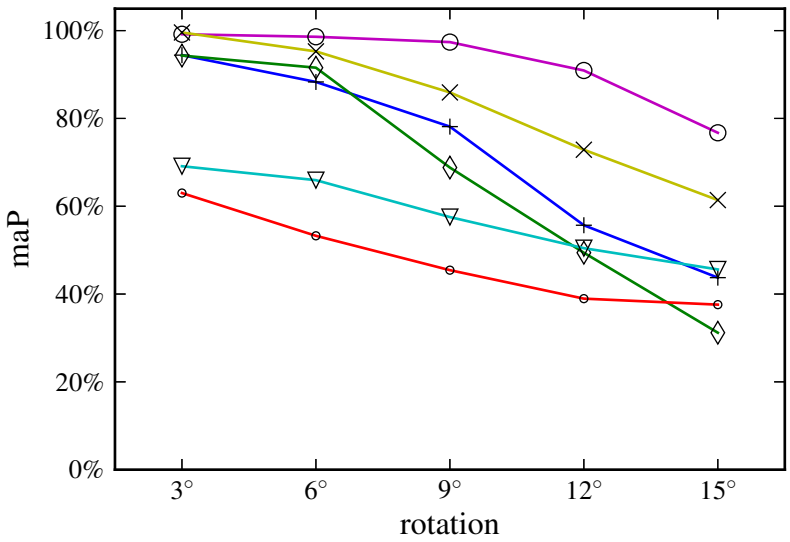
The first experiment determines the best suited binary features. A series of queries are performed with different feature detectors and descriptors while other parameters remain fixed. Table 4.4 shows the average number of features per original image.

Table 4.4: The mean of the number of features per original image for the Copydays + Flickr5k image set.

orb	brisk-brief32	brisk-brief64	brisk	brisk-freak	orb-freak
880.5	806.2	1612.6	1653.7	1492.7	1324.1



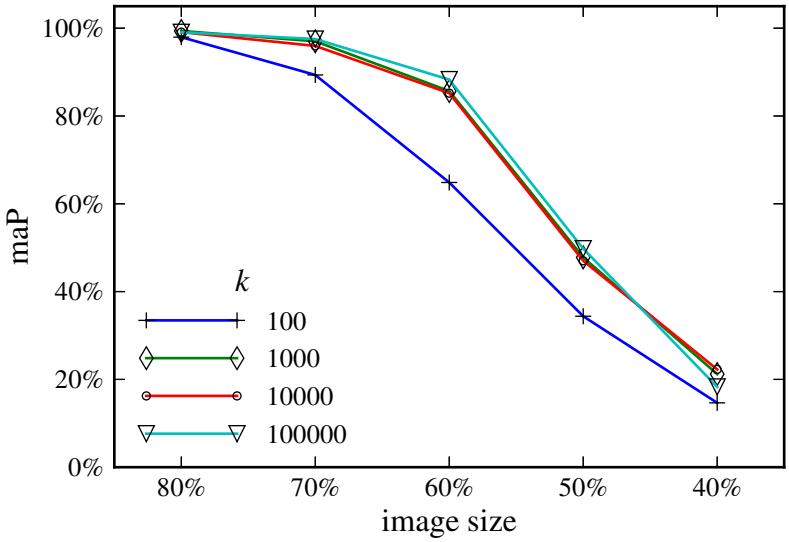
(a) cropping



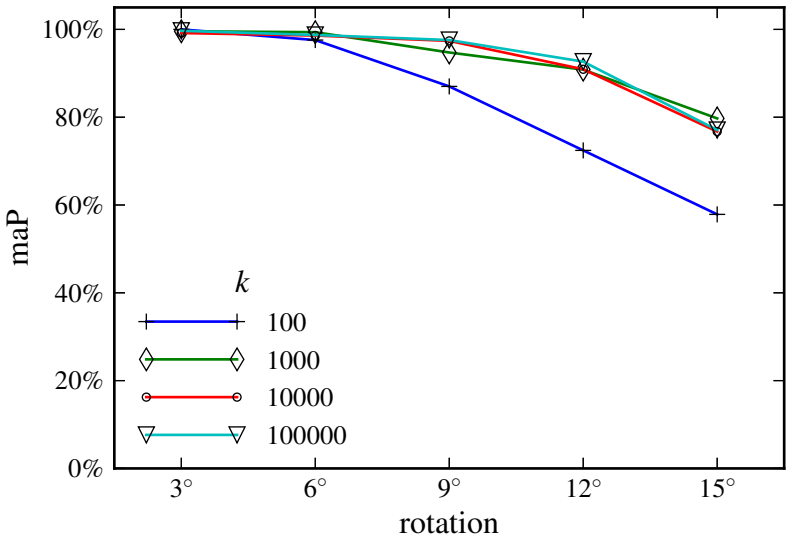
(b) rotation

Figure 4.11: Copydays + Flickr5k: The mean average precision (maP) for querying with cropped and rotated images. Different visual features are used.

The configuration used is 10,000 visual words and kShifts linear$_{55\%}$ clustering algorithm. Figure 4.11 shows the mean average precision (maP) over increasing image transformation for cropped and rotated query images. The results show that ORB features [62] perform best for rotated query images. For cropped query images ORB accuracy drops significantly from 50% on. A possible solution to this would be to increase the number of pyramid levels for the ORB feature calculation, which is set to two for these experiments. Because of these results ORB features are used for the following experiments. The second experiment determines



(a) cropping

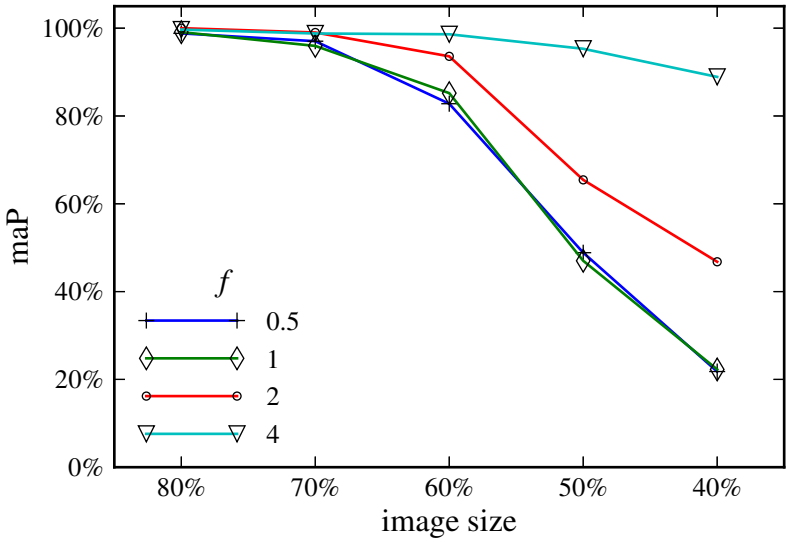


(b) rotation

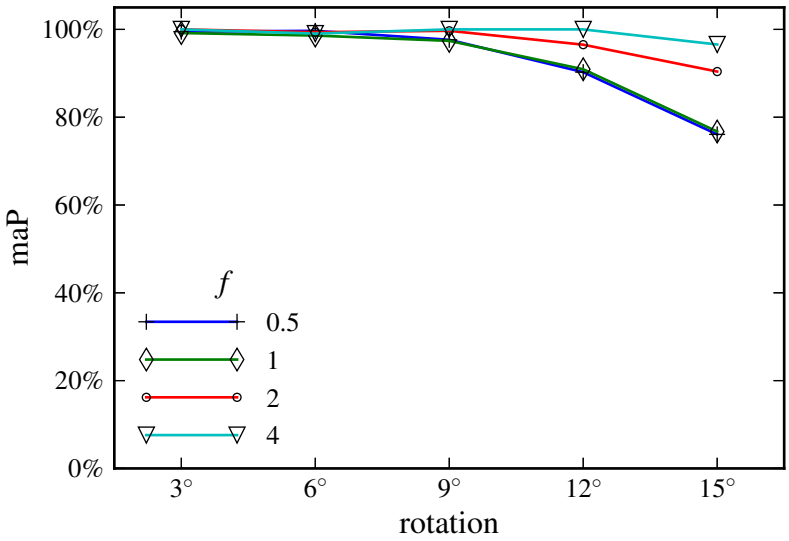
Figure 4.12: Copydays + Flickr5k: The mean average precision (maP) for querying with cropped and rotated images. Different numbers of visual words (k) are used.

the influence of the number of visual words. According to Csurka et al. [18] a higher number of visual-features should lead to increased accuracy. On the other hand the efficiency of the overall system is decreased as processing time for clustering and calculation of bags of

visual words is directly proportional to the number of visual-features. ORB features and the kShifts linear$_{55\%}$ clustering algorithm are used. Figure 4.12 shows the maP over increased image transformations for different number of visual words. The results show an increase in accuracy from 100 to 1,000 visual words. Using more than 1,000 visual words does however not lead to an improvement. It can be assumed that for a specific size of an image set a specific number of visual words is optimal and an increase does not raise search accuracy but lower overall system efficiency. The third experiment determines the influence of image



(a) cropping



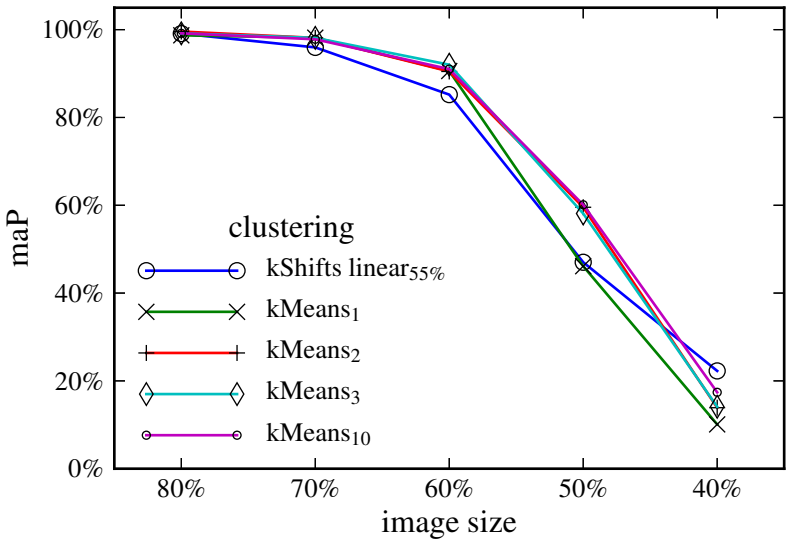
(b) rotation

Figure 4.13: Copydays + Flickr5k: The mean average precision (maP) for querying with cropped and rotated images. Different image resolutions (resolution factor f) are used.

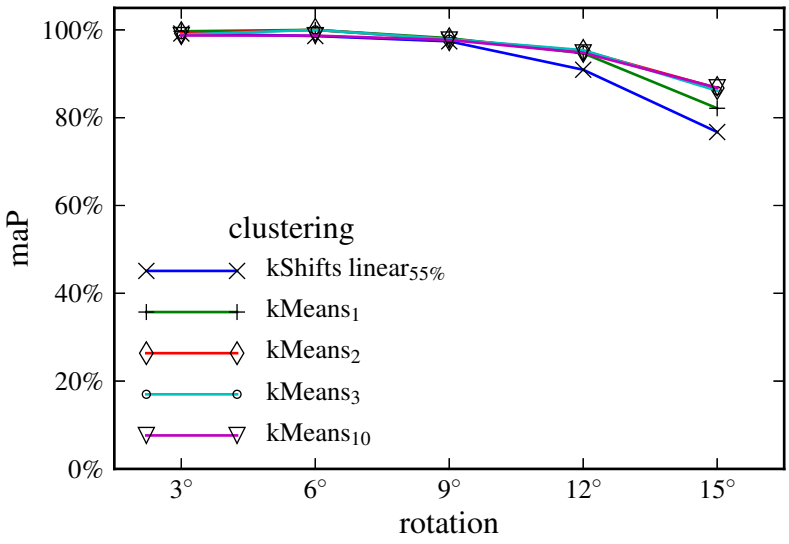
resolution and thus the number of visual features per image. Therefore the image resolution was scaled by a factor f: 0.5, 1, 2 and 4 (where the standard resolution corresponds to the resolution with maximum image dimension $\leqslant 1024$). The maximum number of ORB fea-

tures was set to 1000, 1000, 2000 and 4000 respectively. ORB features, the kShifts linear$_{55\%}$ clustering algorithm and 10,000 visual words are used. Figure 4.13 shows the maP over increased image transformations for different image resolutions. The results show the direct influence of image resolution and number of visual features on the search accuracy. Half of the image resolution is enough to obtain ≈ 1000 visual features, which explains the equality of the results for $f = 0.5$ and $f = 1$.

The last experiment determines the influence of the clustering quality (WCSS). Therefore the four configurations from figure 4.7 are used with ORB features and 10,000 visual words. Figure 4.14 shows the maP over increased image transformations for different clustering algorithms. The results for kMeans$_3$ and kMeans$_{10}$ are nearly on par, suggesting that a



(a) cropping



(b) rotation

Figure 4.14: Copydays + Flickr5k: The mean average precision (maP) for querying with cropped and rotated images. Different clustering algorithms are used with the same overall system configuration. Configuration: ORB features, 10,000 visual words.

specific number of iterations is sufficient for the clustering of a specific number of visual-features. Unexpected the kShifts $\text{linear}_{55\%}$ algorithm is worse than even kMeans_1 for all results but cropping to 40%. However it has to be mentioned that only one run was performed for each algorithm and that the centroids are initialized randomly. Even so the results suggest that kMeans_3 is a better choice over kShifts $\text{linear}_{55\%}$ in respect to efficiency and accuracy.

4.5 Comparison

The final Bags of Visual Words setup is tested with two commonly used databases: The Inria Copydays image set and The Oxford Buildings Dataset. Both image sets are expanded with 1,000,000 distractor images from the Flickr1M image set. The mean average precision is calculated for specific image queries and compared against state of the art methods.

4.5.1 Inria Copydays

The main reason this image set is used in this thesis is its widespread use in relevant papers, which in turn allows comparison of results. However rotation is not among the suggested transformations, even so it is a reoccurring transformation and important part of visual feature detector and descriptor capabilities. Therefore rotation was added as an additional transform. The image set together with the distractor images (Flickr1M) consists of $\approx$ 1,000,000 images. Images are processed with a maximum resolution of 2048×2048 pixels, with larger images being scaled down. This results in $\approx 3 \cdot 10^9$ features which are clustered and processed into bags of visual words. Two different training runs are performed: one with 10,000 and one with 100,000 visual words. Both times binary kMeans_3 is used as clustering algorithm. Image queries with transformed images of the following categories were performed:

Cropping 10% – 80% of the image surface is removed.

Rotation 3°– 24° with cropping to original aspect ratio

Shrinking and Compression Shrinking to 1/16 of original size with jpeg compression from 3 – 75

Strong transformation Transformations like print and scan, blur, paint, etc.

Figures 4.15 and 4.16 show the mean average precisions for cropping and rotation over the respective grade of transform. Surprisingly the variant with 10,000 visual words performs equally, sometimes even better, to the 100,000 variant. Without further investigation this is difficult to explain. Reasons could be that so called "over-fitting" occurs or that the movement of the binary cluster centers is too limited due to their discrete nature. The results range from nearly 100% for very low cropping to about 50% for cropping to 20% of the original image size. For 10,000 visual words a precision of above 90% is obtained for up to

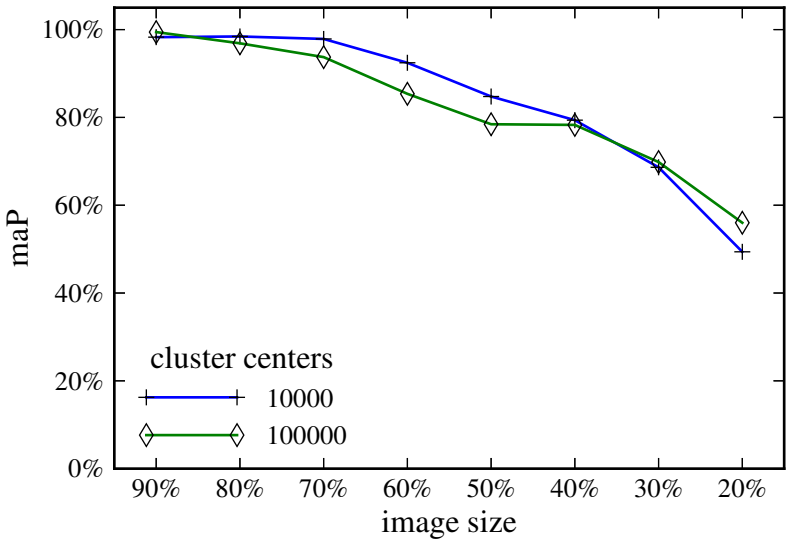


Figure 4.15: Copydays + Flickr1M: The mean average precision (maP) for querying with cropped images. Configuration: ORB features, visual words clustered with kMeans_3.

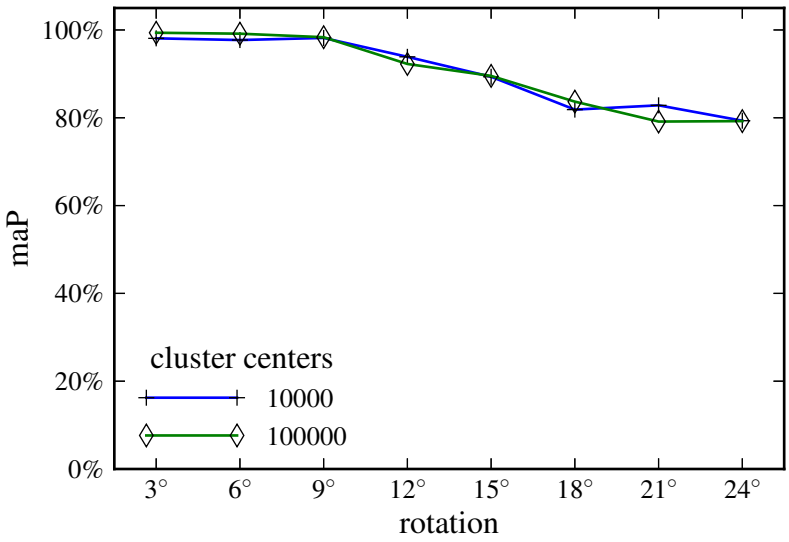


Figure 4.16: Copydays + Flickr1M: The mean average precision (maP) for querying with rotated images. Configuration: ORB features, visual words clustered with kMeans_3.

60% of cropping. The results for rotation show the same tendencies but are overall better as less cropping occurs and the binary features cope well with the transformation. The precisions do not fall under about 80%. For shrinking and compression the images are reduced to $\frac{1}{16} = 6.25\%$ of their size and then compressed with JPEG compression levels ranging from 75 to 3. This however resulted in very low precisions, near to 0% for all compression levels. It can be assumed that the combination of resolution reduction combined with the artifacts, introduced through compression, results in a transformation against which the used binary features are not robust anymore. The strong attacked images obtained from the Inria Copydays image set show better results than the shrinked and compressed. An overall mean average precision of $\approx 20.5\%$ is reached with 100,000 visual words. It can be assumed that the binary features are simply not robust against some transformations of the strong attacked images. The presented results are directly comparable to various state of the art papers. In figure 4.17 results from [33] are presented.

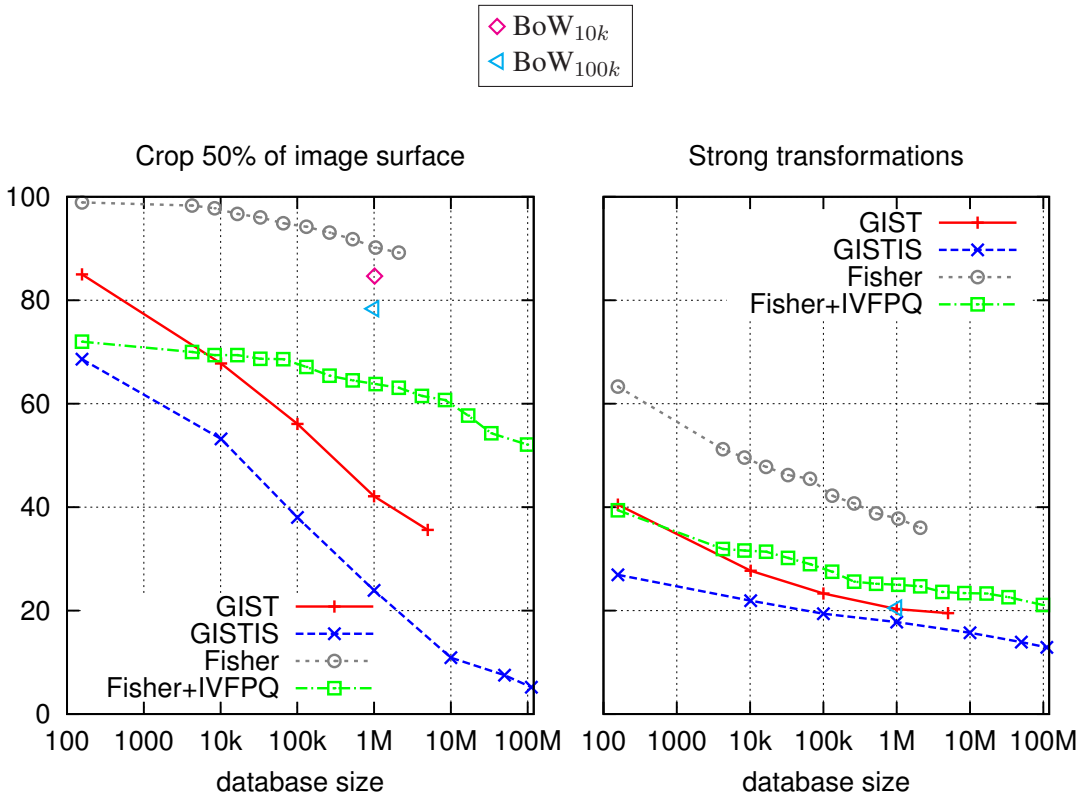


Figure 4.17: Mean average precision (maP) over database size for cropping of 50% and strong attacked images from [33] for the Inria Copydays image set. Additional markers (BoW_{10k}, BoW_{100k}) show the results obtained in this thesis.

The figure shows the mean average precision over different database sizes for a cropping of 50%. Two additional markers show the corresponding results from this thesis, which for cropping are only about 5% lower than the in the paper presented method based on fisher vectors and SIFT features. However for the strong attacked images the results of the SIFT based method show an about 20% higher precision. Furthermore the figure provides a com-

parison to systems based on a global image description, like GIST and GISTIS [23]. Global image description perform clearly worse for a cropping of 50% in this example and more or less equally for strong attacked images. Figure 4.18 from [23] however shows that good results can be achieved for weaker image transformations as cropping to 90%–70%. The figure shows the mean average precision over cropping for cropped images and over database size for strong attacked images and compares various different approaches. It can be compared directly to the results of this thesis (see figure 4.15). Figure 4.19 shows results from the same paper [23]. The mean average precision is plotted over database size and two additional markers show the results from this thesis. An overview of the compared results for cropping to 50% and 80% and strong attacked images is given in table 4.5. Hamming Embedding (HE) [32] performs best which is reasonable, as it is based on SIFT features and extends the bags of visual words approach. In this method visual words are represented as binary signatures which results in a richer representation. Additionally a geometric consistency check is performed on the highest ranked images to further improve results. Bag of Features (BOF) from [23] is a bags of visual words method based on SIFT features. Both methods use a significantly higher number of visual words (200,000) which also helps to increase precision. These comparisons show that the presented bags of visual words approach based on binary

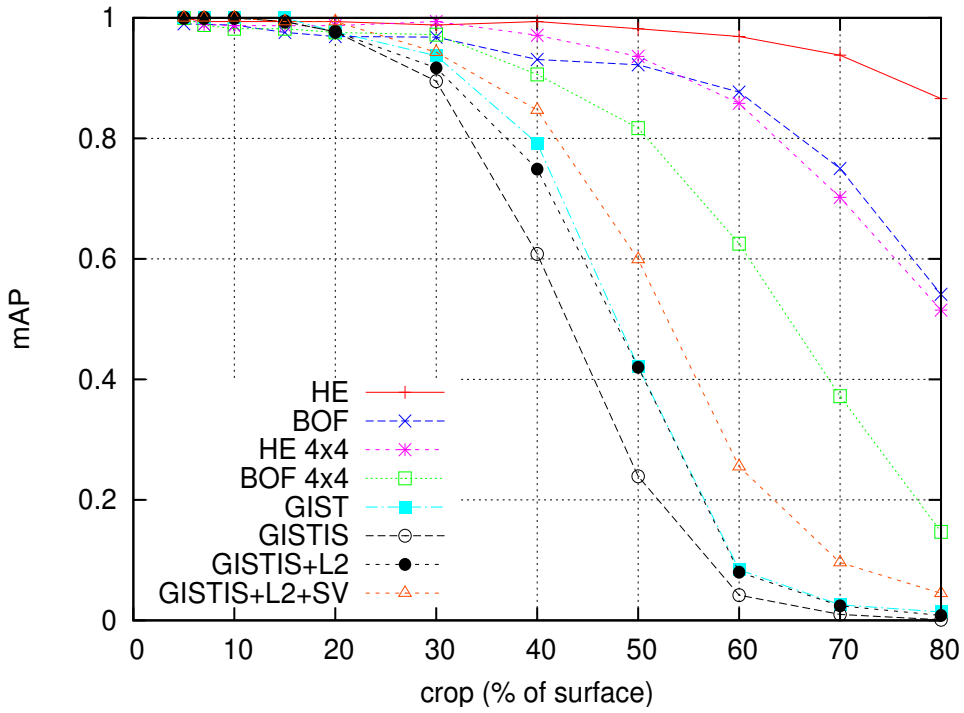
Table 4.5: A comparison of mean average precisions for the Inria Copydays image set + 1M distractor images (BoW_{10k} and BoW_{100k} are the results from this thesis with 10,000 and 100,000 visual words).

	#distractors	maP		
		crop to 50%	crop to 20%	strong attacked
BoW_{10k}	1M	84.8%	49.4%	19.7%
BoW_{100k}	1M	78.4%	56%	20.5%
HE [32]	1M	97.8%	86.6%	83.4%
BOF [23]	1M	91.9%	54.2%	52.6%
GIST [23]	1M	41.9%	1.2%	20.5%
GISTIS [23]	1M	24.0%	0.0%	17.8%
Fisher [33]	1M	90.5%	–	37.8%

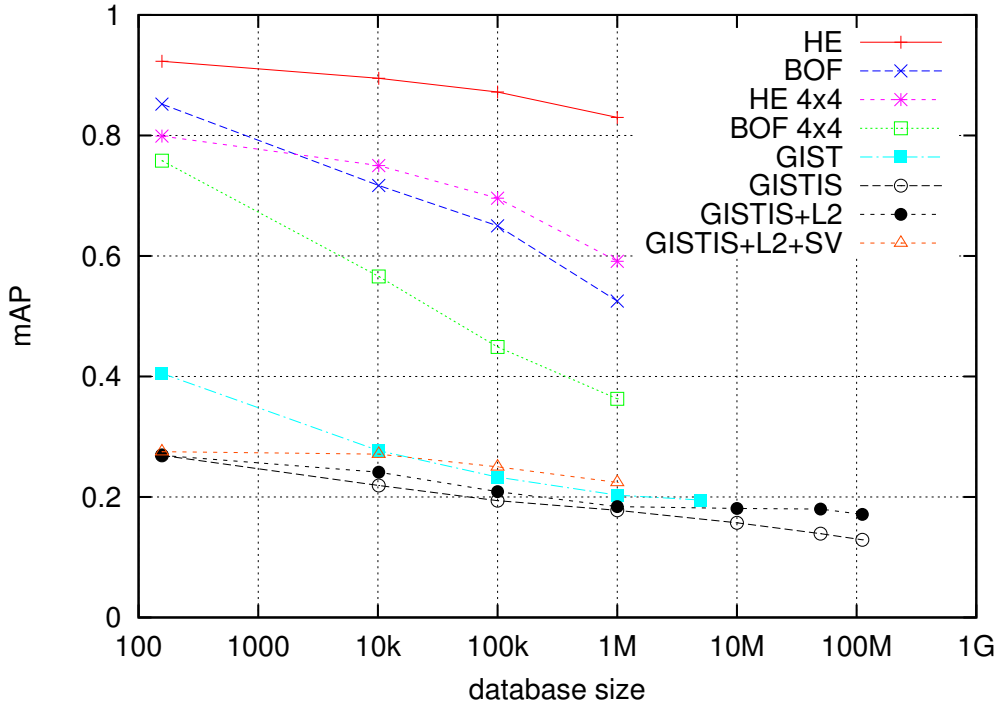
features is competitive in terms of precision with the advantage of significantly faster local features extraction. Figure 4.20 shows the results of 4 exemplary queries from the presented method. Two query images result from cropping to 20% and two from a rotation of 24° with additional cropping to the original aspect ratio.

4.5.2 The Oxford Buildings Dataset

This image set is used because of its use in related work allows for comparison of results. Additionally the high percent of photos showing buildings provide a specific challenge, as buildings tend to have similar local features (e.g. orthogonal corners), which could worsen the performance of the BoW approach. The image set together with the distractor images (Flickr1M) consists of $\approx$ 1,000,000 images. Images are processed with a maximum reso-



(a) maP over cropping



(b) maP for strong attacked images over database size

Figure 4.18: Mean average precisions for the Inria Copydays image set and 1,000,000 distractor images [23].

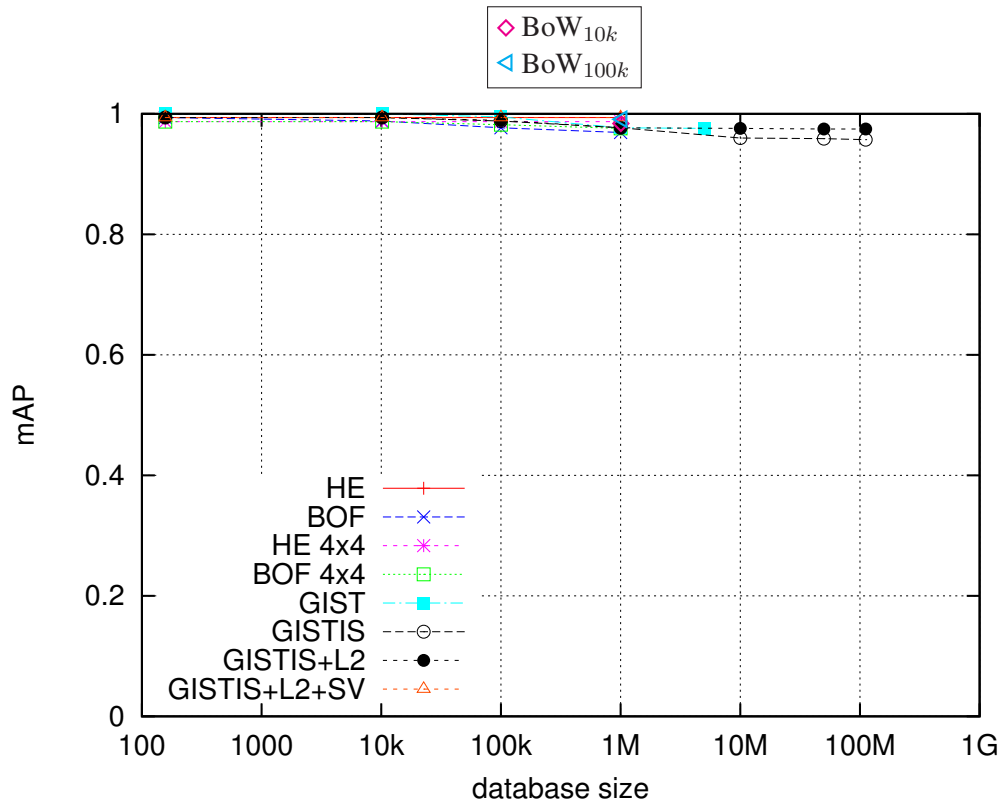


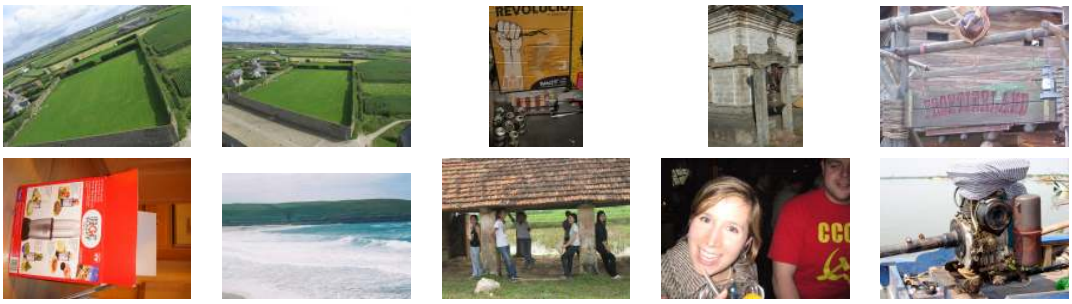
Figure 4.19: Mean average precision (maP) over database size for cropping to 80% from [23] for the Inria Copydays image set. Additional markers (BoW_{10k}, BoW_{100k}) show the results obtained in this thesis.

lution of 2048 $\times$ 2048 pixels, with large images being scaled down. Two different training runs are performed: one with 10,000 and one with 100,000 visual words. Both times binary kMeans_3 is used as clustering algorithm. 157 images (the same number of query images as for the Inria Copydays image set) were randomly selected from the Oxford Buildings Dataset and transformed into query images by the following transformations:

Cropping 10% – 80% of the image surface is removed.

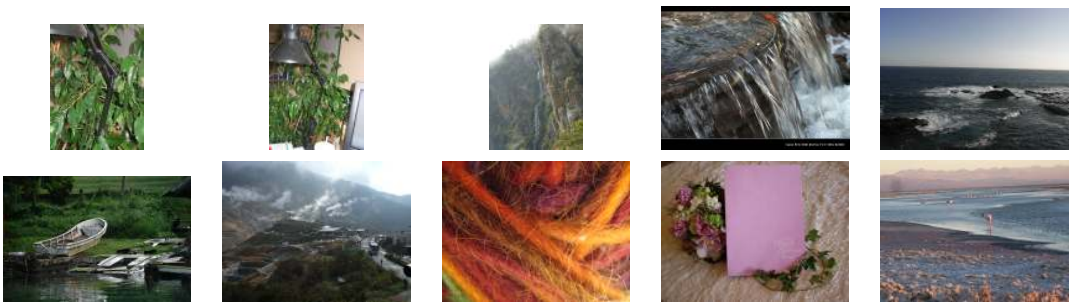
Rotation 3°– 24° with cropping to original aspect ratio

Figures 4.21 and 4.22 show the mean average precisions for cropping and rotation over the respective grade of transform. As for the Inria Copydays image set, the variant with 10,000 visual words performs surprisingly good, but the 100,000 variant has a clear advantage for cropping from 40% on. The results range from nearly 100% for very low cropping to about 50% for cropping to 20% of the original image size. For 10,000 visual words a precision of above 90% is obtained for up to 60% of cropping. The results for rotation show the same tendencies but are overall better as less cropping occurs and the binary features cope well with the transformation. The precisions do not fall under about 80%. The overall results are better than for the Inria Copydays image set, which can be explained by the average appearance of the images. The oxford images consist mainly of buildings which have high

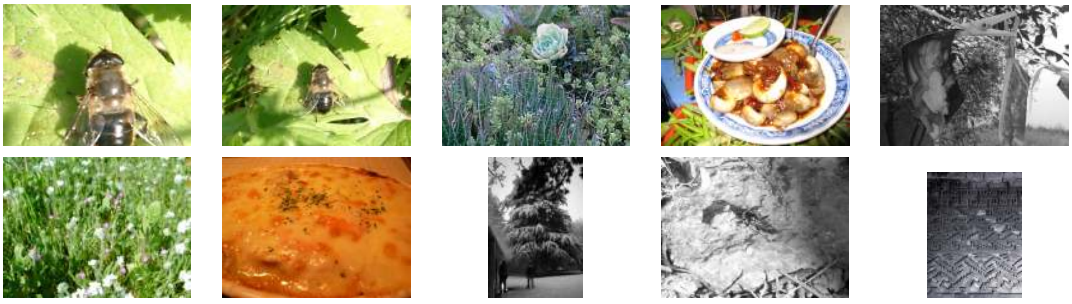


(a)

(b)



(c)



(d)

Figure 4.20: Query examples for Inria Copydays + Flickr1M image set with 10,000 visual words. (a) and (b) show rotated query images, (c) and (d) cropped.

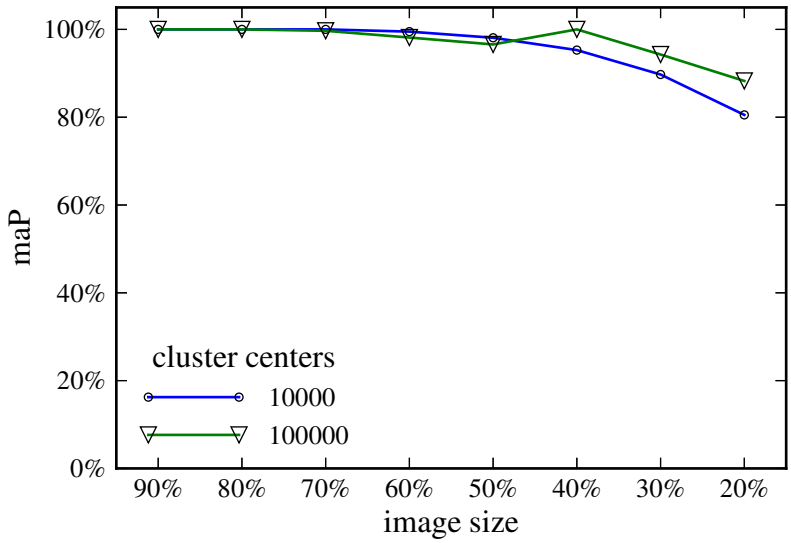


Figure 4.21: The Oxford Buildings Dataset + Flickr1M: The mean average precision (maP) for querying with cropped images. Configuration: ORB features, visual words clustered with kMeans_3.

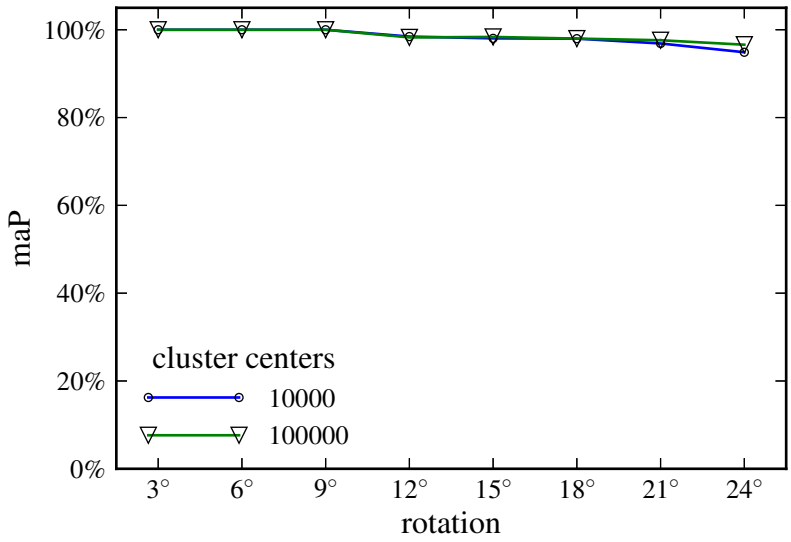


Figure 4.22: The Oxford Buildings Dataset + Flickr1M: The mean average precision (maP) for querying with rotated images. Configuration: ORB features, visual words clustered with kMeans_3.

occurrence of angular corners resulting in very specific local features. This means that the oxford images are to a certain degree visually different from the distractor images, while this is not the case for the Inria Copydays images. On the other side it can be argumented that the oxford images a more similar to each other for the same reason. It seams however that the first effect outweighs the second.

Furthermore the image set contains 55 images which represent specific buildings around Oxford. There are five images per motif and additionally bounding boxes are provided for images, separating foreground and background. These images were cropped according to the bounding boxes and used as query images as well. The mean average precision was calculated with the ground truth and software provided with the Oxford Buildings Dataset. The resulting precision over all 55 queries is $\approx 8.6\%$. Figure 4.23 shows a histogram over the precision of each query. Comparisons can be drawn to various state of the art papers,

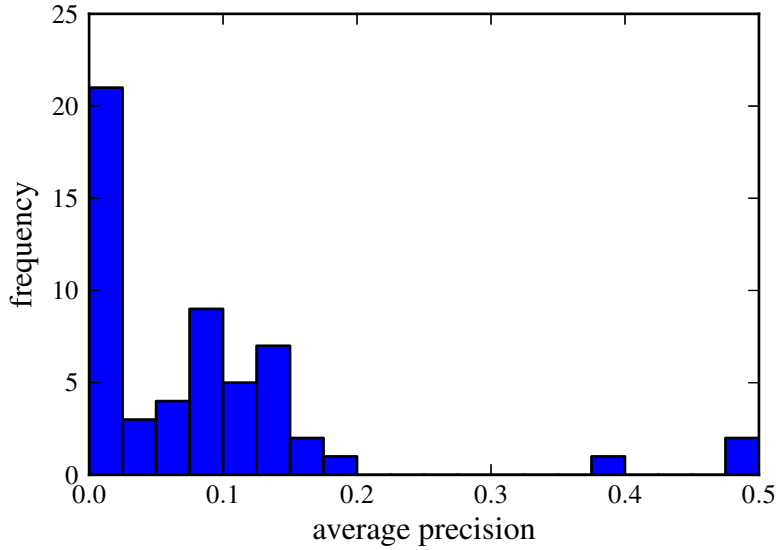


Figure 4.23: The Oxford Buildings Dataset: Histogram of the average precision for querying with the cropped 55 query images distributed with the Oxford Buildings Dataset. Configuration: ORB features, visual words clustered with kMeans_3.

e.g. [9, 72]. These methods are based on SIFT features and perform substantially better. All methods add about 1,000,000 distractor images to the Oxford Buildings Dataset. Although these distractor image sets are different a comparison can be made. Exact results are shown in table 4.6. To score a high precision in this challenge the underlying local features have

Table 4.6: A comparison of mean average precisions for the Oxford Buildings Dataset + 1M distractor images (BoW_{10k} are the results from this thesis with 10,000 visual words).

	#distractors	maP
BoW_{10k}	1M	8.6%
Constrained Keypoint Quantization [9]	1M	55.3%
Bag of Visual Words [72]	1M	41.3%
Geometry-preserving visual phrases [72]	1M	53.2%

to be robust against changes in perspective, different lighting conditions, etc. Under these circumstances SIFT features have a clear advantage over binary features like ORB. This is reflected by the results, as every referenced method outperforms the method presented in this thesis. However following the presented definition for near-duplicate image retrieval two shots of the same building from different viewing angles are not a near-duplicate, meaning that it is not required that one image is scored as similar to the other. Accordingly this "drawback" of binary features can be neglected.

4.5.3 APA-IT Press Images

The APA-IT press image set is used because it provides a real world application example. Press images are sold and published through various media channels like web pages, newspapers, etc. These published images can be transformed and republished without proper licensing. Near-duplicate image retrieval can identify these images and therefore detect copyright infringement. To simulate this use-case, images are transformed to query images in the same way as described for the Oxford Buildings Dataset (rotation and cropping). As consequence of the previous results only 10,000 visual words are clustered for this experiment. Figure 4.24 shows the results for cropping and figure 4.25 for rotation. Two sets of visual words are clustered kMeans_3 and kShifts $\text{linear}_{55\%}$. This provides a comparison of the two clustering algorithms for a large image set (about 1,000,000 images). As implied by the

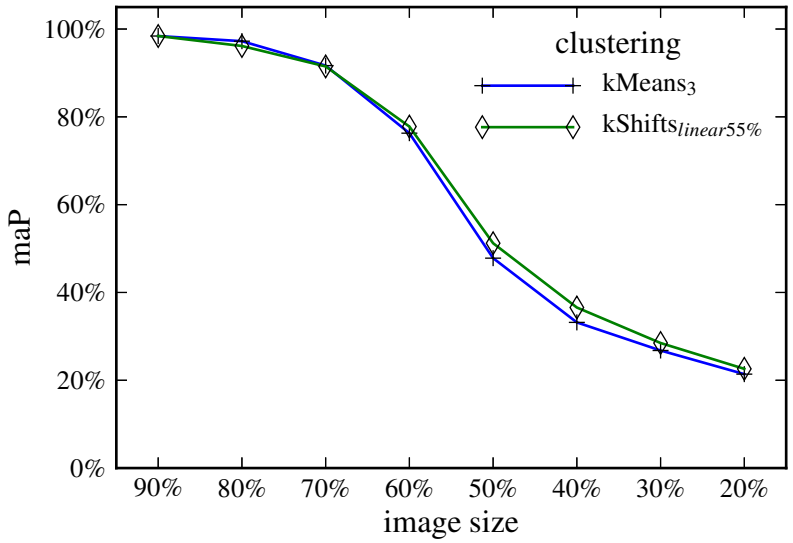


Figure 4.24: The APA-IT Press Images Dataset: The mean average precision (maP) for querying with cropped images. Configuration: ORB features, 10,000 visual words.

previous results, the precision is similar. Therefore the preferred algorithm is the one with shorter runtime. This however depends on the implementation and on the specific hardware (CPU), and no definite suggestion can be given here. Overall the results are worse than for the previous image sets which suggests, that the APA-IT images are more challenging.

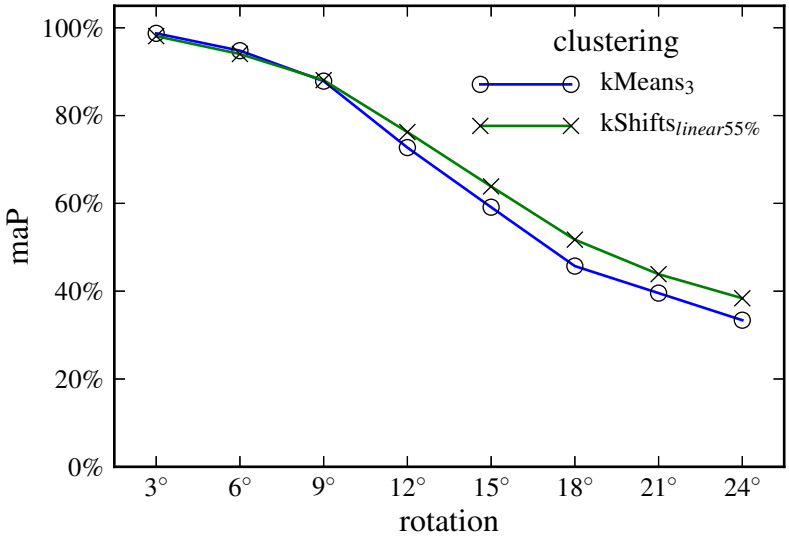


Figure 4.25: The APA-IT Press Images Dataset: The mean average precision (maP) for querying with rotated images. Configuration: ORB features, 10,000 visual words.

This could be explained by the fact that press images tend to have reoccurring motifs, e.g. portrais, and are therefore harder to distinguish.

For all previous results the dot product was used as similarity measure. As it corresponds to the cosine between two normalized bags of visual words its value range is $[0, 1]$ where 1 means identical and 0 totally different (orthogonal). The dot product can be calculated very efficiently. However other distance or similarity measures can be calculated even more efficiently (manhattan distance) or provide better results in specific situations. For comparison four different distances were used for the APA-IT press image set:

- Dot Product
- Manhattan
- Euclidean
- χ^2

The earth mover's distance was considered as well but the chosen implementation from [50], which is already optimized is considered computationally to expensive for this setup. Furthermore the implementation shows significant memory usage ($\approx$ 4 GB) in this case. This is despite the fact that the bags of visual words are represented by a sparse representation and the implementation supports this. However this can be explained by the increased dimensionality of the histograms in comparison to SIFT features (for which the implementation is intended). Figures 4.26 and 4.27 show the results. Dot product and euclidean distance lead to identical results, while the other two distances produce clearly worse results. As the dot product can be calculated with fewer CPU instructions it is clearly advantageous over the euclidean distance.

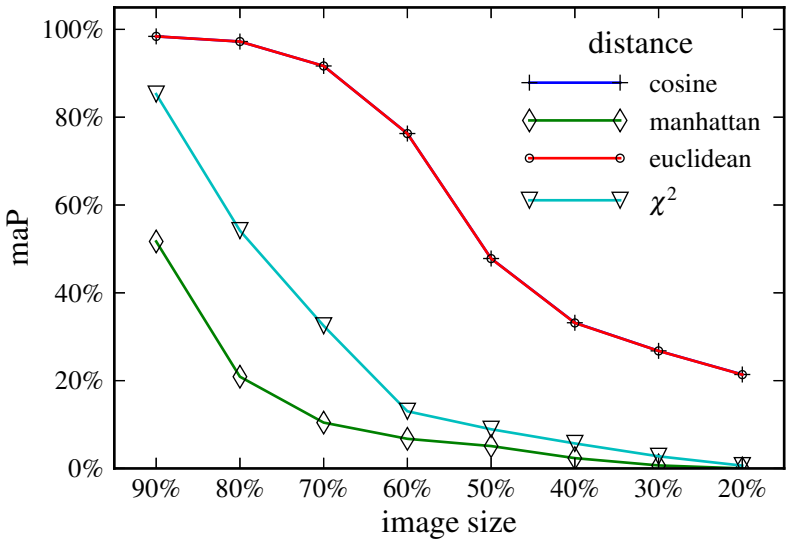


Figure 4.26: The APA-IT Press Images Dataset: The mean average precision (maP) for querying with cropped images. Configuration: ORB features, 10,000 visual words clustered with kMeans_3.

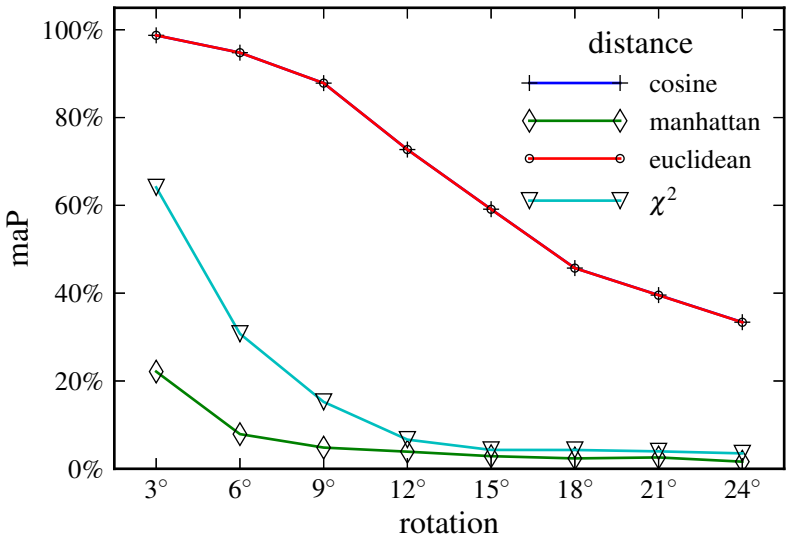


Figure 4.27: The APA-IT Press Images Dataset: The mean average precision (maP) for querying with rotated images. Configuration: ORB features, 10,000 visual words clustered with kMeans_3.

4.5.4 Ancient Coins

To evaluate if the implemented Bags of Visual Words method is applicable for the classification of ancient coins a comparison with [78] is presented. In this work Zambanini and Kampel [78] use an image set of ancient coins[3]. This image set contains 180 images. There are three images per class and thus 60 classes. Additionally the images are grouped into three groups so that each group has one image of every class:

- Group_1
- Group_2
- Group_3

The problem is to identify coin images which belong to the same class. Therefore a labeled database is used as ground truth, which in the context of this thesis are the "original" images. The unlabeled ("transformed") images are used as queries and it is assumed that the best matching image belongs to the same class. Each class and every possible combination of two classes act as ground truth once, which results in 540 queries. Table 4.7 shows the maPs, where the first row states the ground truth or "original" images and the first column the "transformed" query images. The results show that the implemented method is not appli-

Table 4.7: The maPs for the coins data-set used in [78]. The first row states the image groups in the database, the first column states the query image group.

	Group_1	Group_2	Group_3	$\text{Group}_{1\&2}$	$\text{Group}_{1\&3}$	$\text{Group}_{2\&3}$
Group_1		18.2%	17.0%			12.9%
Group_2	20.0%		22.6%		16.7%	
Group_3	18.4%	18.4%		17.4%		

cable for the classification of images of ancient coins, as the mean average precisions range only from about 13% to 23%. As seen in figure 3.5 coins from the same class have a diverse appearance. The method is too sensible in respect to the local differences in the coin structures and can not generalize the overall coin class appearances. In comparison the method of Zambanini and Kampel [78] achieves 71.7%, when using only one image as training sample per class.

[3] `http://www.caa.tuwien.ac.at/cvl/research/ilac/coindata`, accessed August 21, 2013

CHAPTER 5

Summary and Future Work

In this chapter a conclusion is given about the presented work. Furthermore the goals and final results are compared. Finally an outlook on possible future work is given.

In this thesis the state of the art in near-duplicate image retrieval and the methods it relies on were discussed. The state of the art concerning local visual feature detection and description was presented in detail. While the popular SIFT [43] method was introduced about one century ago it is still competitive and improvements for specific tasks have been proposed e.g. reducing the dimensions of a SIFT vector by applying PCA [36]. As run-time is one of the mayor drawbacks of SIFT various attempts were made to develop faster alternatives with similar or better performance. The most prominent and successful being SURF [5] until recently binary descriptors (BRIEF [10], ORB [62], BRISK [38] and FREAK [2]) appeared who may take the place of SURF. In the context of this thesis some of the strengths of SIFT and SURF, e.g. robustness against change of viewpoint and lighting conditions, were less important. On the other hand the significantly lower run-time of binary features was.

An overview on clustering algorithms, used for the generation of visual words, was given. While kMeans generates easy to handle centroids the EM-algorithm generates gaussian mixture models which describe data more accurately and can be used more directly for efficient image representations. For the chosen bags of visual words method only kMeans is suitable of these two and therefore was adapted for binary features. Additionally a second clustering algorithm called kShifts was adapted for binary features and a detailed comparison was given.

Finally the most important methods applicable in near-duplicate image retrieval were presented. Using local visual features directly as image representation leads to best results in terms of quality but comes with the drawback of higher computational costs and run-time. Even so for specific tasks it may be the only feasible solution. Alternative approaches compact the visual local features into a fixed size vector. More information is discarded to gain efficiency. These approaches are clearly superior as database size grows beyond a certain size. Various methods use SIFT for the extraction of local visual features but then reduce its dimensionality. This can be seen as a wasted effort in the feature extraction process. For this

thesis the bag of visual words method was chosen as it provides the best trade-off between efficiency and precision. Additionally it allowed a straight forward adaptation for binary features.

The goal of this thesis was to provide a framework for near-duplicate image retrieval with higher efficiency than current state of the art methods while still maintaining a competitive precision. As global visual features are not robust against specific image transformations, local visual features were chosen. To achieve a lower runtime binary features were chosen over SIFT features. This choice not only effected the runtime of the feature extraction process but also of the generation of the visual words and the image descriptions – the bags of visual words. Both benefited from the binary feature representation and its intrinsically lower memory footprint and faster distance calculations. The bags of visual words methods can be divided into four modules:

1. Extraction of local visual features.
2. Generation of visual words.
3. Generation of bags of visual words.
4. Actual image retrieval.

With the adaptation of binary features the efficiency of the first three modules was improved significantly, while the actual retrieval is untouched. This means several optimizations concerning the actual retrieval based on bags of visual words, which were not considered in this thesis, can be applied unchanged: e.g. inverted files [72], hashing [14] or nearest neighbour approximation [79]. The precision of the implemented system was compared to state of the art methods. The results showed that the method is restricted to specific challenges. The sensitivity of binary features in respect to noise render it unfit for jpeg compressed query images with notable artifacts. The Oxford Buildings and Coins image sets showed that the system can not cope with stronger visual changes like different viewing perspectives or coins with local differences in their 3d structure. On the other hand the presented system is competitive for the intended task of near-duplicate image retrieval. As has been shown it is robust against image transformations like cropping and rotation.

Finally various topics for future work come to mind besides the obvious, already mentioned, improvement by a more sophisticated retrieval system. So far only intensity information has been considered through the extraction of binary local features. Color information could further improve retrieval precision, either through additional features of by extending binary features, following published color extensions for SIFT. To additionally speed up the extraction of binary features a GPU version could be developed. Methods to approximate the vector quantization step for binary features with a precision-speed trade-off could be researched. Another possible improvement would be to use geometric feature matching (e.g. RANSAC [26]) on the top N images. This would even allow to process the more challenging

images discussed in this thesis with high precision and add only minor additional costs as the local features have already been extracted.

APPENDIX A

Source Code

```
1  void nearestNeighbour(__const __global ulong4* feature, __const __global ↩
       ↪ ulong4* center, __global uint* nearest)
2  {
3    const uint gid = get_global_id(0);
4    __private ulong4 v = feature[gid];
5
6    uint minDist = sizeof(ulong4) * 8;
7    uint minId = NCENTERS;
8    uint centerIdx = 0;
9
10   for (int i = 0; i < NCENTERS / CACHE_ELEMENTS; ++i)
11   {
12     __local ulong4 cache[CACHE_ELEMENTS];
13
14     if (get_local_id(0) == 0)
15     {
16       for (int j = 0; j < CACHE_ELEMENTS; ++j)
17       {
18         cache[j] = center[centerIdx++];
19       }
20     }
21
22     barrier(CLK_LOCAL_MEM_FENCE);
23
24     for (int j = 0; j < CACHE_ELEMENTS; ++j)
25     {
26       const ulong4 d4 = POPCOUNT(v ^ cache[j]);
27       const uint dist = d4.x + d4.y + d4.z + d4.w;
28
29       minId = (dist < minDist) ? (j + i * CACHE_ELEMENTS) : minId;
30       minDist = (dist < minDist) ? dist : minDist;
31     }
32   }
33
34   nearest[gid] = minId;
35 }
```

Figure A.1: OpenCL source code for finding nearest neighbour of binary features.

Bibliography

[1] M. Agrawal, K. Konolige, and M. Blas. Censure: Center surround extremas for realtime feature detection and matching. In D. Forsyth, P. Torr, and A. Zisserman, editors, *Computer Vision–ECCV (European Conference on Computer Vision)*, volume 5305 of *Lecture Notes in Computer Science*, pages 102–115. Springer, 2008.

[2] A. Alahi, R. Ortiz, and P. Vandergheynst. Freak: Fast retina keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 510–517, 2012.

[3] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.

[4] S. Arya and D. M. Mount. Algorithms for fast vector quantization. In *Data Compression Conference (DCC)*, pages 381–390, 1993.

[5] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. *Computer Vision–European Conference on Computer Vision (ECCV)*, 3951:404–417, 2006.

[6] R. M. Bolle, J. H. Connell, S. Pankanti, N. K. Ratha, and A. W. Senior. The relation between the roc curve and the cmc. In *IEEE Workshop on Automatic Identification Advanced Technologies (Auto ID)*, pages 15–20, 2005.

[7] A. Bosch, X. Muñoz, and R. Martí. Which is the best way to organize/classify images by content? *Image and Vision Computing*, 25(6):778–791, 2007.

[8] L. Bueno, E. Valle, and R. da S Torres. Bayesian approach for near-duplicate image detection. In *ACM International Conference on Multimedia Retrieval (ICMR)*, pages 15:1–15:8, 2012.

[9] Y. Cai, W. Tong, L. Yang, and A. G. Hauptmann. Constrained keypoint quantization: towards better bag-of-words model for large-scale multimedia retrieval. In *ACM International Conference on Multimedia Retrieval (ICMR)*, pages 16:1–16:8, 2012.

[10] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. *European Conference on Computer Vision (ECCV)*, 6314:778–792, 2010.

[11] Y. Cao, H. Zhang, Y. Gao, and J. Guo. An efficient duplicate image detection method based on affine-sift feature. In *IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT)*, pages 794–797, 2010.

[12] R. Cattell. Scalable sql and nosql data stores. *ACM SIGMOD Record*, 39(4):12–27, 2011.

[13] J. Chen, B. Feng, L. Zhu, P. Ding, and B. Xu. Effective near-duplicate image retrieval with image-specific visual phrase selection. In *IEEE International Conference on Image Processing (ICIP)*, pages 1909–1912, 2012.

[14] O. Chum and J. Matas. Fast computation of min-hash signatures for image collections. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3077–3084, 2012.

[15] O. Chum, J. Philbin, and A. Zisserman. Near duplicate image detection: min-hash and tf-idf weighting. In *British Machine Vision Conference*, volume 3, page 4, 2008.

[16] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[17] F. C. Crow. Summed-area tables for texture mapping. In *ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, pages 207–212, 1984.

[18] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, European Conference on Computer Vision (ECCV)*, pages 1–22, 2004.

[19] D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, PAMI-1(2):224–227, 1979.

[20] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *International Conference on Machine Learning (ICML)*, pages 233–240. ACM, 2006.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B*, 39:1–38, 1977.

[22] W. Dong, Z. Wang, M. Charikar, and K. Li. High-confidence near-duplicate image detection. In *ACM International Conference on Multimedia Retrieval (ICMR)*, pages 1:1–1:8, 2012.

[23] M. Douze, H. Jégou, H. Sandhawalia, L. Amsaleg, and C. Schmid. Evaluation of gist descriptors for web-scale image search. In *ACM International Conference on Image and Video Retrieval*, pages 19:1–19:8, 2009.

[24] J. C. Dunn†. Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4(1):95–104, 1974.

[25] V. Estivill-Castro. Why so many clustering algorithms: a position paper. *ACM SIGKDD Explorations Newsletter*, 4(1):65–75, 2002.

[26] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[27] Z. Harchaoui, M. Douze, M. Paulin, M. Dudik, and J. Malick. Large-scale image classification with trace-norm regularization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3386–3393, 2012.

[28] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, volume 15, page 50, 1988.

[29] F.-C. Huang, S.-Y. Huang, J.-W. Ker, and Y.-C. Chen. High-performance sift hardware accelerator for real-time image feature extraction. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(3):340–351, 2012.

[30] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591, 1959.

[31] R. A. Jarvis and E. A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, 100(11):1025–1034, 1973.

[32] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *European Conference on Computer Vision (ECCV)*, volume I of *LNCS*, pages 304–317. Springer, 2008.

[33] H. Jégou, F. Perronnin, M. Douze, C. Schmid, et al. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(9):1704–1716, 2012.

[34] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. An efficient k-means clustering algorithm: Analysis and implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24(7):881–892, 2002.

[35] T. Kato. Database architecture for content-based image retrieval. In *SPIE Image Storage and Retrieval Systems*, pages 112–123, 1992.

[36] Y. Ke and R. Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–506, 2004.

[37] Y. Ke, R. Sukthankar, and L. Huston. An efficient parts-based near-duplicate and sub-image retrieval system. In *ACM International Conference on Multimedia*, pages 869–876, 2004.

[38] S. Leutenegger, M. Chli, and R. Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2548–2555, 2011.

[39] D. Li, L. Yang, X.-S. Hua, and H.-J. Zhang. Large-scale robust visual codebook construction. In *ACM International Conference on Multimedia*, pages 1183–1186, 2010.

[40] X. Liao, Y. Wang, L. Ding, and J. Gu. A novel duplicate images detection method based on plsa model. In *SPIE International Conference on Machine Vision (ICMV)*, volume 8349, pages 834909–834909–8, 2012.

[41] H. Ling, L. Yan, F. Zou, C. Liu, and H. Feng. Fast image copy detection approach based on local fingerprint defined visual words. *ACM Signal Processing*, 93:2328–2338, 2012.

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

[43] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004.

[44] R. McGill, J. W. Tukey, and W. A. Larsen. Variations of box plots. *The American Statistician*, 32(1):12–16, 1978.

[45] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 525–531, 2001.

[46] O. Miksik and K. Mikolajczyk. Evaluation of local detectors and descriptors for fast feature matching. In *IEEE International Conference on Pattern Recognition (ICPR)*, pages 2681–2684, 2012.

[47] H. Moon and P. J. Phillips. Computational and performance aspects of pca-based face-recognition algorithms. *Perception-London*, 30(3):303–322, 2001.

[48] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, pages 331–340, 2009.

[49] O. Pele and M. Werman. A linear time histogram metric for improved sift matching. In *Computer Vision–ECCV (European Conference on Computer Vision)*, pages 495–508, 2008.

[50] O. Pele and M. Werman. Fast and robust earth mover's distances. In *IEEE International Conference on Computer Vision (ICCV)*, pages 460–467, 2009.

[51] O. Pele and M. Werman. The quadratic-chi histogram distance family. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision–ECCV (European Conference on Computer Vision)*, volume 6312 of *Lecture Notes in Computer Science*, pages 749–762. Springer, 2010.

[52] D. Pelleg, A. Moore, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *International Conference on Machine Learning (ICML)*, volume 1, pages 727–734, 2000.

[53] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[54] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007.

[55] T. Pönitz and J. Stöttinger. Efficient and robust near-duplicate detection in large and growing image data-sets. In *ACM International Conference on Multimedia*, pages 1517–1518, 2010.

[56] T. Pönitz, R. Donner, J. Stöttinger, and A. Hanbury. Efficient and distinct large scale bags of words. In *Workshop of the Austrian Association for Pattern Recognition (AAPR)*, pages 139–146, 2010.

[57] H. Pruscha. *Statistisches Methodenbuch: Verfahren, Fallstudien, Programmcodes*. Springer, 2005.

[58] D. Reynolds. Gaussian mixture models. In *Encyclopedia of Biometrics*, pages 659–663. Springer, 2009.

[59] S. Romberg, M. August, C. X. Ries, and R. Lienhart. Robust feature bundling. In *Advances in Multimedia Information Processing – PCM (Pacific-Rim Conference on Multimedia)*, pages 45–56. Springer, 2012.

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

[61] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. *Computer Vision–European Conference on Computer Vision (ECCV)*, 3951:430–443, 2006.

[62] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571, 2011.

[63] Y. Rubner, C. Tomasi, and L. Guibas. A metric for distributions with applications to image databases. In *IEEE International Conference on Computer Vision (ICCV)*, pages 59–66, 1998.

[64] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[65] S. M. Smith and J. M. Brady. Susan—a new approach to low level image processing. *International Journal of Computer Vision (IJCV)*, 23(1):45–78, 1997.

[66] J. Stöttinger, B. T. Goras, T. Pönitz, N. Sebe, A. Hanbury, and T. Gevers. Systematic evaluation of spatio-temporal features on comparative video challenges. In *Computer Vision–Asian Conference on Computer Vision (ACCV) Workshops*, pages 1–8, 2010.

[67] B. Stroustrup. A history of c++: 1979–1991. In *History of programming languages—II*, pages 699–769. ACM, 1996.

[68] W. Tong, F. Li, R. Jin, and A. Jain. Large-scale near-duplicate image retrieval by kernel density estimation. *International Journal of Multimedia Information Retrieval*, 1:1–14, 2012.

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

[70] S. Vitaladevuni, F. Choi, R. Prasad, and P. Natarajan. Detecting near-duplicate document images using interest point matching. In *IEEE International Conference on Pattern Recognition (ICPR)*, pages 347–350, 2012.

[71] E. M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5):697–716, 2000.

[72] F. Wang, S. Zhang, H. Li, and N. Zhang. Image retrieval using multiple orders of geometry-preserving visual phrases. In *IEEE International Conference on Image Analysis and Signal Processing (IASP)*, pages 1–5, 2012.

[73] Z. Wang and K. Hong. A new method for robust object tracking system based on scale invariant feature transform and camshift. In *ACM Research in Applied Computation Symposium*, pages 132–136, 2012.

[74] B. L. Welch. The generalization of ”student's“ problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

[75] H. Xie, K. Gao, Y. Zhang, J. Li, Y. Liu, and H. Ren. Effective and efficient image copy detection based on gpu. In *Trends and Topics in Computer Vision, ECCV Workshops (European Conference on Computer Vision)*, pages 338–349. Springer, 2012.

[76] D. Xu, T.-J. Cham, S. Yan, L. Duan, and S.-F. Chang. Near duplicate identification with spatially aligned pyramid matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(8):1068–1079, 2010.

[77] T. Yamazaki, T. Fujikawa, and J. Katto. Improving the performance of sift using bilateral filter and its application to generic object recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 945–948, 2012.

[78] S. Zambanini and M. Kampel. Coarse-to-fine correspondence search for classifying ancient coins. In *Computer Vision–ACCV Workshops (Asian Conference on Computer Vision)*, pages 25–36, 2013.

[79] J. Zepeda, E. Kijak, and C. Guillemot. Approximate nearest neighbors using sparse representations. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2370–2373, 2010.

[80] D.-Q. Zhang and S.-F. Chang. Detecting image near-duplicate by stochastic attributed relational graph matching with learning. In *ACM International Conference on Multimedia*, pages 877–884, 2004.

[81] L. Zheng, Y. Lei, G. Qiu, and J. Huang. Near-duplicate image detection in a visually salient riemannian space. *IEEE Transactions on Information Forensics and Security*, 7 (5):1578–1593, 2012.

[82] D. W. Zimmerman. Teacher's corner: A note on interpretation of the paired-samples t test. *Journal of Educational and Behavioral Statistics*, 22(3):349–360, 1997.