Internal Report 2013–number July 2013

Geometry-preserving algorithms for Content-based Image Retrieval

Frank Anemaet

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Geometry-preserving algorithms for content-based image retrieval

Frank Anemaet

July 6, 2013

Abstract

Content-based image retrieval algorithms are frequently based on the bag-of-visual-words (BoV) representation of images. The BoV representation discards all spatial information. In this paper, we propose an algorithm which encodes spatial information into the BoV representation. Our algorithm forms geometry-preserving visual phrases which are invariant to translation and rotation. Furthermore, we demonstrate that our algorithm outperforms other recently proposed geometry-preserving image retrieval algorithms in terms of accuracy.

Keywords: algorithms, content based image retrieval, bag of visual words, computer vision

Acknowledgments

All the research work presented in this thesis has been conducted in the Leiden Institute of Advanced Computer Science (LIACS) at Leiden University. I would like to thank my supervisor Dr. Michael S. Lew for his guidance and support during the master thesis work. Next, I would like to thank my family.

Frank Anemaet
July, 2013

Contents

Glossary

BOW	Bag of words, algorithm for large scale text retrieval.
BOV	Bag of visual words, the most popular algorithm for large scale image retrieval.
CBIR	Search engine designed to search an image database by an image query.
C	a general-purpose programming language initially developed by Dennis Ritchie between 1969 and 1973.
C++	a general purpose programming language with a bias towards systems programming that supports data-abstraction, object-oriented programming and generic programming.
CLI	Command Line Interface (CLI) is a text-based interface that is used to operate software and operating systems.
GUI	Graphical User Interface.
GVP	Geometry- Preserving Visual Phrases. An image retrieval approach that encodes more spatial information through the geometry-preserving visual phrases.
HTML	HyperText Markup Language (HTML) is the main markup language for creating web pages and other information that can be displayed in a web browser.
IDE	An integrated development environment (IDE) is a programming environment that has been packaged as an application program, typically consisting of a code editor, a compiler, a debugger, and a graphical user interface (GUI) builder.

IR	Information Retrieval, is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).
HTML	HyperText Markup Language (HTML) is the main markup language for creating web pages and other information that can be displayed in a web browser.
Java	a high-level programming language.
MSVC	Microsoft Visual C++, an integrated development environment (IDE) product from Microsoft for the C, C++, and C++/CLI programming languages.
SPD	Spatial Pyramid Matching. A computationally efficient extension of an orderless bag-of-features image representation.
Python	Python is an interactive, object-oriented, extensible programming language.
SIFT	Scale Invariant Feature Transform. A method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination.
SURF	Speeded Up Robust features. A novel scale- and rotation-invariant interest point detector and descriptor.
TAG	A label assigned to identify data in memory.
TF-IDF	a weighting scheme consisting of two parts: Term Frequency (TF) and Inverse Document Frequency (IDF). TF is the frequency of the term t occurring in document d. IDF is calculated by taking the log of the number of documents D divided by the number of documents the term T is in.
ORB	oriented BRIEF. A keypoint detector and descriptor extractor.

OpenCV	Short for Open Source Computer Vision, a library of programming functions for real time computer vision. It has C++, C, Python and soon Java interfaces running on Windows, Linux, Android and Mac. The library has more than 2500 optimized algorithms.
QBIC	Query by image content. A CBIR system. QBIC allows queries on large image and video databases based on example images, user-constructed sketches and drawings, selected color and texture patterns, camera and object motion, and other graphical information.
WWW	The World Wide Web is the universe of network-accessible information, an embodiment of human knowledge.

2

List of Figures

List of Tables

1. Introduction

1.1 Content-based Image Retrieval

Large data archives are increasingly common, they have grown far beyond the scale of what we can manually order. The number of data we can store is also increasing rapidly. In the 1980s, a harddrive with a capacity of several megabytes (2^{20} bytes) was considered big. Today, even a personal computer may have a one terabyte (2^{40} bytes) disk or more. A datacenter - at present - may hold several zettabytes (2^{70} bytes). Given a system with the capacity to store a large amount of data, these large data archives would be useless without computational tools such a search engine. Text, in a natural language, has been the most prevalent way to search, categorize and organise large data archives. A text-based search engine may be an effective tool for finding documents, but it may not be efficient for all types of data. For example, it is hard to describe what we perceive in such a way that another person has exactly the same image in their head. This is sometimes the case if a person visits a cinema after having read a book, and claims to have imagined things much differently. Indeed, the natural language may leave space for imagination when communicating a certain scene. The scene an individual may imagine depends on the degree that the words of a natural language correspond to the perception a sender communicates.

There is a rapid growth of image databases, which could be of significant importance to users. To use such a large image database, it is mandatory to have computational tools. Such computational tools could be used in medical diagnosis and crime prevention. In these cases, an accurate interpretation by the machine would be beneficial. For example, what use would a crime prevention system have, if it can not find the suspect in a data archive.

Search engines as seen today, such as Google™and Yahoo®, frequently rely on a verbal description in a natural language. One may question to which degree verbal descriptions, also known as user annotated tags, describe the content of the image. The transformation of an image into a natural language may leave out significant information to a user, because a user annotated tag may describe only so much as a natural language permits it to. There are other reasons why a verbal description may not be ideal: Image annotation can be a time consuming task - depending on the number of images to be annotated. Users may lack the time or interest to annotate each individual image and choose to simply annotate the set of images or to not annotate at all. Tags could simply be incorrect and not represent the visual information in any way, they may simply not capture all visual content in an image

and tags may be language specific.
To search image effectively, image search engines require a different approach - perhaps not relying on user annotation - or - verbal descriptions at all. One method is to use the visual content that is present in an image as query for an image search engine - or - in other words, to search by image. In this case, the user would need to have a similar image on their machine - or - to draw the visual content. An image search engine capable of searching by visual content is known as a Content Based Image Retrieval system (CBIR). Given an image, a CBIR system should find visually similar images in a large image database. A variety of studies on CBIR systems has been done [Veltkamp 2000].
In the next paragraph we will describe the main problem in this work. Then existing CBIR systems and the current research challenges in the field of CBIR will be discussed. We will then discuss the purpose of this work, followed by the thesis content and finally we will discuss the contributions of this work.

1.2 Problem description

The state-of-the-art content based image retrieval systems created in the last decade are frequently based on the bag-of-visual-words (BoV) model. Despite its popularity the BoV model discards all spatial information that is available in the images. If an algorithm could include the spatial information, it may increase the precision of an CBIR system. Recent works propose extensions to the BoV model to improve the accuracy of CBIR systems. These include Spatial Pyramid Matching (SPM) [Lazebnik 2006] and Geometry- Preserving Visual Phrases (GVP) [Zhang 2011]. Nevertheless, the SPM algorithm encodes spatial information very strictly, thus it is not invariant to rotation, translation or scale. GVP on the other hand, is only invariant to translation. Thus, none of these algorithms fully capture the spatial information that is lost in the original BoV model.

1.3 Existing CBIR systems

In the last decades several CBIR systems have been created. These CBIR systems started to appear in the early nineties. Some of these systems are used commercially - even though the problem of robust computer vision is still largely unsolved. We will briefly discuss some CBIR systems which have been created in the last decades. A more detailed list of CBIR systems may be found in the references [Veltkamp 2000].

QBIC (Query-By-Image-Content) was one of the first content based image retrieval systems developed in 1995 by IBM Almaden Research Center (USA). In the QBIC system, search by image is used as supplement rather than replacement of the traditional text-based image search engine. It is used in two of their commercial products: DB2, a data-management product and Ultimedia Manager. The QBIC system relies on simple low-level features such as color and texture. To be more specific, an image is described by a vector of average color coordinates and a color histogram. Texture is described by three features: coarseness, contrast, and direction.

VisualSEEK is another CBIR system that has been developed in the early years, it first appeard in 1996. The system was created by Smith et al at Colombia University. The system had a client-server architecture and ran on the World Wide Web. The authors used Netscape Navigator to interact with the system on client side. Results could be thumbnails, images or video. A user could query the system using its query tools and text annotation tools. A spatial color-based approach was available to query the system. The system had a grid interface at which a user could sketch regions and position them on the grid. In addition, the user had the ability to assign properties such as color, size and absolute location.

ImageScape was developed in 1999, at the end of the early years. The ImageScape image retrieval used various methods to query an image database. The user of this system could query an image database by sketching and by image icons (visual concepts). That is, given a user-drawn sketch the system would find visually similar images.

Anaktisi is a CBIR system which appeard in 2010. It was created at the University of Thrace by Zagoris et al. The system has been built using .NET, C and has a Web interface. A user may query the system using an example image, image sketches, image sketches of other users and keywords. The system is based on color and texture features.

Google Image Search, allows to query by text and image. Querying by image has only been available since 2013 and the implementation is proprietary. Thus, in this system a user can not query by sketch. Nevertheless, various properties of the images to return may be selected such as the size, colors in the image and the type of image (face, photo, clip-art, line drawing or animated).

In CBIR new research areas have emerged, take, for example the automated tagging of images. Such an algorithm would find concepts within images, and thus automatically describe images based on their visual content and the problem is then reduced to text-based searching. In this chapter we will describe some of the CBIR research areas (including challenges) in more detail.

1.4 Research areas in CBIR

CBIR is an active research area, just in the last year there have been more than 400 related publications. The area may be split into several research areas, of which many are directly related to the computer vision area. This paragraph describes some of the research areas in the field of Content Based Image Retrieval (CBIR).

Semantic gap

Low-level features such as texture, shape or salient points are commonly used in computer vision based systems such as a content-based image retrieval systems. At the time of writing, we as users of such systems have the ability to think and describe in much higher conceptual levels than machines. There is no theory or strict rules to overcome this problem. This results in a gap between high-level features and low-level features and is often referred to as the 'semantic gap' in the CBIR community.
A user of a CBIR system expects to retrieve visually similar images given a query however if we are far from bridging this gap it will not return visually similar images unless the data-set range is somewhat narrow. Of course, we could question if a user wishes visually similar results or conceptually similar results. Nevertheless, in either way the semantic gap has to be reduced if we wish to have accurate CBIR systems on large data collections.

Image segmentation

Image segmentation is a fundamental problem in computer vision. In this paragraph we will briefly discuss the image segmentation problem and some of the current techniques that have been created. The goal of image segmentation is to cluster pixels into salient image regions. A salient image region is an individual surface, object, or natural parts of an object. To simplify, image segmentation is the division of an image into several regions. These regions may well be the objects. Indeed, a pratical application of an image segmentation algorithms would be to seperate all the objects in an image. Every individual pixel in an image belongs to one more regions. If a system could seperate an object from a background using image segmation correctly, it is likely to influence the accuracy of a CBIR system and computer vision systems in general. There are many other practical applications of image segmentation including medical applications, locating objects in satellite images, Face Recognition, Fingerprint Recognition etc. A variety of image segmentation techniques have been created over the last decade, but perfect image segmentation can generally not be achieved due to oversegmentation and undersegmentation. In the case of oversegmentations, pixels are included into a segment that belong to another segment. In the case of undersegmentation, pixels belonging to its segment are included in other segments. Image segmentation techniques include: thresholding a histogram and slicing (Threshold based segmentation), detecting edges in image which are assumed to be edges of regions (Edge based segmentation), techniques at which an algorithm tries to find the middle of a region and expand from there (Region based segmentation). The threshold based segmentation technique may be useful if we have bright non-overlapping objects and a dark background, however this may only function well under those exact conditions. The thresholding operation is a grey value remapping operation g defined by:

$$g(v) = \begin{cases} 0 & \text{if} \quad v < t \\ 1 & \text{if} \quad v \geqslant t, \end{cases}$$

where v represents a grey value, and t is the threshold value. Given a gray valued image, this will map the image into two segments, as shown by the binary output. If we wish to have more segments, we may simply increase the number of threshold values, as shown in the formula below:

$$g(v) = \begin{cases} 0 & \text{if} \quad v < t_1 \\ 1 & \text{if} \quad t_1 \leqslant v < t_2 \\ 2 & \text{if} \quad t_2 \leqslant v < t_3 \\ \vdots & \vdots \quad \vdots \\ n & \text{if} \quad t_n \leqslant v. \end{cases}$$

As mentioned earlier, another technique for image segmentation is known as edge-based segmentation. This is generally achieved using the following stages: compute an edge image (for example by applying an convolution matrix to the original image), process the resulting edge image in such a way that closed object boundaries remain and transform this result to an segmented image by filling in the object boundaries. The final image segmentation technique we will briefly discuss is Region based segmentation. There are two common operations in region based image segmentation: splitting and merging. Region based segmentation based on merging has these stages: (1) obtain an oversegmentation of an image, (2) merge similar segments and (3) goto step 2 until no segments can be merged. Region based segmentation based on splitting has these stages: (1) obtain an initial segmentation of the image, (2) split each segment that is inhomogeneous and (3) go to step 2 until all segments are homogeneous.

Curse of high-dimensionality

Ideally a content based image retrieval system finds visually similar images given a query image. One way to achieve a higher accuracy in a computer vision based system would be to simply increase the number of features. This would result in high-dimensional vectors in a high-dimensional vector space. This would consume much more memory and computation time and the question arises as to what is the use of the distances between these high-dimensional vectors. This is often referred to as the curse of (high-)dimensionality, a term introduced by mathematician Richard Bellman.

Searching with relevance feedback

If a search engine is queried by a user, ambiguities may arise. Even in the text-based search engines this is prevalent, for example, a single word in a natural language may have two meanings. There is no way to determine which meaning the user wishes to search for without involving the user, in other words to ask feedback. If we search based on visual content of an image, it is certain that this ambiguities will arise. Interpreting images is a complex task, and may be more ambigious than interpreting text. Relevance feedback in CBIR has typically the following stages [Zhou 2003]: (1) the system provides initial results, (2) user provides a judgement as to which images are relevant to the query and (3) the system learns and tries again. Zhou mentions there are typically has the following problems: (1) small sample size. The number of images as user marks as relevant and irrelevant is typically very small in a round in comparison an image database of several thousands of images or even more. (2) Asymmetry in training sample. The desired output of information retrieval is not necessarily a binary decision on each point as given by a classifier, but rather a rank-ordered top-k returns and (3) Real time requirement. The algorithm should be fast enough to return results in a reasonable amount of time. A variety of relevance feedback algorithms have been created, based on different assumptions. Indeed, it is questionable wheter a user may wish to give binary feedback (select positive/negative examples), select only positive examples or perhaps give feedback to the system in another way.

Future CBIR challenges

There are many challenges to overcome if we wish to have more accurate CBIR systems. These include:

- Concept detection in the presence of complex backgrounds
- Multi-modal analysis and retrieval
- Experiential multimedia exploration
- Interactive search
- Performance evaluation

1.5 Thesis contributions

The contributions of this work are:

- A CBIR system
- Implementation and evaluation of state-of -the-art image retrieval algorithms
- An new algorithm for image retrieval

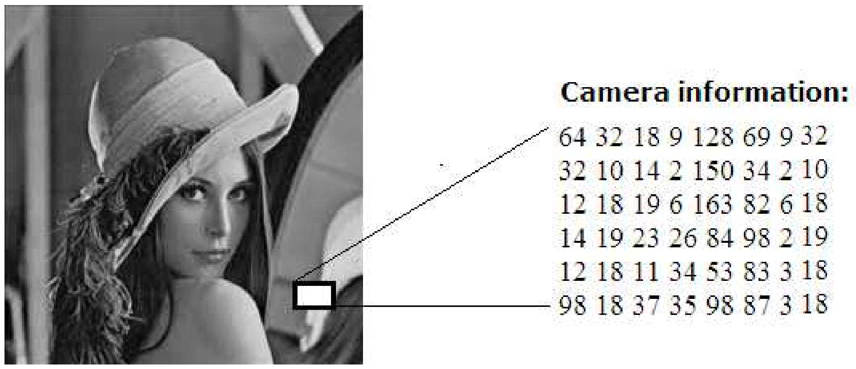
1.6 Thesis organisation

This research has focused on image retrieval. In Chapter 2 we will discuss related work. It contains an overview of the state of the art image retrieval algorithms such as bag of visual words (BoV), geometry-preserving visual phrases (GVP) and spatial pyramid matching (SPD).
In Chapter 3 we will discuss our approach including implementation details.
In Chapter 4 we will evaluate state of the art image retrieval algorithms.
In Chapter 5 and 6 we will illustrate the results and conclude this work.

2. Background

2.1 Introduction to Computer Vision

Generally we assume that perceiving is not a complex task, as we do it each day with the least of effort. It may be hard to imagine that millions of atoms inside us are directly responsible for our perception of the universe. We may recognize objects in various conditions, albeit rotated, translated, scaled or deformed.
In 1966, a professor at MIT university named Minsky, gave an assignment to a student (Sussman) to make the computer describe what it perceived given an connected camera. The project was not ment as a joke, Minsky and Sussman expected the project to succeed within the given time frame. Since then, there are many decades of research into a field called computer vision. The field is not to be confused with image processing (transforming an image as a whole), in computer vision assignment described above the computer had to describe what it sees - in other words - the goal of computer vision is to create software that can interpret digital images.
In computer vision, the machine has nothing more than a grid of numbers or a matrix. This matrix is derived from a certain input device, such as a camera. Given such a matrix, these algorithms are often designed to reason about the visual information that is encoded within this matrix, as shown in figure 2.1.



```
64 32 18 9 128 69 9 32
32 10 14 2 150 34 2 10
12 18 19 6 163 82 6 18
14 19 23 26 84 98 2 19
12 18 11 34 53 83 3 18
98 18 37 35 98 87 3 18
```

Figure 2.1: The data a computer has, given an image

This reasoning may be a decision such as "there is a face in the image" or a new representation. To make such a decision, a computer requires to have a large database containing images of faces and non-faces. An computer vision algorithm needs to be trained with this data. This trained computer vision algorithm may then predict whether a face is present in an image. Even with thousands of training images, such an algorithm may not be 100% precise.

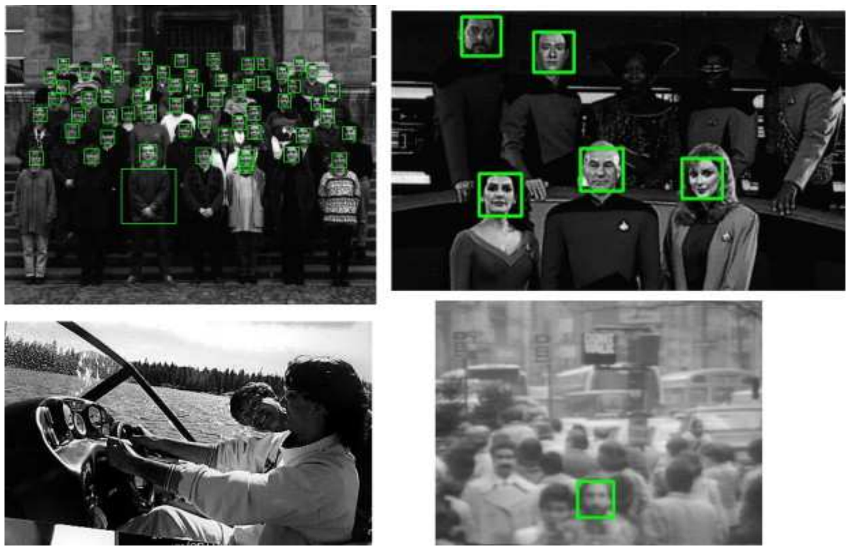


Figure 2.2: False detection of faces, Pan 2013

We could describe figure 2.1 in detail, which to us is nothing more than an ordinary task: "There is a white woman in the picture wearing a hat, the woman has long hair and is looking towards the camera. The woman is probably standing and looking over her shoulder. The woman likely has brown or black hair". It is very challenging - at present - to derive such a description from a computer. A single object - which is present in an image - may translate, change in rotation, illumination, scale or even deform. The description could go to a higher-level: "The woman in the image is Lena Söderberg, a Swedish model and the image dates back to 1972. The image is used in many image processing algorithms and scientific publications.". Of course, to derrive such a description of images from a computer, if at all possible at present, is a complex task. Images may contain even more complicated scenes, in which there are multiple objects cooperating according to certain rules. For example, a game of soccer or a highway.

Formally, we could define computer vision as ̈the science of creating similar capability in computers and, if possible, improve upon it ̈. [Demaagd 2012]. Algorithms for specific tasks have been developed in the last decades such as searching, classification, tracking, detection and recognition. Initially, these algorithms were based on simple low-level features such

as color or shape. However, this is only very limited understanding - and more complex features, such as salient-point based features were created in the decades. One of the most frequently mentioned feature in computer vision is the Scale-Invariant Feature Transform (SIFT) [Lowe 2004]. Informally, the SIFT algorithm finds distinct salient points within an image and describes them. These salient points are to some degree invariant to scale, illumination, rotation and translation. Increasing the number of features in a computer vision algorithm may increase the precision of such a system. This however comes at a cost: the memory and processor usage will increase - and the extra computation time may not count up to the amount of precision gained.
The field of computer vision is very broad, it is multidisciplinary: its related to the fields of mathematics, physics, biology, engineering and computer science. Many fields within computer science are related to computer vision including machine learning, signal processing, robotics and artificial intelligence.
A research area within the field are search engines. The user may query a search engine in various ways such as by text, sketching or by an example. As stated earlier, user annotated descriptions may not always be reliable. If we wish for a computer to comprehend the visual data for a high search accuracy, we require algorithms that can understand the data to some extend. Systems that search an image database based on the visual content of a query image are known as content based image retrieval (CBIR) systems.

2.2 Bag of words model

Modeling data, for example modeling text-documents, is required for a large number of computational tasks. These include the classification, segmentation, visualization and retrieval of data. Despite the popularity of internet search engines - or information retrieval systems - today, natural language processing is still an open problem. One models for information retrieval is the popular bag-of-words model (BoW), which has its origins in text retrieval.
The key idea is that each document is summarized by the count of its words:

x = {x1, ..., xd} .

Where x is the vector representation of a text document as defined by the BoV model. The xs can be represented as the frequency, weights or simply the presence/absence. This vector representation of documents loses all structure which might have been present in a text document before. For this particular reason, the model is sometimes referred to as an order-less document representation. Even though the structure is lost, it has been proven to be an effective model for classification and information retrieval tasks.
Text document are represented as a vector holding the frequency of each word. The ith position of a vector defined by the BoV model is the frequency of the ith word in that document. To clarify, it contains the frequency of each possible word in a dictionary. The dictionary contains all unique words which are present in all documents. The dictionary is

Table 2.1: Example message table

#	Message
1	the quick brown fox
2	the quick rabbit ran and ran
3	the fox ran
4	ran ran ran ran

Table 2.2: BOV vectors for messages

#	X1	X2	X3	X4	X5	X6	X7	X8
#	and	at	brown	fox	quick	rabbit	ran	the
1	0	0	1	1	1	0	1	1
2	1	0	0	0	1	1	2	1
3	0	0	0	1	0	0	1	1
4	0	0	0	0	0	0	4	0

sometimes referred to as code-book - and - the words referred to as code-words. The length of a vector defined by BoV is the number of words in the dictionary. As many unique words may appear in the collection of text documents, we may get very long vectors. The same dictionary will be used for all vectors, and thus the creation of a dictionary is an essential step in the BoV model. The presence and absence in text documents varies - and for this reason such a vector can be very sparse.

To clarify the BoW we will make a small example of the BoV model on four text messages:
As shown above, each text document is represented by a vector. The set of all words is known as the dictionary or code-book. In this example, the size of the code-book is eight. If we would use real documents, such as scientific articles the code-book (and thus length of the vector) could become very long. In this particular example, it appears that document four is the least similar to the other text documents. These vectors may be represented as binary vectors, instead of based on word frequency. This would result in these binary vectors:

Once the documents are represented as vectors in a vector space, various computational tasks can be done. For example, the vectors may be clustered using a clustering algorithm, such as k-means where k is the number of clusters. Clustering may be useful for various tasks, such as grouping documents by content. Another computational task is information retrieval (IR), which is one of our main interests in this work. If we consider the vectors our text-document database - we may extract the BoV representation of a query document and find the nearest BoV vectors in the vector space to our query vector. The nearest neighbors

Table 2.3: BOV binary vectors for messages

#	X1	X2	X3	X4	X5	X6	X7	X8
#	and	at	brown	fox	quick	rabbit	ran	the
1	0	0	1	1	1	0	1	1
2	1	0	0	0	1	1	1	1
3	0	0	0	1	0	0	1	1
4	0	0	0	0	0	0	1	0

will be the most related documents. Different query vectors defined by the BoV model will be at different locations in the vector space. And thus, the BoV model can be used as a retrieval system. The extraction of the BoV vectors for the entire task may be a very computationally expensive task - depending on the number of documents in the dataset. Nevertheless, this model has been used with millions of documents and thus is quite scale-able.

2.3 Bag of visual words model

Inspired by the success of text-retrieval models, scientists have tried to mimic the BoW model for image, known as the bag-of-visual-words (BoV). In this representation an image is considered as a document with visual words. Like the BoW model, a code-book is formed for all possible words - but in contrast to the BoW model these words are 'visual words'. Each image will be represented as a vector defined by visual words. Of course, this may be somewhat counter-intuitive - what is a visual word?
The process of forming visual words for the BoV model is shown in Figure 2.1. In the BoV model, the first step in identifying visual words is to find a set of key points (also referred to as salient points) (a). These key points are distinct areas of an image - not all areas of the image are useful or descriptive of the content. A wide variety of key point detection algorithms have been created such as the SIFT key-point detection algorithm. The second step is to cluster the detected key points to form distinct visual words (b). A visual word is then the center of a cluster. Given a new image (c) we can find the nearest visual words. Similar to the text-based BoV model, we may represent images as vectors or histograms defined by BoV (d).

Of course, visual words will be much more ambiguous than words from the natural languages. In addition, it is questionable how large such a visual vocabulary or dictionary should be. Visual words in the BoV model may not be as representative as we describe visual objects that we can name in the real world. In addition to this, all spatial information of the image is lost - similar to the loss of text document structure in the BoW model.

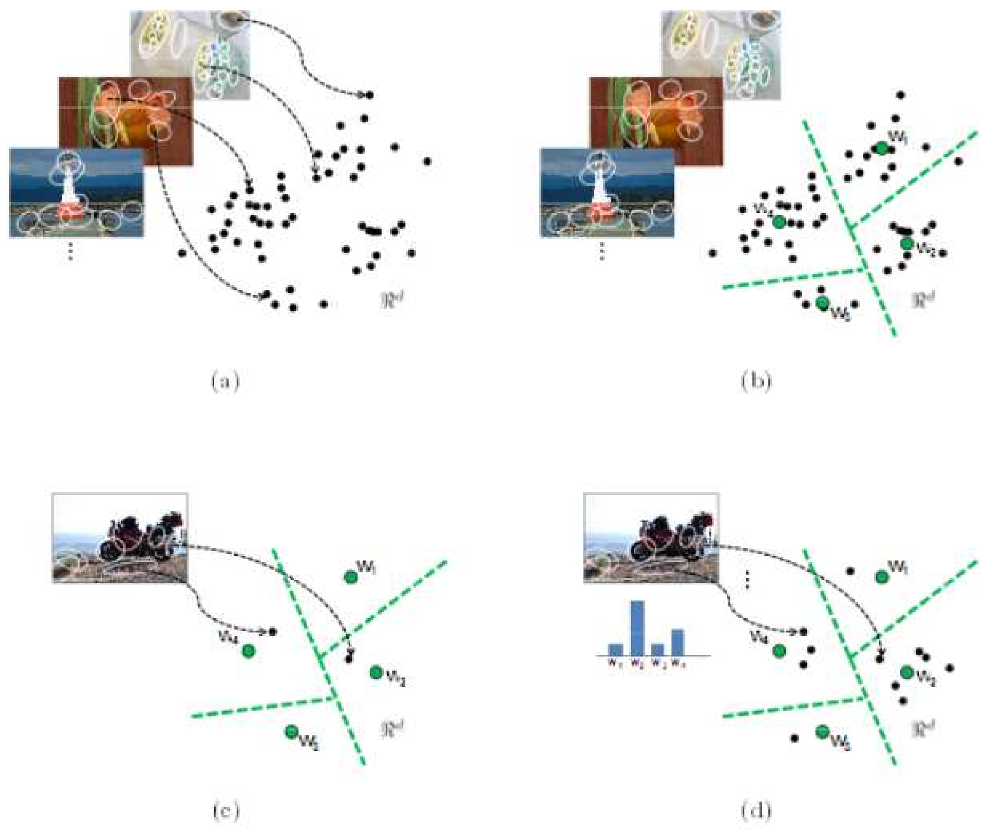


Figure 2.3: The process of forming visual words using the BoV model. Image from Graumann, visual object recognition.

Thus, these steps are mandatory for the BoV model:

- Methods to detect key points and describe local patches. This step is mandatory since these key points are used to form visual words. Various key point detectors such a Difference of Gaussian (Dog) and Laplacian of Gaussian (Log) exist and will be described in the next paragraph.
- Quantizing local descriptors into "visual words" or code-words. The collection of visual words are stored in a so called code-book. Thus an image will be represented by the frequency of all possible visual words. It is questionable how long such a code-book should be.
- Indexing and search. We may apply algorithms from text-retrieval systems, such as the inverted file.

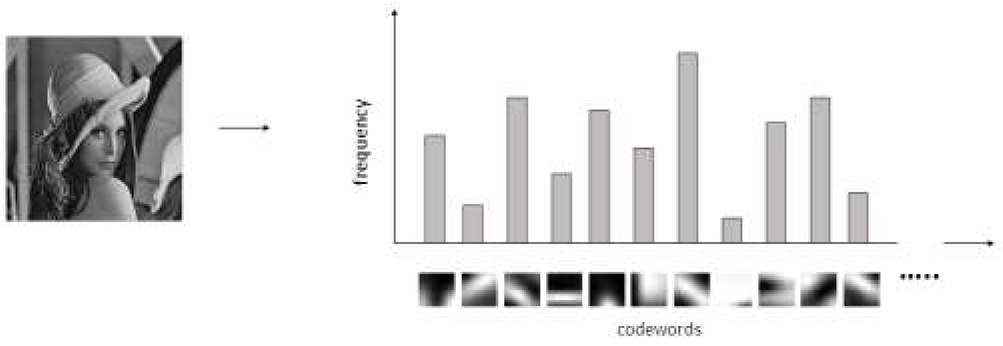


Figure 2.4: An image represented as vector using the BoV methodology

2.4 Key point Detector

The BoW model incorporates a pre-processing step consisting of the key point detector and local descriptor, as shown in (a) of Figure 2.2. Key points, also known as salient point, are interesting points in the image. If we observe an image, for example of two birds in the sky, we may find that 97% of the image is irrelevant to describing what is in the image. The human visual system detects interesting points, key point detection algorithms have been inspired by that. To be more specific, the goal of the key point descriptor step is to find locally stable points that ideally are robust to change in scale, rotation, translation and illumination. Various key point detection (and description) algorithms have been created such as SIFT, SURF and ORB. Each of these descriptors have different features, for example, the maximum angle an object may rotate along an axis to still be detected.

SIFT

The SIFT (Scale Invariant Feature Transform) keypoint detection and description algorithm is one of the most cited algorithms in computer vision [David G. Lowe 1999]. The keypoint detection algorithm consists of three stages: Scale-Space Extrema Detection, Keypoint Localisation and Orientation Assignment. Initially we wish to find keypoints that are invariant to scale change and rotation. The first step of the SIFT algorithm is to create a scale-space representation. Given an input image, the SIFT algorithm creates increasingly blurred out images by applying a Gaussian Blur. Each step will leave out some detail. The SIFT algorithm then takes the original image and scales it down by a factor two. The blurring processing is executed for that image again. This process keeps repeating a certain number of steps known as octaves. Finally, we will have many representations of the same image. The next step of the Scale-Space Extrema Detection stage is to apply a Laplacian of Gaussian (LoG) operator. To generate the LoG operation, the scale space is used. The difference between two scales of the scale space is used to calculate the LoG representation. This is

done by substracting the two scale space representations, for each octave. The next step is to find interesting keypoints, these are local minima and maxima. This is done by comparing the pixels in the current scale, the scale above and scale below. Next we arrive at the stage of Keypoint Localisation. First it removes keypoints with an intensity value lower than a certain value. Keypoints near the edges will also be removed. The final step is Orientation Assignment.
The goal of the second stage is to remove the keypoints that have a low contrast or are located near the edge, thus to eliminate the "weak" keypoints and to determine the location and scale of each keypoint. The aim of the third stage is to assign a consistent orientation to the keypoints based on local image properties. This is then used to create keypoint descriptors. A histogram (small area) around the keypoint is created. An orientation is assigned to this histogram. The final step is to create the SIFT keypoint descriptor, which we describe in the next paragraph.

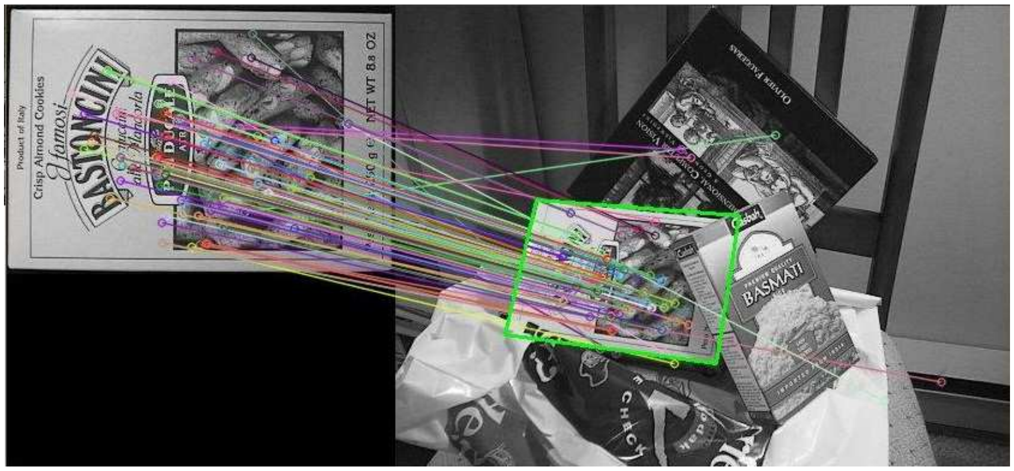


Figure 2.5: Image matching based on SIFT keypoints

SURF

SURF (Speeded-Up Robust Features) is an patented algorithm created by Bay et al. The algorithm is a scale- and rotation-invariant keypoint detector and descriptor. The general idea, like SIFT, is to find distinct image keypoints which can be used to do coresspondence matching between two images. These keypoints should be robust to scale, rotation, illumination and viewpoint transformation. Their implementation outperformed the SIFT implementation yet was much faster.

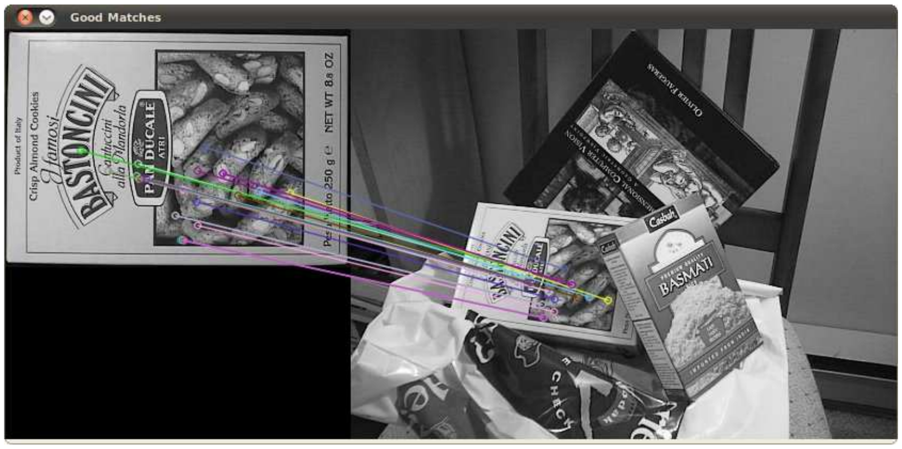


Figure 2.6: Image matching based on SURF keypoints

ORB

ORB (Oriented Brief) is an alternative to the SIFT and SURF algorithms published [Rublee et al 2011]. The authors claim that their algorithm is two orders of magnitude faster than the SIFT algorithm. Unlike the SIFT and SURF algorithm, ORB is not patented. The algorithm is based on the FAST algorithm to detect keypoints and the BRIEF descriptor. The SIFT descriptor is of size 128 elements. The SURF descriptor is of size 64 or 128 elements.

2.5 Local descriptor

Once we have extracted the key points, we may wish to describe the area the identified the key points in. This is known as a local descriptor. Ideally we would like these local descriptors to be invariant to many properties such as scale, distortion or rotation. Some of the well known local description algorithms are SIFT, SURF, LBP, BRIEF and HOG. We will only describe the first two descriptors. The creation of the SIFT descriptor has three general steps: (1) take a 16x16 window around an keypoint, (2) divide it into 4x4 cells, (3) compute histogram of image gradients in each cell (8 bins each). Finally, all these histograms and orientation are concatenated into a 128 elements SIFT descriptor.

2.6 Spatial Pyramid Matching

The BoV model introduced above discards any spatial information. Lazebnik et al introduced Spatial-Pyramid Matching (SPM) which encodes spatial information. The method had initially been created for recognizing scenes such as highway, office, street, forest etc. The SPM model is inspired by the work of Oliva and Torralba, in which they found that

people can recognize scenes while overlooking various details - and thus perceive scenes in a holistic way. Thus scenes may be recognized or classified based on the spatial layout of the image while neglecting the details. An essential concept in SPM is the increasingly finer grids over the feature space. First, let us construct a sequence of grids 0, ..., L, such that at each gird level l there are 2^l cells. In total there will be $2^d l$ cells. For each level we construct an histogram H by concatenating the cells. Finally, we concatenate the histograms of all levels into one (large) histogram.
The model discards the geometric in-variance that was present in the BoV model. That is, because of the strict encoding of the image into sub-cells, it is not invariant to translation, rotation or scale. Changing one or more of these properties may have a negative impact on the results when the SPM model is used as compared to the BoV model. However, in the model of Lazebnik the SPM algorithm outperforms the BoV model. Thus, depending on the image data-set the SPM algorithm may outperform the SPM algorithm in terms of precision.

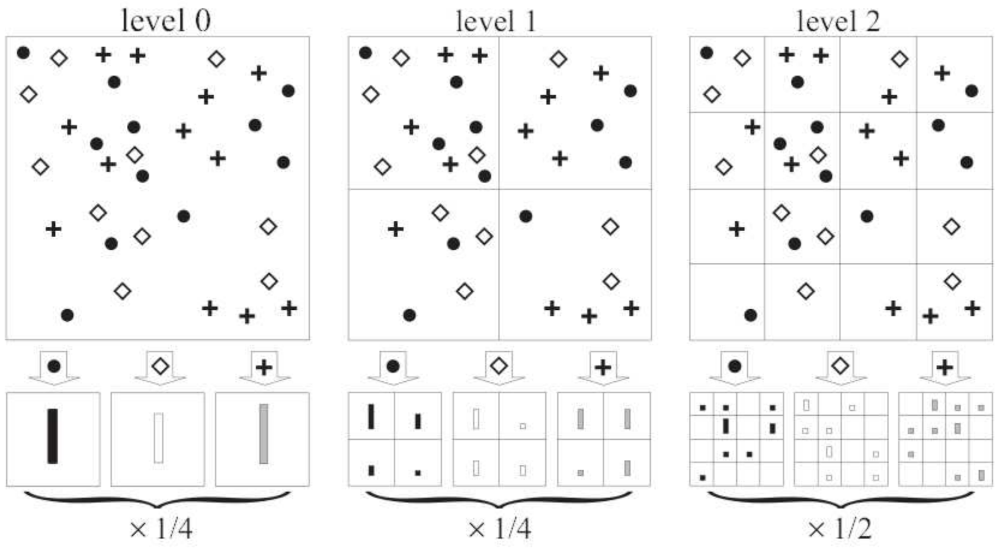


Figure 2.7: Spatial pyramid matching, a toy example (Source: Lazebnik).

2.7 Geometry-preserving Visual Phrases

Zhang et al proposed an algorithm that extends the Bag-of-visual-words model with spatial information, known as Geometric-preserving Visual Phrases (GVP). In their work they claim the GVP algorithm has a higher precision than the BoV model. The GVP algorithm captures both local and long-range spatial relationships. The model, as proposed by the authors, is only robust to translation.

Spatial information is encoded trough the formation of geometric-preserving visual phrases. A geometric-preserving visual phrase (GVP) is defined as k visual words in a certain spatial layout. Thus, a GVP is a set of (unique) visual words which are extracted using the BoV methodology. The GVPs which can be formed depend on the spatial layout - different spatial layouts will form different GVPs. Unlike the SPD model it does not strictly impose a certain spatial order in which the visual words should appear. The number of visual words in a geometric-preserving visual phrase can be of any length greater or equal than two.
In an image, a number of GVPs may appear, and we represent images as collections of GVPs. Thus: Each image is represented as a vector defined by GVP where the ith position is the frequency of a visual phrase, as shown in the figure below. This is somewhat similar to the BoV representation, where the ith position is the frequency of a visual words.
A vector defined by GVP can be very long, in example: exponential to the amount of words we find. It is impractical to calculate these vectors directly. The authors propose to find GVPs by an algorithm known as co-occurring phrases. This algorithm identifies co-occurring GVP phrases, which occur given two images.

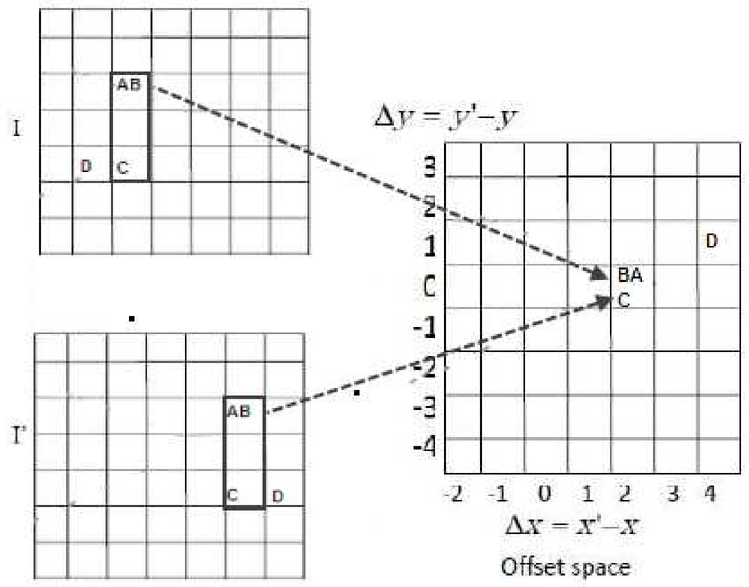


Figure 2.8: Illustration of the co-occurrences of the same GVP.

Each image is divided into a certain predefined spatial layout - as shown in figure 2.7. Both image I and I' should be defined by the same spatial layout. In this example, a 6x7 regions raster is used for both I and I'. Naturally, a 6x7 sub-division may not be optimal and perhaps it would be better to use a spatial layout of a different size. Nevertheless, the authors do not strictly impose any spatial layout.

In this algorithm, an offset space is created in which it identifies geometry-preserving visual phrases. A geometry preserving visual phrase is an offset at which there appear more than one words. In this example, there exist various GVPs of length k=2 such as <b,f >, <d,f >, <a,c >, <a,b >and <b,c >. There is one GVP of length k=3: <a,b,c >. Note that visual words occuring may also be used in addition to the GVPs. In the example the visual word <a >is appearing in the offset space. For each visual word that is present in both I and I' we calculate its offset value $(\Delta X, \Delta Y)$.
The algorithm calculates the offset of the word j in the offset space based on the location in image I and I'. The x value is the I'x subtracted by Ix and likewise the y value is I'y subtracted by Iy.
For example, if we wish to find the location of word C in the offset space, we find its location in image I: (3,3) and in I': (5,3). The we calculate the offset by subtracting I' and I and find (2,0), which is shown in the offset space in figure 2.7. A visual word may appear both on the positive and negative side of the axis.
We show a possible implementation of the algorithm in pseudo code for creating the offset space below:

Algorithm 1 Algorithm for creating the offset space in pseudo code using a dynamic programming approach

```
offset[][]
I[]=words[0..N]
I'[]=words[0..M]
for $i = 0$ to N do
    for $j = 0$ to M do
        if (I[i].word == I[j].word) then
            O = new word();
            O.x = I'[j].x - I[j].x;
            O.y = I'[j].y - I[j].y;
            offset[Ox][Oy].insert(I[j].word);
        end if
    end for
end for
```

Note that the locations of the visual words in vectors I and I' should have been determined previously according to a certain spatial layout. Finally, we can easily count the number of GVPs iterating over the offset space:

The GVP model, as presented by the authors, only considers translation and ignores rotation and scale change. Alas, this method might be improved in terms of accuracy by including rotation and scale in-variance to some degree.

Algorithm 2 Algorithm for determining the number of GVPs

```
numberOfGVP = 0;
for $i = 0$ to N do
    for $j = 0$ to M do
        if { offset[i][j] ¿ 1 } then
            numberOfGVP++;
        end if
    end for
end for
```

2.8 Weighting schemes

Term weighting schemes may influence the accuracy of a text retrieval system. A weighting scheme can take into account the frequency of visual words or simply neglect it, which may influence the accuracy of such a system.

Binary

The binary weighting scheme is a vector defined by BoV that for each element of the vector indicates if a visual word is present or absent. The length of this vector is the number of all possible visual words and the frequency of the visual word is simply neglected. The memory consumption of binary vectors defined by BoV may be very small - we can store eight elements in one byte as a byte contains eight bits. An example of the binary weighting scheme is shown in the table in paragraph 2.1. The binary weighting scheme is perhaps the most simple of all the BoV weighting schemes.

TF-IDF

Term Frequency Inverse Document Frequency (TF-IDF) is a weighting scheme consisting of two parts: Term Frequency (TF) and Inverse Document Frequency (IDF). TF-IDF is simply the multiplication of TF and IDF factors. Formally:

tf-idf(t) = tf(t,d) x idf(t)
where t is the term occurring in document d.

This value will be the highest if the term occurs frequently in a small number of documents. If the term appears in all documents it is not representative and thus very low. To find Term Frequency (TF) we store the frequency of the term t occurring in document d. The Inverse Document Frequency (IDF) is calculated by taking the log of the number of documents —D— divided by the number of documents the term T is in.
Finally, the TF-IDF factor is calculated by multiplying the previously calculated factors.

3. Approach

Introduction

All algorithms (BOV,SPD,GVP,OUR) will be implemented in C++ for speed and portability. We will evaluate these algorithms based on compuation time and accuracy. We will use the following architecture for the system:

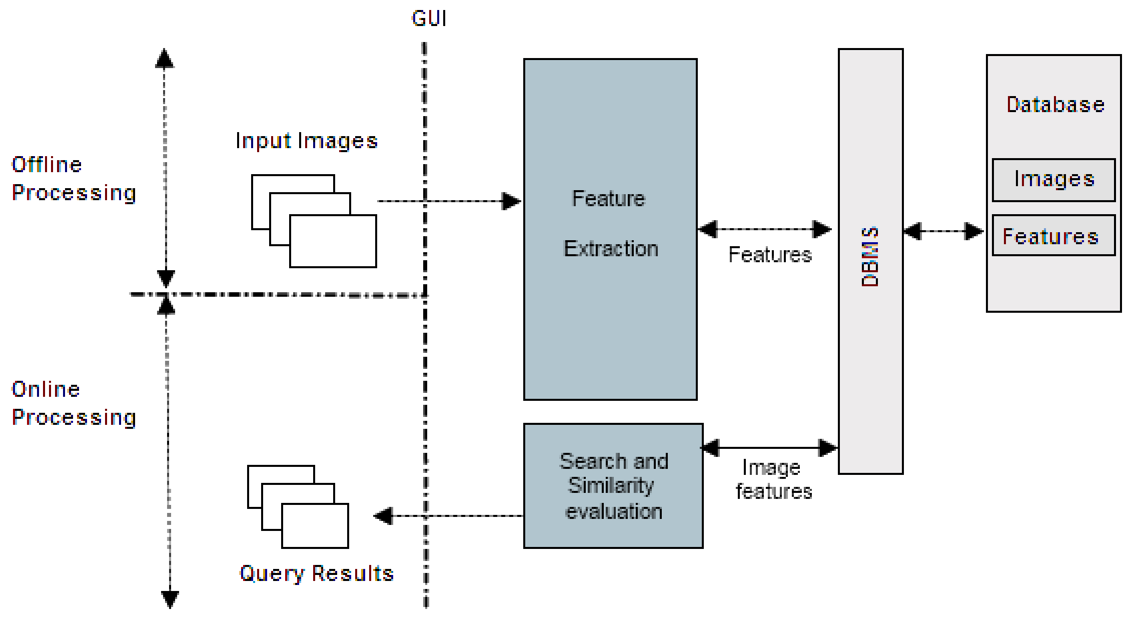


Figure 3.1: Component diagram of IRS.

It consists of an offline processing stage, which extracts all features from the dataset and an online processing set, at which a user may query the system. Note that in the feature step, we extract both salient point features and the BoV features. The architecture and system is described in more detail in the appendix. In our experiments, we will use various image datasets which we will describe in the next section.

Datasets

Datasets: In our experiments we will use various datasets which we named Web4k, Web18k and Web40k. All of these sets were collected from the World Wide Web (www) and are of the size 64x64. None of the algorithms used color as feature, for this reason we show grayscale images. The Web4k consists of 4148 digital images which we downloaded from the web. The images are divided into five groups: motorcycles, faces, airplanes, statues and cars. The objects are shown from various angles and scales. The Web18k consists of 18.745 digital images. These were divided into various categories: car, face and fish. The Web40k set conists of 41.182 digital images. These were divided into several groups: airplane, bike, car, fish and face.

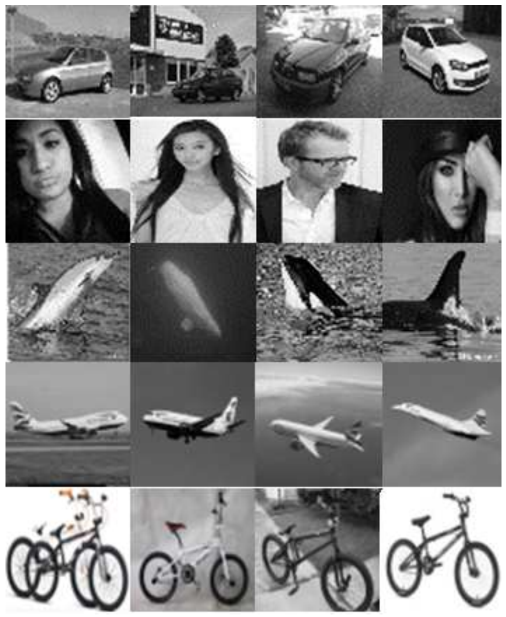


Figure 3.2: Example images of the Web40k set.

Experiments

In our first experiment we wish to implement and evaluate several state of the art algorithms including the BoV baseline, SPM and GVP. The SURF features will be used to reduce the computation time. Both SURF keypoints and descriptors will be used. All algorithms will be programmed in C++ to achieve a high computation time and use the OpenCV API (a computer vision library) available at *http://opencv.org*. The OpenCV library can be used on various operating systems including Linux, Windows and Mac OS X.
For the first dataset we will use a dictionary size of 8000 visual words and a threshold of 1000 for the SURF algorithm. Since the other sets are larger, we decided to use a dictionary size of 40000 visual words and a threshold of 4000 for the other datasets. All algorithms will

use the same parameters for each dataset upon evaluation. For each dataset we have more than 50 query images.

The experiments will be conducted on an Intel Core i5-3570K CPU @ 3.40 Ghz machine running the Windows 7 64-bit operating system (service pack 1).

Extended Geometry-preserving Visual Phrases

The GVP algorithm presented by Zhang et al. is limited to translation in-variance. A large set of images may be neglected simply because they are in a different rotation or scale. If we wish to preserve geometrical phrases we should also incorporate rotation and scale in-variance. If these are incorporated the precision may become higher. We propose a solution that only slightly affects the memory usage and retrieval time. The algorithm we propose is based on the BoV model and ment to extend it by preserving spatial information.

We define a visual phrase as k visual words in a certain spatial layout. First we create a different offset space. Given an image I and I' from which we find the same word X resulting in two vectors X and X'. From these two vectors we calculate the L2 distance. We then create an offset space of size L2 x L2 where we store the word based on its L2 size. This allows for the visual words to rotate within an area in their image space.

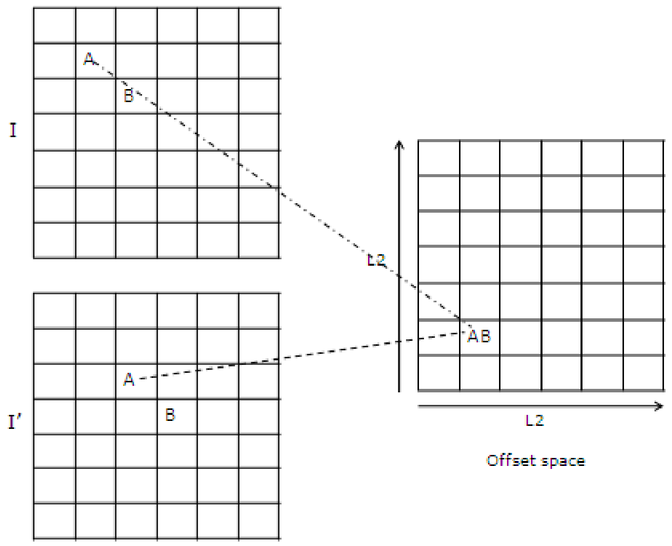


Figure 3.3: Illustration of our visual phrase identification

We list the creation of our new offset space below, in this example we used a simple L2 distance:

Thus, a if we have a visual phrase of length k=2 it will be found in this model - independent of where the second word is located as long as it is within the same distance.

Algorithm 3 Algorithm for creating the offset space in pseudo code using a dynamic programming approach

```
offset[][]
I[]=words[0..N]
I'[]=words[0..M]
for $i = 0$ to N do
    for $j = 0$ to M do
        if (I[i].word == I[j].word) then
            O = new word();
            O.d = $\sqrt{(I'[j].x - I[j].x)^2 + (I'[j].y - I[j].y)^2}$
            offset[O.d][O.d].insert(I[j].word);
        end if
    end for
end for
```

This allows for deformation and rotation to some degree without significantly increasing the computation time.

4. Results

The BoV algorithm performed the lowest on all data sets in terms of precision. The SPD algorithm improved the accuracy in general as compared to BoV, but GVP and our algorithm did significantly better. Our algorithm outperformed the GVP algorithm on average by a factor of 1% to 1.6%. Compared to the BoV algorithm it did 6% to 21.3% better, depending on the number of queries and dataset.

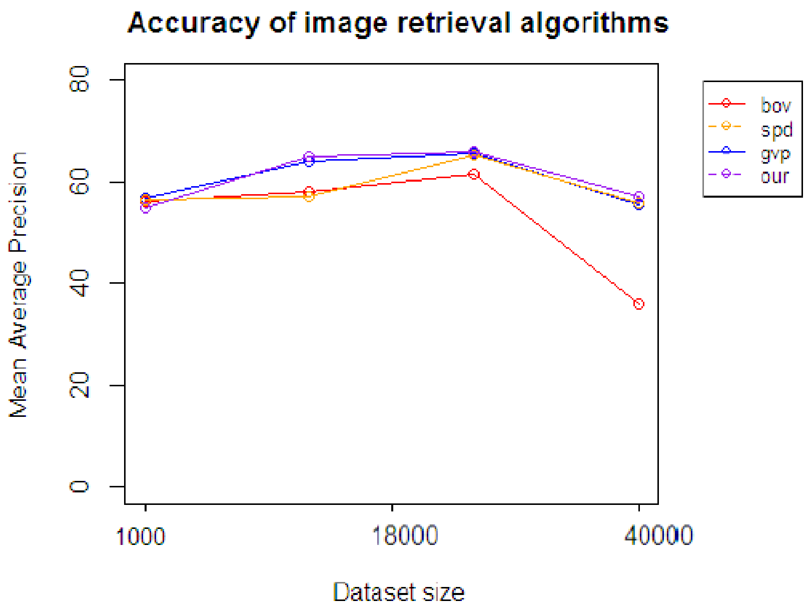


Figure 4.1: Accuracy of image retrieval algorithms

	Web1k	Web4k	Web18k	Web40k
BOV+SURF	56.2	58	61.46	35.9
GVP+SURF	56.9	64	65.63	55.6
OUR+SURF	54.8	65	65.97	57.2
SPD+SURF	56.5	57	65.28	55.89

BoV had the lowest query time, which is logical since the other algorithms are based on BoV. SPD had a shorter querying time than GVP (both for k=2 and k=3) and our algorithm.

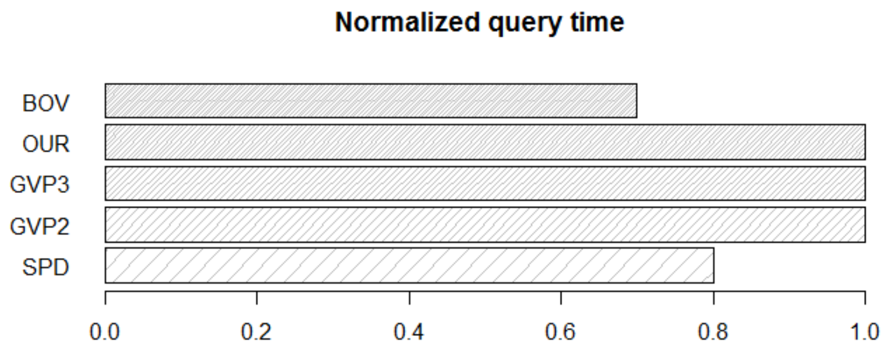


Figure 4.2: Query time for the algorithms

Furthermore, we found that various parameters such as the dictionary size and spatial layout heavily influences the accuracy. Which spatial layout is ideal depends may be dependent on the image size. For example, a 10x10 raster may be ideal for small images, but on a satellite images we may wish a much larger spatial raster.

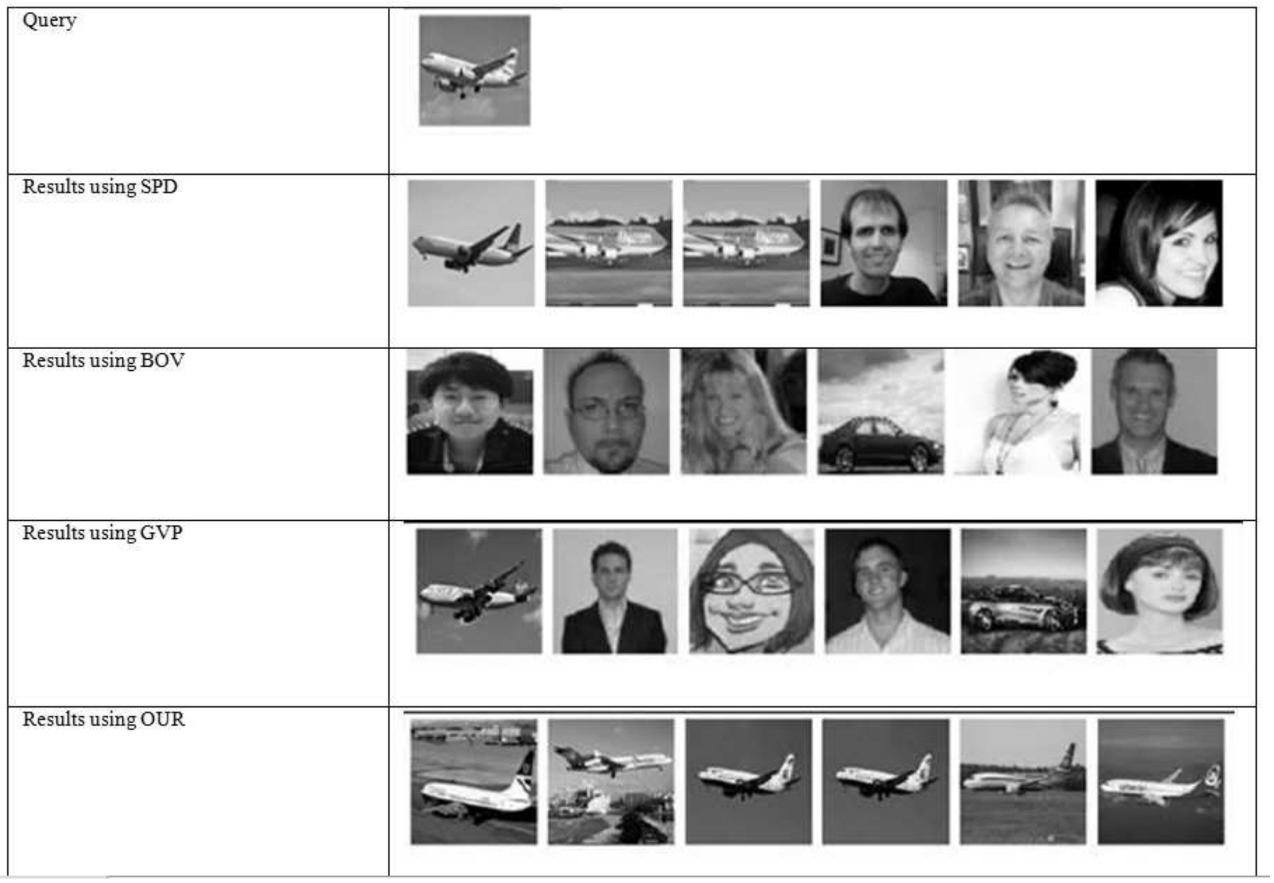


Figure 4.3: Illustration of search results using the algorithms

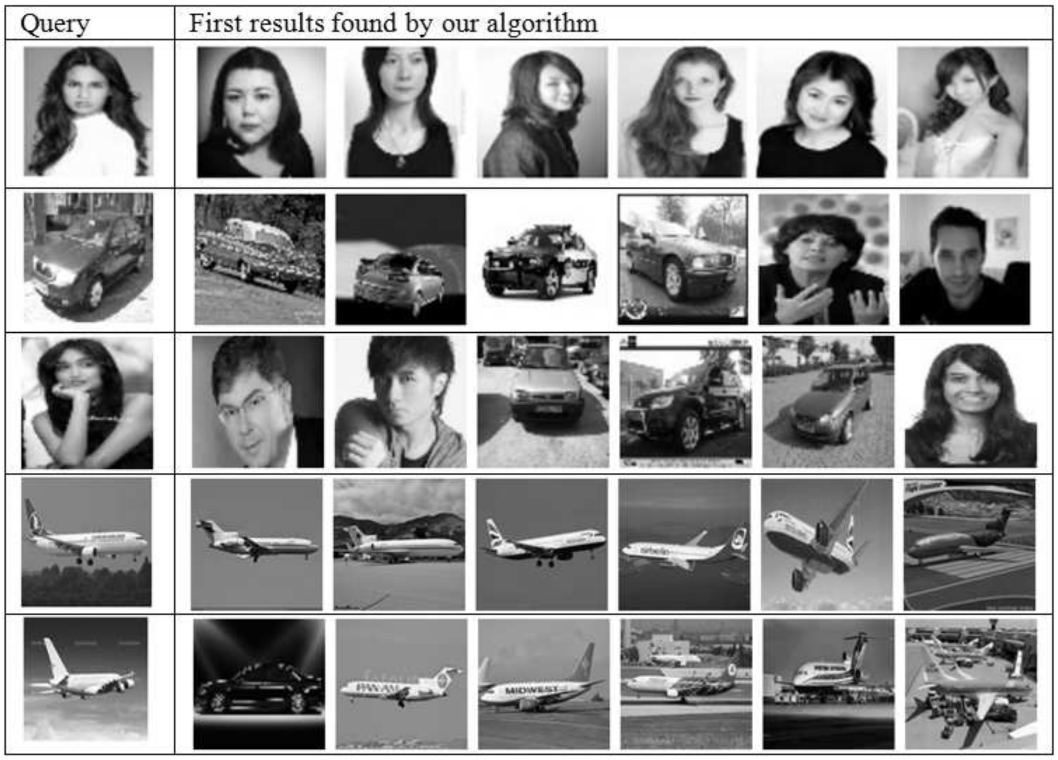


Figure 4.4: Retrieval results of our algorithm

5. Conclusions

Our key contribution is an algorithm for image retrieval that outperforms the state-of-the-art CBIR algorithms known as BOV, SPM and GVP in terms of precision.
We found that the accuracy of the GVP and our algorithm is heavily dependent on these parameters:

1. dictionary size
2. spatial layout
3. length of GVP phrases (GVP only)

The BoV baseline had the lowest precision on average for the datasets, as we expected. The SPD algorithm can be more accurate than the BoV algorithm, but due its strong spatial encoding this may not always be the case. The GVP algorithm was more precise than BoV and SPD on these datasets, but was less precise than our algorithm.
We found that SPD had a shorter computation time than GVP and our algorithm. BoV had the shortest computation, as expected.
Furthermore we created a CBIR system, which is capable of using various state-of-the-art algorithms and to use large datasets.

6. Future work

A new research may focus on evaluating these CBIR algorithms on larger datasets, perhaps beyond the scale of a single machine in a distributed setting. Another option would be to evaluate these algorithms for content-based retrieval in videos. A third option would be to evaluate these algorithms for autonomous robotic navigation.

A. IRS system

A.1 Introduction

IRS (Image Retrieval System) is a content based image retrieval system created in **C++**. The motivation for C++ is portability to a variety of platforms. At present, IRS has been tested on the Microsoft Windows platform, however it should be easy to port. The IRS system makes use of the computer vision library known as OpenCV. It has C++, C, Python and soon Java interfaces running on Windows, Linux, Android and Mac. The library has more than 2500 optimized algorithms. The large number of computer vision algorithms and portability of this library are the main reasons for inclusion of this library.
Content based image retrieval (CBIR) systems are often split into three sub systems which run on different machines: graphical user interface (GUI), database management system (DBMS) and the image processing engine. For example, the GUI runs in the webbrowser of a users client machine while the processing and database are on one or more powerful server machines. The current implementation of IRS does not include a client-server architecture, as our main goal was to evaluate and improve on existing CBIR algorithms. A client-server architecture can however, easily be incorporated.
IRS is built in the following way:

- Algorithms in C++
- GUI interface in HTML
- Database in C integrated with Image Processing library

The user interface is a simple web interface instead of a desktop interface, since we think the web will be around for a long time. A user may query an image database given an example image. There is no support for text-based querying at the moment. Querying the image database is done by an image query. The system we created can be used for research and/or education. It provides a starting point for evaluating and implementing new CBIR algorithms in future research.
The system contains implementations of state of the art CBIR algorithms including Bag of visual words (BoV), Spatial Pyramid Matching (SPM) and Geometry-preserving visual phrases (GVP). It has support for various weighting schemes including binary and TF-IDF.

A.2 Block Diagram of the System

The system we created uses an architecture that is typically seen in CBIR systems. Initially, d-dimensional vectors are extracted from an image database based on an computer vision algorithm. In our work we have used the SURF descriptor, which we have used as initial step for the other algorithms such as BoV and GVP. It is important to note that this image database does not contain any text-annotation. Once the feature vectors are extracted for each of the images, we may query the system. For an query image, the system extracts a feature vector and calculates the distance between the query and the feature vectors in memory. Finally, the system returns visually similar images.
To be more specific, the software is divided into two steps: offline and online processing. Offline-processing may take a long time, depending on the size of the dataset. For each image in the dataset, the software will extract features (For example, SIFT keypoints and descriptors).

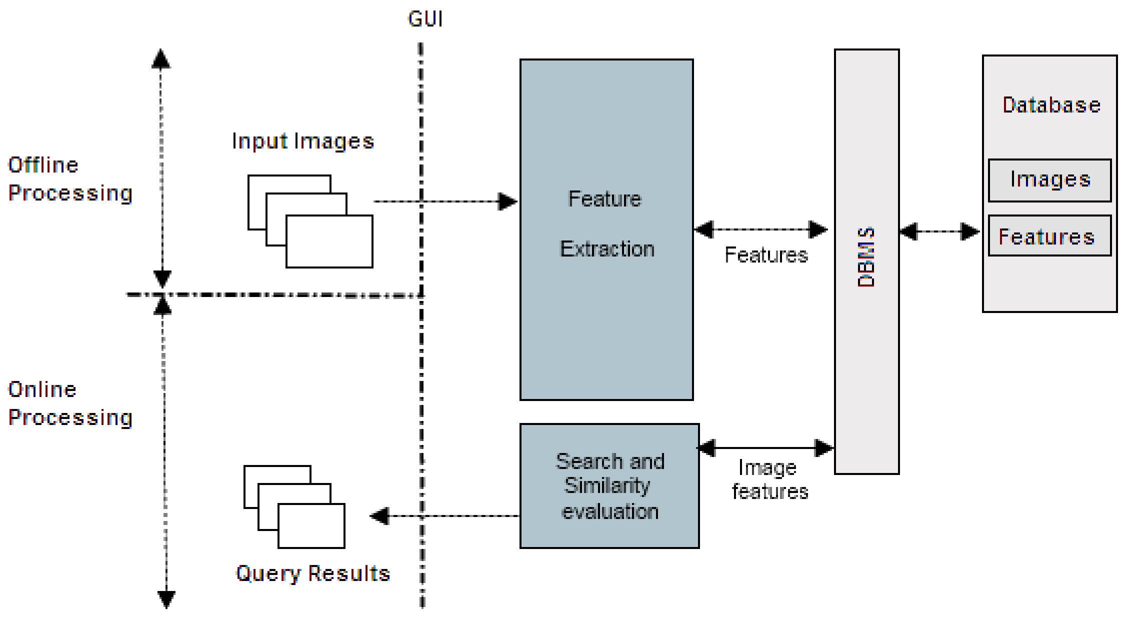


Figure A.1: Component diagram of IRS.

A.3 Feature Extraction

Feature Extraction is a core step of the IRS system. Recent algorithms such as BOV, GVP, SPD and our algorithm use salient point features as input for these algorithms. A wide variety of salient point detectors have may be used, however we chose to use the SURF keypoint detector.

A.4 Requirements

In order to compile the software, one needs

- C++ compiler
- 64 bit operating system (currently only tested with Windows 7).

The system requirements for the CBIR system are:

- More than 4GB ram for large datasets
- Windows 7 64 bit
- MSVC 2010 x64 redistributable

A.5 Example usage

First, the content based image retrieval system has to be started. Upon starting, the user should specify the desired CBIR algorithm. Current choices are:

- Bag of visual words (BoV)
- Spatial Pyramid Matching (SPM)
- Geometry-preserving visual phrases (GVP)
- Our

This will start the initialization step, which may take a while. After initialization is completed, a user may select a query image from the disk. The output will be given as a HTML file.

A.6 Results

Output of the system can be shown as a rectangular grid within a web-browser. The query image is shown on top, and the results are shown below the horizontal line. In the example below most of the images are relevant, and a small fraction is irrelevant. Al though the computer vision problem is not yet solved, we think the field is very promising. At the moment the rectangular grid is the only way to display the results - nevertheless the display of the results can easily be modified.

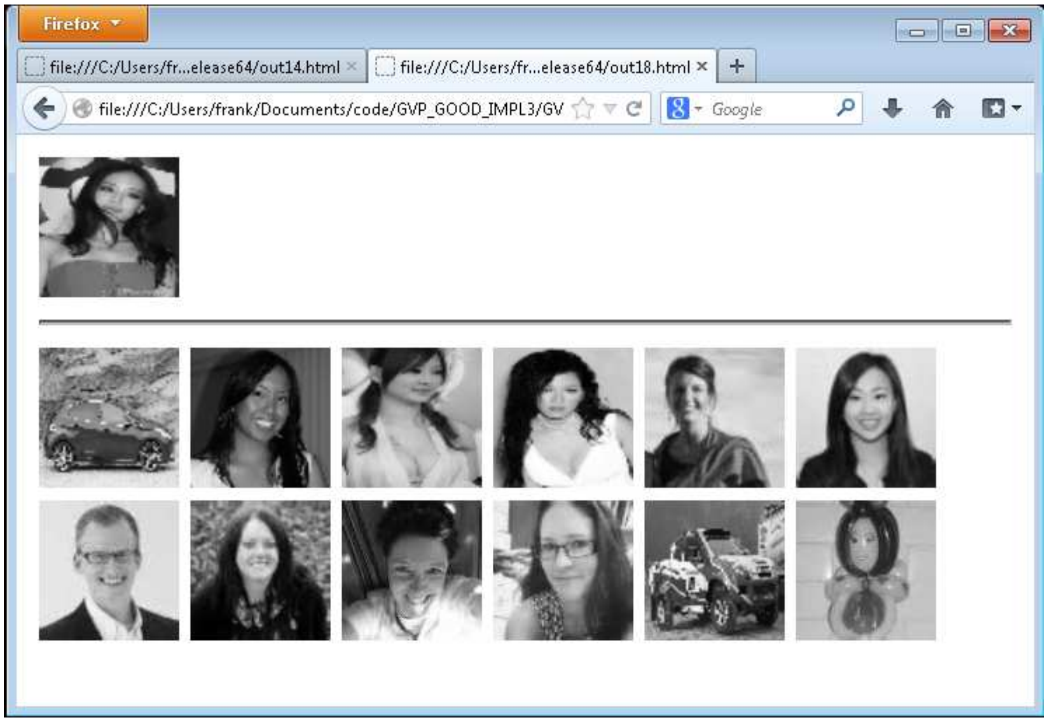

Firefox
file:///C:/Users/fr...elease64/out14.html
file:///C:/Users/fr...elease64/out18.html
file:///C:/Users/frank/Documents/code/GVP_GOOD_IMPL3/GV
Google

Figure A.2: Results of an example query in our CBIR system

Bibliography

[1] Efficient and Accurate Face Detection using Heterogeneous Feature Descriptors and Feature Selection Computer Vision and Image Understanding (October 2012), doi:10.1016/j.cviu.2012.09.003 by Hong Pan, Yaping Zhu, Liangzheng Xia

[2] Image retrieval with geometry-preserving visual phrases In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on (June 2011), pp. 809-816, doi:10.1109/cvpr.2011.5995528 by Yimeng Zhang, Zhaoyin Jia, Tsuhan Chen

[3] ORB: An efficient alternative to SIFT or SURF In Computer Vision (ICCV), 2011 IEEE International Conference on (November 2011), pp. 2564-2571, doi:10.1109/iccv.2011.6126544 by E. Rublee, V. Rabaud, K. Konolige, G. Bradski

[4] Spatial-bag-of-features In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on (June 2010), pp. 3352-3359, doi:10.1109/cvpr.2010.5540021 by Yang Cao, Changhu Wang, Zhiwei Li, Liqing Zhang, Lei Zhang

[5] img(Anaktisi): A Web Content Based Image Retrieval System, 2009 Second International Workshop on Similarity Search and Applications, Konstantinos Zagoris, Savvas A. Chatzichristofis, Nikos Papamarkos and Yiannis S. Boutalis.

[6] Learning OpenCV: Computer Vision with the OpenCV Library (01 October 2008) by Gary Bradski, Adrian Kaehler

[7] TF-IDF uncovered: a study of theories and probabilities In Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (2008), pp. 435-442, doi:10.1145/1390334.1390409 by Thomas Roelleke, Jun Wang

[8] Speeded-Up Robust Features (SURF) Comput. Vis. Image Underst., Vol. 110, No. 3. (June 2008), pp. 346-359, doi:10.1016/j.cviu.2007.09.014 by Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool

[9] Evaluating bag-of-visual-words representations in scene classification In MIR '07: Proceedings of the international workshop on Workshop on multimedia information retrieval (2007), pp. 197-206, doi:10.1145/1290082.1290111 by Jun Yang, Yu G. Jiang, Alexander G. Hauptmann, Chong W. Ngo

[10] Mind As Machine: A History of Cognitive Science (31 August 2006) by Margaret Boden

[11] Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, Vol. 2 (09 October 2006), pp. 2169-2178, doi:10.1109/cvpr.2006.68 by S. Lazebnik, C. Schmid, J. Ponce

[12] VisualSEEk: a Fully Automated Content-Based Image Query System. In ACM Multimedia, Boston, MA, November 1996 by John R. Smith, Shih-Fu Chang.

[13] Tools and Techniques for Color Image Retrieval. In IST/SPIE Symposium on Electronic Imaging: Science and Technology (EI'96) - Storage and Retrieval for Image and Video Databases IV, Volume 2670, San Jose, CA, February 1996 byJohn R. Smith, Shih-Fu Chang.

[14] Histograms of oriented gradients for human detection Computer Vision and Pattern Recognition, IEEE Computer Society Conference on In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, Vol. 1 (June 2005), pp. 886-893 vol. 1, doi:10.1109/cvpr.2005.177 by N. Dalal, B. Triggs

[15] Distinctive Image Features from Scale-Invariant Keypoints Int. J. Comput. Vision In International Journal of Computer Vision, Vol. 60, No. 2. (1 November 2004), pp. 91-110, doi:10.1023/b:visi.0000029664.99615.94 by David G. Lowe

[16] PCA-SIFT: a more distinctive representation for local image descriptors Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on In 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2 (2004), pp. 506-513, doi:10.1109/cvpr.2004.1315206 by Yan Ke, R. Sukthankar

[17] Relevance feedback in image retrieval: A comprehensive review Multimedia Systems, Vol. 8, No. 6. (4 April 2003), pp. 536-544, doi:10.1007/s00530-002-0070-3 by Xiang S. Zhou, Thomas S. Huang

[18] Video Google: a text retrieval approach to object matching in videos Computer Vision, IEEE International Conference on In Proceedings Ninth IEEE International Conference on Computer Vision, Vol. 2 (03 October 2003), pp. 1470-1477 vol.2, doi:10.1109/iccv.2003.1238663 by Sivic, Zisserman

[19] The many faces of configural processing Trends in Cognitive Sciences, Vol. 6, No. 6. (June 2002), pp. 255-260, doi:10.1016/s1364-6613(02)01903-4 by Daphne Maurer, Richard L. Grand, Catherine J. Mondloch

[20] Next-Generation Web Searches for Visual Content Computer, Vol. 33, No. 11. (2000), pp. 46-53, doi:10.1109/2.881694 by Michael S. Lew

[21] CBIR: from low-level features to high-level semantics In Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 3974 (April 2000), pp. 426-431 by X. S. Zhou, T. S. Huang edited by B. Vasudev, T. R. Hsing, A. G. Tescher, R. L. Stevenson

[22] Content-based image retrieval at the end of the early years Pattern Analysis and Machine Intelligence, IEEE Transactions on, Vol. 22, No. 12. (06 December 2000), pp. 1349-1380, doi:10.1109/34.895972 by A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, R. Jain

[23] Content-Based Image Retrieval Systems: A Survey, Remco C. Veltkamp, Mirela Tanase, 2010.

[24] Query by Image and Video Content: The QBIC System Computer, Vol. 28, No. 9. (September 1995), pp. 23-32, doi:10.1109/2.410146 by Myron Flickner, Harpreet Sawhney, Wayne Niblack, et al.

[25] Practical Computer Vision with SimpleCV (July 2012), Kurt Demaagd,Anthony Oliver,Nathan Oostendorp,Katherine Scott.