Topics in Content Based Image Retrieval
Fonts and Color Emotions

Martin Solli
to Melvin
Abstract

Two novel contributions to Content Based Image Retrieval are presented and discussed. The first is a search engine for font recognition. The intended usage is the search in very large font databases. The input to the search engine is an image of a text line, and the output is the name of the font used when printing the text. After pre-processing and segmentation of the input image, a local approach is used, where features are calculated for individual characters. The method is based on eigenimages calculated from edge filtered character images, which enables compact feature vectors that can be computed rapidly. A system for visualizing the entire font database is also proposed. Applying geometry preserving linear- and non-linear manifold learning methods, the structure of the high-dimensional feature space is mapped to a two-dimensional representation, which can be reorganized into a grid-based display. The performance of the search engine and the visualization tool is illustrated with a large database containing more than 2700 fonts.

The second contribution is the inclusion of color-based emotion-related properties in image retrieval. The color emotion metric used is derived from psychophysical experiments and uses three scales: activity, weight and heat. It was originally designed for single-color combinations and later extended to include pairs of colors. A modified approach for statistical analysis of color emotions in images, involving transformations of ordinary RGB-histograms, is used for image classification and retrieval. The methods are very fast in feature extraction, and descriptor vectors are very short. This is essential in our application where the intended use is the search in huge image databases containing millions or billions of images. The proposed method is evaluated in psychophysical experiments, using both category scaling and interval scaling. The results show that people in general perceive color emotions for multi-colored images in similar ways, and that observer judgments correlate with derived values.

Both the font search engine and the emotion based retrieval system are implemented in publicly available search engines. User statistics gathered during a period of 20 respectively 14 months are presented and discussed.
Acknowledgements

A number of people have helped and supported me during this work. First and foremost, I would like to thank my supervisor Reiner Lenz. His support and guidance, valuable suggestions, and never ending pile of ideas, made this work possible. I would also like to thank my co-supervisor Professor Björn Kruse, and both former and current colleagues at the department for creating an inspiring working atmosphere and enjoyable coffee breaks.

The companies Matton Images (Stockholm, Sweden) and Picsearch AB (publ) are acknowledged for providing image material and interesting ideas.

On a personal level, I would like to express my deepest gratitude to my family and friends for all their encouragement. A special thanks to my beloved Elin and our son Melvin, who experienced his first months in life while I was writing this thesis.

Thank you

Martin Solli
Norrköping, Sweden
February 2009
Contents

Abstract v
Acknowledgements vii
Table of Contents ix

1 Introduction 1
   1.1 Background .............................................. 1
   1.2 Aim and Motivation ...................................... 2
   1.3 Originality and Contributions .......................... 3
   1.4 Outline of the Thesis .................................. 3

2 Image Retrieval: Past, Present and Future 5
   2.1 Research Interests ..................................... 5
   2.2 Commercial Implementations ............................ 9

3 Font Retrieval 13
   3.1 Introduction ........................................... 13
   3.2 Background ............................................ 15
   3.3 Character Segmentation ................................ 21
       3.3.1 Skew estimation and correction .................. 22
       3.3.2 Character segmentation .......................... 22
   3.4 Basic Search Engine Design ............................ 24
       3.4.1 Eigenfonts basics ................................. 26
       3.4.2 Character alignment and edge filtering ........ 27
       3.4.3 Selection of eigenimages ........................ 28
   3.5 Choosing Components .................................. 31
       3.5.1 Image scaling and interpolation ................ 32
       3.5.2 Features not derived from eigenimages ........ 33
       3.5.3 Measuring similarity .............................. 33
   3.6 Results .................................................. 34
       3.6.1 Method composition ............................... 34
Chapter 1

Introduction

In this chapter the background of this thesis is described, together with its aim and motivation. Next the originality and contributions are discussed. The chapter is ended with an outline of the thesis.

1.1 Background

In the beginning of this decade we witnessed a digital revolution in imaging, not least for applications targeting the mass market. The use of digital cameras increased dramatically, and thereby the amount of digital images. Many of us have private collections with thousands of digital images. Naturally, we share images with each other, and also publish them for instance on the Internet. Those image collections are important contributors to the public domain of the Internet, nowadays containing several billion images. Private image collections, or images on the Internet might be the most obvious example, but the use of digital imaging has spread to many application areas. Modern hospitals are good examples, where large collections of medical images are managed and stored every day. Newspapers, image providers, and other companies in the graphic design industry, are now using digital images in their workflow and databases. A third example is the security industry, where surveillance cameras can produce tremendous amounts of image material.

Some image collections are highly organized with keywords, making text-based search efficient for finding a specific image, or images with a particular content. However, most images are poorly labeled, or not labeled at all. Consequently, with the amount of images increasing, it is necessary to find other ways of searching for images. As an alternative to text-based search, we can think of tools that can "look into images", and retrieve images, or organize large image collections, based on image content. The research area based on this idea is called Content Based Image Retrieval (CBIR).
2

Introduction

Content Based Image Retrieval has been an active topic in our research group for many years now. Reiner Lenz together with Linh Viet Tran [93] were one of the first groups working on image database search. In the beginning of this decade the research was continued by Thanh Hai Bui [5]. Contributions were implemented in the publicly available search engine ImBrowse\(^1\), a search engine focusing on general descriptors for broad image domains. Since then the amount of research targeting in CBIR has increased dramatically. With this thesis we continue the in-house tradition in Content Based Image Retrieval, but with another focus, now centered on the specialized topics of font retrieval and emotion based image retrieval.

1.2 Aim and Motivation

The overall aim of this thesis is to explore new ways of searching for images based on various content features, with focus on new and specialized topics.

The first part of the thesis will focus on font retrieval, or font recognition, a special topic related to both texture and shape recognition. The main idea is that the user uploads an image of a text line, and the retrieval system will recognize the font used when printing the text. The output is a list with the most similar fonts in the database. The motivation for creating such a system, or search engine, is that choosing an appropriate font for a text can be both difficult and very time consuming. One way to speed up the selection procedure is to be inspired by others. But if we find a text written with a font that we want to use, we have to find out if this font is available in some database. Examples are the collection on our own personal computer, or a database owned by a company selling fonts, or a database with free fonts. The presented font recognition system can be applied on any font collection. However, the intended usage is the search in very large font databases, containing several thousand fonts. Proposed methods are mainly developed for the 26 basic characters in the Latin alphabet (basically the English alphabet).

The second part of the thesis will explore how to use color emotions in CBIR. Color emotions can be described as emotional feelings evoked by single colors or color combinations. They are typically expressed with semantic words, such as "warm", "soft", "active", etc. The motivation for this research is to include high level semantic information, such as emotional feelings, in image classification and image retrieval systems. Emotional responses based on objects, faces etc. are often highly individual, and therefore one has to be careful when including them in classification of general image databases. However, the emotional response evoked by color content, as part of the color perception process, is more universal. Combining color emotions with CBIR enables us to discover new ways of searching for images, with a strong connection to high

\(^1\)http://media-vibrance.itn.liu.se/
level semantic concepts. The presented method can be used standalone, or in combination with other methods.

1.3 Originality and Contributions

The presentation of the font retrieval system is the only publicly available description of font recognition for very large font databases. We evaluate our methods searching a database of 2763 fonts. To the best of our knowledge this database is several times larger than any database used in previous work. We describe how well-known methods can be improved and combined in new ways to achieve fast and accurate recognition in very large font databases. The search engine is accompanied by a tool for visualizing the entire database. Another novel contribution is that the retrieval method has been implemented in a publicly available search engine for free fonts, which provides the opportunity to gather user statistics and propose improvements based on real-world usage.

In the second part of the thesis we present a novel approach in Content Based Image Retrieval, incorporating color emotions. The method is a contribution in the new but upcoming research field of emotion or aesthetic based image retrieval. Unique for the retrieval method presented in this thesis is that a comprehensive user study is incorporated in the overall presentation.

Parts of the material presented in this thesis have contributed to other publications. Two per-reviewed conference papers have been published ([81] and [82]), and two journal papers are currently under review ([80] and [79]).

1.4 Outline of the Thesis

The thesis is organized as follows. In the next chapter we start with a short overview of the past and current research in Content Based Image Retrieval, together with future prospects. The chapter is concluded with an overview of commercial implementations that in some way take image content into consideration.

Chapter 3 will describe our visual search engine, or recognition system for fonts. We start with an introduction and a discussion about related research, followed by a description and evaluation of the proposed recognition method. Then some experiments in visualizing the entire font database are presented. The chapter ends with the implementation of an online font search engine for free fonts, together with user statistics gathered during a period of 20 months.

In chapter 4 we present an image retrieval method based on color emotions. Introduction and background is followed by fundamentals in color emotion research and psychophysical experiments. Then the proposed method using color emotions in image retrieval is presented. The subsequent section contains an extensive psychophysical evaluation of color emotions for multi-colored images,
with particular focus on the proposed retrieval method. We end the chapter with a description of an online implementation, together with some user statistics gathered during a period of 14 months.

Conclusions are presented in chapter 5, and ideas for future work are discussed in chapter 6.
Chapter 2

Image Retrieval: Past, Present and Future

Contributions presented in this thesis are all related to Content Based Image Retrieval. In this chapter we describe a few aspects of past and current research in CBIR, together with future prospects. We mainly follow the presentation in Datta et. al. [13]. The chapter is concluded with an overview of commercial implementations.

2.1 Research Interests

We present a short overview discussing how CBIR methods have evolved in the research community over the years, providing references to some contributions. This is not a complete survey. Instead we mainly follow Datta et. al. [13], from now on referred to as "Datta et. al.", where the reader can find an extensive list of references. Apart from the list of references, the survey discusses general but important questions related to image retrieval, like user intent, the understanding of the nature and scope of image data, different types of queries and visualization of search results.

Datta et. al. define CBIR as "any technology that helps to organize digital picture archives by their visual content". A fundamental concept and difficulty in CBIR is the semantic gap, which usually is described as the distance between the visual similarity of low-level features, and the semantic similarity of high-level concepts. Whether the visual or the semantic similarity is the most important depends on the situation. If we, for instance, want to retrieve images of any sports car, a measurement of the semantic similarity is preferred, where the phrase ”sports car” can contain a rather loose definition of any car that seems to be built for speed. If, instead, we are interested in a Shelby Cobra
Daytona Coupe (a unique sports car built in 1964-65), the importance of the exact visual similarity will increase. Consequently, the preference for a visual or semantic similarity depends on the query and the application in mind.

To illustrate the growth of CBIR research, Datta et. al. have conducted an interesting exercise. Using Google Scholar and the digital libraries of ACM, IEEE and Springer, they searched for publications containing the phrase "Image Retrieval" within each year from 1995 to 2005. The findings show a roughly exponential growth in interest in image retrieval and closely related topics. The web page http://wang.ist.psu.edu/survey/analysis/ contains more bibliometrical measurements. They, for instance, queried Google Scholar with the phrase "image retrieval" together with other CBIR-related phrases, to find trends in publication counts. The research area with the strongest increase in publication counts seems to be Classification/Categorization. However, all areas, except Interface/Visualization, have increased considerably over the years.

We continue the overview with a brief description of the early years of CBIR (prior year 2000). An often cited survey covering this period of time is Smeulders et. al. [77]. They separated image retrieval into broad and narrow domains, depending on the purpose of the application. A narrow domain typically includes images of limited variability, like faces, airplanes, etc. A broad domain includes images of high variability, for instance large collections of images with mixed content downloaded from the Internet. The separation into broad and narrow domains is today a well-recognized and widely used distinction.

The early years of image retrieval were dominated by low level processing and statistical measurements, typically focusing on color and texture, but also on shape signatures. An important contribution is the use of color histograms, describing the distribution of color values in an image. Among the earliest use of color histograms was that in Swain and Ballard [85]. As an enhancement, Huang et. al. [35] proposed the color correlogram that take into consideration the spatial distribution of color values. Manjunath and Ma [49] focused on shape extraction and used Gabor filters for feature extraction and matching. Research findings were (as today) often illustrated in public demo search engines. A few with high impact factor were IBM QBIC [21], Pictoseek [27], VisualSEEK [78], VIRAGE [29], Photobook [64], and WBIS [95].

Datta et. al. divide current CBIR technology into two problem areas: (a) how to mathematically describe an image, and (b) how to assess the similarity between images based on their descriptions (also called signatures). They find that in recent years the diversity of image signatures has increased drastically, along with inventions for measuring the similarity between signatures. A strong trend is the use of statistical and machine learning techniques, mainly for clustering and classification. The result can be used as a pre-processing step for image retrieval, or for automatic annotation of images. An example of the later is the ALIPR (Automatic Linguistic Indexing of Pictures - Real-
2.1 Research Interests

Time) system, described by Li and Wang [45, 46]. Moreover, images collected from the Internet have become popular in clustering, mainly because of the possibility to combine visual content with available metadata. Examples can be found in Wang et. al. [98] and Gao et. al. [23].

Datta et. al. continue with a trend from the beginning of this decade, the use of region-based visual signatures. The methods have improved alongside advances in image segmentation. An important contribution is the normalized cut segmentation method proposed by Shi and Malik [75]. Similarly, Wang et. al. [94] argue that segmented or extracted regions likely corresponds to objects in the image, which can be used as an advantage in the similarity measurement. Worth noticing is that in this era, CBIR technologies started to find their way into popular applications and international standards, like the insertion of color and texture descriptors in the MPEG-7 standard (see for instance Manjunath et. al. [50]).

Another development in the last decade is the inclusion of methods typically used in computer vision into CBIR, for instance the use of salient points or regions, especially in local feature extraction. Compared to low-level processing and statistical measurements, the extraction and matching of local features etc. tend to be computational more expensive. The shortage of computer capacity in the early years of CBIR probably delayed the use of local feature extraction in image retrieval. Datta et. al. have another explanation. They believe the shift towards local descriptors was activated by "a realization that the image domain is too deep for global features to reduce the semantic gap". However, as described later in this section, a contradicting conclusion is presented by Torralba et. al. [88].

Also texture features have long been studied in computer vision. One example applied in CBIR is texture recognition using affine-invariant texture feature extraction, described by Mikolajczyk and Schmid [52]. Another important feature is the use of shape descriptors. The recent trend is that global shape descriptors (e.g. the descriptor used in IBM QBIC [21]) are replaced by more local descriptors. Recently, the use of local invariants, such as interest points and corner points, are being used in image retrieval, especially object-based retrieval. The already mentioned paper about affine-invariant interest points by Mikolajczyk and Schmid [52] is a good example, together with Grauman and Darrell [28], describing the matching of images based on locally invariant features. Another well-known method for extracting invariant features is the Scale-invariant feature transform (SIFT), presented by Lowe [47]. Such local invariants were earlier mainly used in for instance stereo matching. Another popular approach is the use of bags of features, or bags of keypoints, as in Csurka et. al. [11]. A bag of keypoints is basically a histogram of the number of occurrences of particular image patterns in a given image. Datta et. al. conclude that in this domain we have a similar shift towards local descriptors.

We briefly mention the topic of relevance feedback. The feedback process
typically involves modifying the similarity measure, the derived image features, or the query based on feedback from the user. The paper by Rui et al. [67] is often mentioned as one of the first attempts. For readers interested in relevance feedback we refer to the overview by Zhou and Huang [104].

Another topic we briefly mention is multimodal retrieval, which can be described as retrieval methods combining different media, for instance image content, text and sound. A good example is video retrieval, where the attention from researchers has increased dramatically in recent years. A major part of the increased popularity can most likely be devoted to TRECVID [76], an annual workshop in video retrieval where participants can evaluate and compare their retrieval methods against each other. Similar competitions, or test-collections, focusing on image retrieval tasks are also becoming popular. One example is the PASCAL Visual Object Classes Challenge [19], where the goal is to recognize objects from a number of visual object classes in realistic scenes. Another example is the CoPhIR (Content-based Photo Image Retrieval) test-collection (see http://cophir.isti.cnr.it/), with scalability as the key issue. The CoPhIR collection now contains more than 100 million images. Finally, we mention ImageCLEF - The CLEF Cross Language Image Retrieval Track (see for instance [8] or http://imageclef.org/), an annual event divided into several retrieval tasks, like photographic, medical, etc.

What the future holds for CBIR is a challenging question. Datta et al. lists a few topics of the new age. They start with the combination of words and images, and foresee that “the future of real-world image retrieval lies in exploiting both text- and content-based search technologies”, continuing with “there is often a lot of structured and unstructured data available with the images that can be potentially exploited through joint modeling, clustering, and classification”. A related claim is that research in the text domain has inspired the progress in image retrieval, particularly in image annotation. The ALIPR system (Li and Wang [45, 46]), mentioned earlier, is a recent example that has obtained a lot of interest from both the research community and the industry. Datta et al. conclude that automated annotation is an extremely difficult issue that will attract a lot of attention in upcoming research. Another topic of the new age is to include aesthetics in image retrieval. Aesthetics can relate to the quality of an image, but more frequently to the emotions a picture arouses in people. The outlook of Datta et al. is that “modeling aesthetics of images is an important open problem”, that will add a new dimension to the understanding of images. They presented the concept “personalized image search”, where the subjectivity in similarity is included into image similarity measures by incorporating “ideas beyond the semantics, such as aesthetics and personal preferences in style and content”. The topic of emotion based image retrieval is further discussed in a later section of this thesis. Other hot topics mentioned by Datta et al. are images on the Internet, where the usually available meta data can be incorporated in the retrieval task, and the possibility
2.2 Commercial Implementations

A selection of some large and interesting commercial services for image retrieval is listed below. The selection only includes services that in some way take image content into consideration. Two of the leading players in image search are Google and Picsearch:

**Google** (www.google.com): Probably the best known Internet search engine. Google’s Image search is one of the largest, but retrieval is mainly based on text, keywords, etc. However, they are working on search options that are based on image content. Earlier they made it possible to search for images containing faces, images with news content, or images with photo content. Recently other options were introduced. They added the feature of searching for clip art or line drawings. What Google intend to do with their Image search in the future is hard to know, but they are certainly interested in content based retrieval, see for instance the paper by Jing and Baluja [37]. Another tool confirming the interest in CBIR is the Google Image Labeler, where users are asked to help Google improve the quality of search results by labeling images.

**Picsearch** (www.picsearch.com): Another big player in image retrieval, with an image search service containing more than three billion pictures. According to recent research (see [88]), Picsearch presents higher retrieval
accuracy (evaluated on hand-labeled ground truth) than many of their competitors, for instance Google. So far, the only content based retrieval mode included in their search engine (except color or black&white images) is an opportunity to choose between ordinary images and animations.

Examples of other search engines providing similar image search services are Cydral (www.cydral.com) and Exalead (www.exalead.com/image), both with the possibility to search for color or gray scale images, and images containing faces. Common for search engines presented above is that their focus is on very large databases containing uncountable numbers of image domains. As an alternative, the following retrieval services are focusing on narrow image domains or specific search tasks.

**Riya / Like.com** (www.riya.com / www.like.com): Riya was one of the pioneers introducing face recognition in a commercial application. Initially they created a public online album service with a face recognition feature, providing the opportunity to automatically label images with names of persons in the scene (the person must have been manually labeled in at least one photo beforehand). After labeling it is possible to search within the album for known faces. After the success with the online album service Riya moved on to other tasks, and are now developing the Like.com visual search, providing visual search within aesthetically oriented product categories (shoes, bags, watches, etc.). The Riya website has also moved on to become a broader visual search engine, handling for instance both people and objects. However, it is unclear to what extent the Riya visual search is based on image content.

**Polar Rose** (www.polarrose.com): Polar Rose is another pioneer in commercialization of face recognition. They target both photo sharing and media sites, and private users. The later through a browser plugin, which enables users to name people they see in public online photos. Then the Polar Rose search engine can be used for finding more photos of a person.

**Pixsta** (www.pixsta.com): Pixsta is a close competitor to Like.com, also working on visual similarity between product images, like shoes, jewellery, bags, etc.

An upcoming niche in image retrieval is to identify and track images for instance on the Internet, even if images have been cropped or modified. The intended usage is for instance to prevent others from using copyrighted images. One of the most interesting companies is Idée Inc.

**Idée Inc.** (http://ideinc.com): Idée develops software for both image identification and other types of visual search. Their product TinEye allows the user to submit an image, and the search engine finds out where
and how that image appears on the Internet. The search engine can handle both cropped and modified images. The closely related product PixID is also incorporating printed material in the search. Moreover, they have a more general search engine called Piximilar that uses (according to Idée) color, shape, texture, luminosity, complexity, objects and regions to perform visual search in large image collections. The input can be a query image, or colors selected from a color palette. In addition, Idée is currently working on a product called TinEye Mobile, which allows the user to search for commercial products by using a mobile phone’s camera. It will be interesting to see the outcome of that process.

Two other search engines with similar tracking functions are TrackMyPicture (www.trackmypicture.com) and Photopatrol (www.photopatrol.eu). However, with a Swedish and a German webpage respectively, they target rather narrow consumer groups.

One of the contributions presented in this thesis is an emotion based image search engine. Similar search strategies have not yet reached commercial interests, with one exception, the Japanese emotional visual search engine EVE (http://amanaimages.com/eve/). The search engine seems to be working with the emotion scales soft - hard, and warm - cool. However, the entire user interface is written in Japanese, making it impossible for the author of this thesis to further investigate the search engine.

EVE ends this summary of commercial implementations using content based image retrieval. However, the list is far from being complete. For instance, two of all the image services not included in the summary are the image provider Matton (www.matton.com), and the first large Internet search engine Altavista (www.altavista.com), both interested in content based retrieval. The number of commercial implementations are steadily increasing, thus within a few months the list will be even longer. It is interesting to notice that current implementations are focusing either on small image domains or specific search tasks (like faces, shoes, etc.), or on simple search modes in broader domains and large databases (like ordinary images vs. animations).
Chapter 3

Font Retrieval

This chapter will describe our attempts in creating a visual search engine, or recognition system for fonts. The basic idea is that the user submits an image of a text line, and the search engine tells the user the name of the font used when printing the text. Proposed methods are developed for the 26 basic characters in the Latin alphabet (basically the English alphabet). A system for visualizing the entire font database is also proposed.

3.1 Introduction

Choosing an appropriate font for a text can be a difficult problem since manual selection is very time consuming. One way to speed up the selection procedure is to be inspired by others. But if we find a text written with a font that we want to use, we have to find out if this font is available in some database. Examples of databases can be the database on our own personal computer, or a database owned by a company selling fonts, or a database with free fonts. In the following a search engine for font recognition is presented and evaluated. The intended usage is the search in very large font databases, but the proposed method can also be used as a pre-processor to Optical Character Recognition. The user uploads an image of a text, and the search engine returns the names of the most similar fonts in the database. Using the retrieved images as queries, the search engine can be used for browsing through the database. The basic workflow of the search engine is illustrated in Fig. 3.1. After pre-processing and segmentation of the input image, a local approach\footnote{In font recognition terminology, a local approach typically extracts features for words or single characters, whereas a global approach usually extracts features for a text line or a block of text.} is used, where features are calculated for individual characters.
Figure 3.1: Structure of the search engine. The input is an image of a text line, typically captured by a scanner or a digital camera. A pre-processing unit will rotate the text line to a horizontal position, and perform character segmentation. Then the user will assign letters to images that will be used as input to the recognition unit, and the recognition unit will display a list of the most similar fonts in the database.

Our solution, which we call Eigenfonts, is based on eigenimages calculated from edge filtered character images. Both the name and the method are inspired by the Eigenfaces method, used in the context of face recognition. Improvements, mainly in the pre-processing step, for the ordinary eigen-method are introduced and discussed. Although the implementation and usage of eigenimages is rather simple and straight forward, we will show that the method is highly suitable for font recognition in very large font databases. Advantages of the proposed method are that it is simple to implement, features can be computed rapidly, and descriptors can be saved in compact feature vectors. Since the intended usage is the search in very large font databases the compact descriptors are important for short response times. Other advantages are that the method shows robustness against various noise levels and image quality, and it can handle both overall shape and finer details. Moreover, training is not required, and new fonts can be added without re-building the system.

We use three font databases: original character images rendered directly from the font files, a database where characters from these fonts were printed and then scanned, and a third database containing characters from unknown fonts (also printed and scanned). To resemble a real life situation, the search engine is evaluated with the printed and scanned versions of the images. A few examples of images from the original database can be seen in Fig. 3.2. In total, the database contains 2763 different fonts. To the best of our knowledge this is the largest font database used in publicly available descriptions of font search engines. The retrieval accuracy obtained with our method is comparable and often better than the results described based on smaller databases. Our evaluation shows that for 99.1% of the queries, the correct font name can be
3.2 Background

Figure 3.2: Examples of character a

found within the five best matches. In the following chapters when we refer to 'testdb1' or 'testdb2', we use collections of images from the printed and scanned test database. More about the font databases can be found in Section 3.6. The current version of the search engine contains fonts for the English alphabet only. The findings are also implemented in a publicly available search engine for free fonts.²

The rest of this chapter is organized as follows: In the next section we present the background for this research, followed by a section describing the pre-processing step, including skew estimation and correction, and character segmentation. Then the basic design of the search engine is described in Section 3.4. This includes a description of the eigenfonts method together with the most important design parameters. More parameters, mainly for fine tuning the system, are discussed in Section 3.5. In Section 3.6, we summarize the search engine and evaluate the overall performance. This will also include a description of the font databases, and experiments concerning the quality of the search image. In Section 3.7 we show some attempts in visualizing the entire font database. The final online implementation of the search engine is described in Section 3.8, together with user statistics gathered during a period of 20 months. Then we draw conclusions and discuss the result in Section 5.1, and future work can be found in Section 6.1.

3.2 Background

There are two major application areas for font recognition or classification; as a tool for font selection, or as a pre-processor for OCR systems. A difference between these areas is the typical size of the font database. In a font selection task, we can have several hundreds or thousands of fonts, whereas in OCR systems it is usually sufficient to distinguish between less than 50 different fonts. As mentioned earlier, the database used in this study contains 2763 different fonts. To our knowledge this database is several times larger than any database used in previous work. The evaluation is made with the same database, making it harder to compare the result to other results. However, since the intended usage is in very large databases, we believe it’s important to make the evaluation with a database of comparable size.

The methods used for font recognition can roughly be divided into two main categories: either local or global feature extraction. The global approach

²http://media-vibrance.itn.liu.se/ffont/
Font Retrieval

typically extracts features for a text line or a block of text. Different filters, for instance Gabor filters, are commonly used for extracting the features. The local approach sometimes operates on sentences, but more often on words or single characters. This approach can be further partitioned into two sub-categories: known or unknown content. Either we have a priori knowledge about the characters, or we don’t know which characters the text is composed of. In this research we utilize a local approach, with known content.

For many years font recognition research was dominated by methods focusing on the English or Latin alphabet (with minor contributions focusing on other alphabets, for instance the Arabic alphabet [55], and the South Asian script Sinhala [65][66]). In recent years font recognition for Chinese characters has grown rapidly. However, there are major differences between those languages. In the English alphabet we have a rather limited number of characters (basically 26 upper case, and 26 lower case characters), but a huge number of fonts. There is no official counting, but approximately 50 000 – 60 000 unique fonts (both commercial and non commercial) can be found for the English, or closely related alphabets. The huge number of fonts is probably caused by the limited number of characters, making it possible to create new fonts within feasible time. For Chinese characters, the number of existing fonts is much fewer, but in the same time, the number of possible characters is much, much larger. The effect is that for Chinese characters, font recognition based on a global approach is often more suitable than a local approach, since you don’t need to know the exact content of the text. For the English alphabet, one can take advantage of the smaller number of characters and use a local approach, especially if the content of the text is known. However, there is no standard solution for each alphabet. Methods can be found that are using exactly the opposite approach, for both Chinese and English characters. Some researchers are debating whether font recognition for Chinese or English characters is the most difficult task. For instance Yang et. al. [102] claims that "For Chinese texts, because of the structural complexity of characters, font recognition is more difficult than those of western languages such as English, French, Russian, etc". However, we believe several researchers will disagree with such a statement. It’s probably not fair to compare recognition accuracy for Chinese respectively English fonts, but results obtained are rather similar, indicating that none of the alphabets are much easier than the other. Moreover, research concerning completely different alphabets, like the Arabic or Persian alphabet, report similar results. In the remaining part of this section we give a summary of past font recognition research. The focus will be on methods working with the English alphabet (since our own research has the same focus), but font recognition for Chinese characters will also be discussed.

A good overview can be found in Sandra Larsson’s master thesis [41], carried out at ITN, Linköping University. Her goal was to investigate the possibility to design a font search engine by evaluating if traditional shape descriptors can
be applied in font recognition. Focus was entirely on the English alphabet. A prototype font search engine was developed to evaluate the performance of the most promising descriptors. A majority of the descriptors where eliminated early in the evaluation process. For instance, standalone use of simple shape descriptors, like perimeter, shape signature, bending energy, area, compactness, orientation, etc., were found inappropriate. They might work in practice if they are combined with other descriptors, but due to time limitations this was never studied in detail.

Among more advanced contour based methods, Fourier descriptors were of highest interest. By calculating Fourier descriptors for the object boundary, one can describe the general properties of an object by the low frequency components and finer details by the high frequency components. While Fourier descriptors have been of great use in optical character recognition and object recognition, they are too sensitive to noise to be able to capture finer details in a font silhouette. Consequently, Fourier descriptors are not suitable for the recognition of fonts. The same reasoning can be applied to another popular tool frequently used for representing boundaries, the chain code descriptor (see Freeman [22] for an early contribution). Experiments showed similar drawbacks as with Fourier descriptors. For printed and scanned character images, the noise sensitivity is too high. Another drawback with contour-based methods is that they have difficulties handling characters with more than one contour. Thus at the end of the master thesis, the focus shifted towards region-based methods.

The most promising approach involving region-based methods are originating from the findings of Sexton et. al. [71][70], where geometric moments are calculated at different levels of spatial resolution, or for different image regions. At lower levels of resolution, or in sub-regions, finer details can be captured, and the overall shape can be captured at higher levels of resolution. A common approach is some kind of tree-decomposition, where the image is iteratively decomposed into sub-regions. One can for instance split the image, or region, into four new regions of equal size (a quad-tree). Another approach evaluated in the thesis is to split according to the centre of mass of the shape (known as a kd-tree decomposition), resulting in an equal number of object (character) pixels in each sub-region.

One of the methods investigated in Larsson [41], that was included in the final evaluation, involves a four level kd-tree decomposition. Each split decomposes the region into two new regions based on the centroid component. The decomposition alternates between a vertical and horizontal split, resulting in totally 16 sub-regions. For each sub-region, three second order normalized central moments are saved, together with the coordinate of the centroid normalized by the height or width of the sub-region. Also, the aspect ratio of the entire image is added to the feature vector. For evaluating the recognition performance, different test sets and test strategies were developed. The one mentioned here is similar to the one utilized later in this section. To resemble
a real life situation, test characters are printed with a regular office printer, and then scanned with an ordinary desktop scanner. The best metric for comparing feature vectors were found to be the Manhattan distance. Using this metric, the probability that the best match is the correct font is 81.0%, and the probability to find the correct font within the five best matches is 95.7%. However, in Larsson [41], results obtained from the printed/scanned evaluation only involve the mean score for characters ‘a’ and ‘b’ (to be remembered when comparing with results from other studies). If all characters from the English alphabet are included in the mean score the retrieval accuracy usually decreases.

To our knowledge, only one search engine is publicly available for font selection or identification: WhatTheFont. The engine is commercially operated by MyFonts.com, a company focusing on selling fonts mainly for the English alphabet. A local method that seems to be the starting point for WhatTheFont can be found in Sexton et. al. [71]. They identify the font from a selection of characters by comparing features obtained from a hierarchical abstraction of its binary image at different resolutions. Images are decomposed into smaller images, and geometrical properties are then calculated from each sub-image based on recursive decomposition. Experiments were carried out on different decomposition strategies and on the weighting and matching of feature vectors. Their database consisted of 1300 unique uppercase glyphs with 50 different fonts rendered at 100 pts/72 dpi. Testing was carried out with 100 randomly selected images scaled to 300% and blurred. The best performance was achieved with a non-weighted kd-tree metric at 91% accuracy for a perfect hit. We are not aware of newer, publicly available, descriptions of improvements that are probably included in the commercial system. However, in an article about recognition of mathematical glyphs Sexton et. al. [70] describe and apply their previous work on font recognition.

Another example of a local approach for the English alphabet is presented by Östürk et. al. [83], describing font clustering and cluster identification in document images. They evaluated four different methods (bitmaps, DCT coefficients, eigencharacters, and Fourier descriptors), and found that they all result in adequate clustering performance, but the eigenfeatures result is the most parsimonious and compact representation. They used a rather limited set of fonts, since documents like magazines and newspapers usually do not have a large variety of fonts. Their goal was not primarily to detect the exact font; instead fonts were classified into clusters.

Lee and Jung [43] proposed a method using non-negative matrix factorization (NMF) for font classification. They used a hierarchical clustering algorithm and Earth Mover Distance (EMD) as distance metric. Experiments are performed at character-level, and a classification accuracy of 98% was shown for 48 different fonts. They also compare NMF to Principal Component Analysis, with different combinations of EMD and the $L_2$ distance metric (the combination of PCA and the $L_2$ metric has similarities with the approach presented in...
3.2 Background

This thesis). Their findings show that the EMD metric is more suitable than the $L_2$-norm, otherwise NMF and PCA produce rather similar results, with a small advantage for NMF. The drawback with the EMD metric is that the computational cost is high, making it less suitable for real-time implementations. The authors favor NMF since they believe characteristics of fonts are derived from parts of individual characters, compared to the PCA approach where captured characteristics to a larger extent describe the overall shape of the character. However, it can be discussed whether the motivation agrees with human visual perception. If humans are evaluating font likeness, the overall shape is probably a major component in the likeness score. A similar NMF-approach is presented by Lee et. al. [42].

Another local approach was proposed by Jung et. al. [38]. They presented a technique for classifying seven different typefaces with different sizes commonly used in English documents. The classification system uses typographical attributes such as ascenders, descenders and serifs extracted from word images as input to a neural network classifier.

Khoubyari and Hull [39] presented a method where clusters of words are generated from document images and then matched to a database of function words from the English alphabet, such as "and", "the" and "to". The font or document that matches best provides the identification of the most frequent fonts and function words. The intended usage is as a pre-processing step for document recognition algorithms. The method includes both local and global feature extraction. A method with similar approach is proposed by Shi and Pavlidis [74]. They use two sources for extracting font information: one uses global page properties such as histograms and stroke slopes, the other one uses information from graph matching results of recognized short words such as "a", "it" and "of". This approach focuses on recognizing font families.

We continue with a few methods using a global approach. A paper often mentioned in this context is Zramdini and Ingold [106]. Global typographical features are extracted from text images. The method aims at identifying the typeface, weight, slope and size of the text, without knowing the content of the text. Totally eight global features are combined in a Bayesian classifier. The features are extracted from classification of connected components, and from various processing of horizontal and vertical projection profiles. For a database containing 280 fonts, a font recognition accuracy of 97% is achieved, and the authors claim the method is robust to document language, text content, and text length. However, they consider the minimum text length to be about ten characters.

An early contribution to font recognition is Morris [54], who considered classification of typefaces using spectral signatures. Feature vectors are derived from the Fourier amplitude spectra of images containing a text line. The method aims at automatic typeface identification of OCR data. The method shows fairly good results when tested on 55 different fonts. However, they are
only using synthetically derived noise-free images in their experiments. Images containing a lot of noise, which is common in OCR applications, will probably have a strong influence on the spectral estimation.

Also Avilés-Cruz et. al. [1] use an approach based on global texture analysis. Document images are pre-processed to uniform text blocks, and features are extracted using third and fourth order moments. Principal Component Analysis reduces the number of dimensions in the feature vector, and classification uses a standard Bayes classifier. 32 commonly used fonts in Spanish texts are investigated in the experiments. Another early attempt was made by Baird and Nagy [2]. They developed a self-correcting Bayesian classifier capable of recognizing 100 typefaces, and demonstrated significant improvements in OCR-systems by utilizing this font information.

As mentioned earlier, font recognition for Chinese characters is a rapidly growing research area. An approach using local features for individual characters is presented by Ding et. al. [17]. They recognize the font from a single Chinese character, independent of the identity of the character. Wavelet features are extracted from a character images. After a Box-Cox transformation and LDA (Linear Discriminant Analysis) process, discriminating features for font recognition are extracted, and a MQDF (Modified Quadric Distance Function) classifier is employed to recognize the font. Evaluation is made with two databases, containing totally 35 fonts, and for both databases the recognition rates for single characters are above 90%.

Another example of Chinese font recognition, including both local and global feature extraction, is described by Ha and Tian [30]. The authors claim that the method can recognize the font of every Chinese character. Gabor features are used for global texture analysis to recognize a pre-dominant font of a text block. The information about the pre-dominant font is then used for font recognition of single characters. In a post-processing step, errors are corrected based on a few typesetting laws, for instance that a font change usually takes place within a semantic unit. Using four different fonts, a recognition accuracy of 99.3% can be achieved. A similar approach can be found in an earlier paper by Miao et. al. [51], written by partly the same authors.

In Yang et. al. [102], Chinese fonts are recognized based on Empirical Mode Decomposition, or EMD (should not be confused with the distance metric with the same abbreviation, called Earth Mover Distance). The proposed method is based on the definition of five basic strokes that are common in Chinese characters. These strokes are extracted from normalized text blocks, and so-called stroke feature sequences are calculated. By decomposing them with EMD, Intrinsic Mode Functions (IMFs) are produced. The first two, so-called stroke high frequency energies are combined with the five residuals, called stroke low frequency energies, to create a feature vector. A weighted Euclidean distance is used for searching in a database containing 24 Chinese fonts, receiving an average recognition accuracy of 97.2%. The authors conclude that the pro-
The proposed method is definitely suitable for Chinese characters, but they believe the method can be applicable to other alphabets where basic strokes can be defined properly.

In the global approach by Zhu et. al. [105], text blocks are considered as images containing specific textures, then Gabor filters are used for texture identification. With a weighted Euclidean distance (WED) classifier, an overall recognition rate of 99.1% is achieved for 24 frequently used Chinese fonts and 32 frequently used English fonts. The authors conclude that their method is able to identify global font attributes, such as weight and slope, but less appropriate for distinguishing finer typographical attributes. Similar approaches with Gabor filters can be found in Ha et. al. [31] and Yang et. al. [101].

Another method evaluated for both English and Chinese characters is presented by Sun [84]. It is a local method operating on individual words or characters. The characters are converted to skeletons, and font-specific stroke templates (based on junction points and end points) are extracted. Templates are classified to belong to different fonts with a certain probability, and a Bayes decision rule is used for recognizing the font. Twenty English fonts and twenty Chinese fonts are used in the evaluation process. The recognition accuracy is rather high, especially for high quality input images, but the method seems to be very time consuming and consequently not suitable for larger databases or implementations with real-time requirements.

As an example of Arabic font recognition we refer to Moussa et. al. [55]. In summary, they present a non-conventional method using fractal geometry on global textures. For nine different fonts, they receive a recognition accuracy of 94.4%.

In conclusion, for many years font recognition was dominated by methods focusing on the English alphabet, but recently, research concerning Chinese characters has increased considerably. For Chinese characters (and similar scripts), the recognition is often based on a global approach using texture analysis. For the English alphabet (and similar alphabets), a local approach operating on single characters or words is more common, especially when a rough classification is not accurate enough.

### 3.3 Character Segmentation

In this section two pre-processing steps are described: skew estimation and correction, and character segmentation. A survey describing methods and strategies in character segmentation can be found in [6]. The final segmentation method proposed below is based on a combination of previously presented methods.
3.3.1 Skew estimation and correction

Since many of the input text lines will be captured by a scanner or a camera, characters will often be slightly rotated. If the skew angle is large, it will influence the performance of both the character segmentation and the database search. An example of a skewed input image can be seen in Fig. 3.3. Detection and correction of skewed text lines consists of the following steps:

1. Find the lower contour of the text line (see Fig. 3.4).
2. Apply an edge filter to detect horizontal edges.
3. Use the Hough-transform to find near-horizontal strokes in the filtered image (see Fig. 3.5).
4. Rotate the image by the average angle of near-horizontal strokes.

3.3.2 Character segmentation

Since the search technology is based on individual characters, each text line needs to be segmented into sub-images. This approach is usually called dissection, and a few examples of dissection techniques are:

1. **White space and pitch**: Uses the white space between characters, and the number of characters per unit of a horizontal distance (limited to fonts with fixed character width).
2. **Projection analysis**: The vertical projection (also known as the vertical histogram) can be used for finding spaces between characters and strokes.
3. **Connected component analysis**: Uses connected black regions for segmentation.

Here we use vertical projection (an example in Fig. 3.6) and the upper contour profile (an example in Fig. 3.7). Different strategies using vertical projection and the contour profiles have been examined. These are:

**Second derivative to its height** [6]
Segmentation decisions are based on the second derivative of the vertical projection to its height. The result can be seen in Fig. 3.8 and Fig. 3.9.

**Peak-to-valley function** [48]
This is a function designed for finding breakpoint locations within touching characters.

\[
pv(x) = \frac{V(lp) - 2 \times V(x) + V(rp)}{V(x) + 1} \quad (3.1)
\]
Character segmentation is extraordinarily interesting.

Figure 3.3: An input image, rotated 0.5 degrees clockwise.

Figure 3.4: Lower contour of the text line.

Figure 3.5: Hough-transform for finding lines in a filtered lower contour image.

Figure 3.6: Vertical projection of the text line.

Figure 3.7: Upper contour of the text line.

Figure 3.8: Second derivative to its height.

Figure 3.9: Segmentation using the derivative and the second derivative to its height.
Where $V(x)$ is the vertical projection function, $x$ is the current position, and $lp$ and $rp$ are peak locations on the left and right side. The result can be seen in Fig. 3.10 and Fig. 3.11. A maximum in the Peak-to-valley function is assumed to be a segmentation point.

**Break-cost** [91]

The break-cost is defined as the number of pixels in each column after an AND operation between neighboring columns. Candidates for break positions are obtained by finding local minima in a smoothed break-cost function. The result can be seen in Fig. 3.12 and Fig. 3.13.

**Contour extraction** [6]

The upper and lower contours are analyzed to find slope changes which may represent possible minima in a word. In this work, the second derivative and the second derivative to its height of the upper contour are used. The result is shown in Fig. 3.14, Fig. 3.15 and Fig. 3.16.

The proposed method is a hybrid approach, where weighted combinations of the above methods are used. The final segmentation result for a regular and an italic text line can be seen in Fig. 3.17 and Fig. 3.18. As shown in the figures, the methods do not always deliver a perfect result. Improvements are necessary, especially when italic fonts are used. However, the segmentation result usually contain enough individual characters to be used by the search engine.

### 3.4 Basic Search Engine Design

In this section we describe the theoretical background of the proposed search engine, starting with a description of the eigenfonts method. Then we discuss
3.4 Basic Search Engine Design

Figure 3.13: Segmentation using the break-cost function.

Figure 3.14: The second derivative of the upper contour.

Figure 3.15: The second derivative to its height of the upper contour.

Figure 3.16: Segmentation result from contour based methods.

Figure 3.17: Final segmentation result for a regular text line.

Figure 3.18: Final segmentation result for an italic text line.
important design parameters, like alignment and edge filtering of character images.

3.4.1 Eigenfonts basics

We denote by $I(char,k)$ the $k^{th}$ image (font) of character $char$, reshaped to a column vector. Images of characters from different fonts are quite similar in general (pixel values are not randomly distributed); therefore images can be projected to a subspace with lower dimensions. The principal component analysis (or Karhunen-Loeve expansion) reduces the number of dimensions, leaving dimensions with highest variance. Eigenvectors and eigenvalues are computed from the covariance matrix containing all fonts of each character in the original database. The eigenvectors corresponding to the K highest eigenvalues describe a low-dimensional subspace on which the original character images are projected. The coordinates in this subspace are stored as the new descriptors. The first five eigenimages for character 'a' can be seen in Fig. 3.19.

The proposed method works as follows: The 2-D images are reshaped to column vectors, denoted by $I(char,k)$ (where $I(a,100)$ is the 100$^{th}$ font image of character 'a', as described above). For each character we calculate the mean over all font images in the database

$$m(char) = \frac{1}{N} \sum_{n=1}^{N} I(char,n)$$  \hspace{1cm} (3.2)

where $N$ is the number of fonts. Sets of images will be described by the matrix

$$I(char) = (I(char,1),...,I(char,N))$$  \hspace{1cm} (3.3)

For each character in the database, the corresponding set (usually known as the training set) contains images of all fonts in the database. From each image in the set we subtract the mean and get

$$\tilde{I}(char,k) = I(char,k) - m(char)$$  \hspace{1cm} (3.4)
The covariance matrix is then given by

$$C(\text{char}) = \frac{1}{N} \sum_{n=1}^{N} \hat{I}(\text{char}, n) \hat{I}(\text{char}, n)' = AA'$$ (3.5)

where \(A = [\hat{I}(\text{char}, 1), \hat{I}(\text{char}, 2), ..., \hat{I}(\text{char}, N)]\). Then the eigenvectors \(u_k\), corresponding to the \(K\) largest eigenvalues \(\lambda_k\), are computed. If it is clear from the context we will omit the \(\text{char}\)-notation. The obtained eigenfonts (eigenimages) are used to classify font images. A new query image, \(Q\) (containing character \(\text{char}\)), is transformed into its eigenfont components by

$$\omega_k = u_k'(Q - m)$$ (3.6)

for \(k = 1, ..., K\). The weights, \(\omega_1, ..., \omega_K\), form a vector that describes the representation of the query image in the eigenfont basis. The vector is later used to find which font in the database describes the query image best. Using the eigenfonts approach requires that all images of a certain character are of the same size and have the same orientation. We also assume that they have the same color (black letters on a white paper background is the most obvious choice). We therefore apply the following pre-processing steps before we compute the eigenfont coefficients: 1) Grey value adjustments: If the character image is a color image, color channels are merged, then gray values are scaled to fit a pre-defined range. 2) Orientation and segmentation: If character images are extracted from a text line, the text line is rotated to a horizontal position prior to character segmentation (as described in section 3.3). 3) Scaling: Character images are scaled to the same size.

### 3.4.2 Character alignment and edge filtering

In the design process, the first thing to consider is the character alignment. Since we are using the eigenfonts method, images must have the same size, but the location of the character within each image can vary. We consider two choices: each character is scaled to fit the image frame exactly, or the characters are aligned according to their centroid value, leaving space at image borders. The later requires larger eigenfont images since the centroid value varies between characters, which will increase the computational cost. Experiments showed that frame alignment gives significantly better retrieval accuracy than centroid alignment.

Most of the information about the shape of a character can be found in the contour, especially in this case when shapes are described by black text on a white paper background. Exceptions are noise due to printing and scanning, and gray values in the contour due to anti-aliasing effects when images are rendered. Based on this assumption, character images were filtered with
different edge filters before calculating the eigenimages. The images used are rather small and therefore we use only small filter kernels (max 3 × 3 pixels). When several filters are used, the character image is filtered with each filter separately, and then filter results are added to create the final result. Experiments with many different filter kernels resulted in the following filters used in the final experiments (four diagonal filters, one horizontal, and one vertical edge filter):

\[
H = \begin{pmatrix}
-1 & -2 & -1 \\
0 & 0 & 0 \\
1 & 2 & 1
\end{pmatrix}, \quad V = \begin{pmatrix}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1
\end{pmatrix}
\]

\[
D_1 = \begin{pmatrix}
2 & 1 & 0 \\
1 & 0 & -1 \\
0 & -1 & -2
\end{pmatrix}, \quad D_2 = \begin{pmatrix}
0 & 1 & 2 \\
-1 & 0 & 1 \\
-2 & -1 & 0
\end{pmatrix}
\]

\[
D_3 = \begin{pmatrix}
-1 & 0 \\
0 & 1
\end{pmatrix}, \quad D_4 = \begin{pmatrix}
0 & -1 \\
1 & 0
\end{pmatrix}
\]

The retrieval result for character ‘a’, filtered with different filters can be seen in Table 3.1. The result in the column marked PM corresponds to the percentage of when the correct font is returned as the best match (Perfect Match), and the T5 column when the correct font can be found within the five best matches (Top 5). The same notation will be used in the rest of this thesis. The table shows that the combination of one horizontal and two diagonal filters gives the best result. Some of the experiments with varying image sizes are listed in the same table, showing that images of size 25 × 25 pixels seem to be a good choice. Reducing the image size without losing retrieval accuracy is beneficial since the computational load will decrease. Sizes below 15 × 15 pixels decreased the retrieval accuracy significantly.

To verify the result from character ‘a’, a second test was carried out with characters ‘d’, ‘j’, ‘l’, ‘o’, ‘q’ and ‘s’, from testdb2. The result for different combinations of filters can be seen in Table 3.2 and Table 3.3. The retrieval results vary slightly between different characters, but usually filter combinations \(H + D_1 + D_2\) and \(H + D_3\) perform well. We choose the first combination, a horizontal Sobel filter together with two diagonal filters. The vertical filter does not improve the result, probably because many characters contain almost the same vertical lines.

### 3.4.3 Selection of eigenimages

The selection of the number of eigenvectors, here called eigenimages, has a strong influence on the search performance. The number of selected eigenvectors is a tradeoff between accuracy and processing time. However, increasing the number of eigenimages used leads first to an increased performance, but the
Table 3.1: Retrieval accuracy for different filters and filter combinations. The best results are printed in **bold**. (Character ‘a’ from testdb1. PM=Perfect Match, T5=Top 5)

<table>
<thead>
<tr>
<th>Image size</th>
<th>Filter</th>
<th>PM</th>
<th>T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>40 × 40</td>
<td>H</td>
<td>73</td>
<td>95</td>
</tr>
<tr>
<td>40 × 40</td>
<td>V</td>
<td>57</td>
<td>83</td>
</tr>
<tr>
<td>40 × 40</td>
<td>D₁</td>
<td>69</td>
<td>97</td>
</tr>
<tr>
<td>40 × 40</td>
<td>D₂</td>
<td>66</td>
<td>94</td>
</tr>
<tr>
<td>40 × 40</td>
<td>D₃</td>
<td>66</td>
<td>86</td>
</tr>
<tr>
<td>40 × 40</td>
<td>D₄</td>
<td>56</td>
<td>87</td>
</tr>
<tr>
<td>40 × 40</td>
<td>H + D₁ + D₂</td>
<td>77</td>
<td>97</td>
</tr>
<tr>
<td>40 × 40</td>
<td>H + D₁</td>
<td>75</td>
<td>94</td>
</tr>
<tr>
<td>40 × 40</td>
<td>H + D₂</td>
<td>63</td>
<td>96</td>
</tr>
<tr>
<td>40 × 40</td>
<td>H + V</td>
<td>73</td>
<td>98</td>
</tr>
<tr>
<td>25 × 25</td>
<td>H + D₁ + D₂</td>
<td>82</td>
<td>98</td>
</tr>
<tr>
<td>20 × 20</td>
<td>H + D₁ + D₂</td>
<td>81</td>
<td>98</td>
</tr>
<tr>
<td>15 × 15</td>
<td>H + D₁ + D₂</td>
<td>80</td>
<td>96</td>
</tr>
<tr>
<td>10 × 10</td>
<td>H + D₁ + D₂</td>
<td>53</td>
<td>78</td>
</tr>
</tbody>
</table>

Table 3.2: Retrieval accuracy for different filter combinations, for character ‘d’, ‘j’, and ‘l’. The best results are printed in **bold**. (From testdb2. PM=Perfect Match, T5=Top 5)

<table>
<thead>
<tr>
<th>Filter</th>
<th>d PM</th>
<th>d T5</th>
<th>j PM</th>
<th>j T5</th>
<th>l PM</th>
<th>l T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>88</td>
<td>100</td>
<td>86</td>
<td>99</td>
<td>72</td>
<td>82</td>
</tr>
<tr>
<td>V</td>
<td>86</td>
<td>98</td>
<td>70</td>
<td>94</td>
<td>58</td>
<td>78</td>
</tr>
<tr>
<td>D₁</td>
<td>90</td>
<td>100</td>
<td>82</td>
<td>98</td>
<td>64</td>
<td>85</td>
</tr>
<tr>
<td>D₂</td>
<td>91</td>
<td>100</td>
<td>80</td>
<td>98</td>
<td>66</td>
<td>84</td>
</tr>
<tr>
<td>H + V</td>
<td>89</td>
<td>100</td>
<td>82</td>
<td>98</td>
<td>68</td>
<td>85</td>
</tr>
<tr>
<td>H + V + D₁ + D₂</td>
<td>88</td>
<td>100</td>
<td>80</td>
<td>99</td>
<td>66</td>
<td>85</td>
</tr>
<tr>
<td>H + D₁ + D₂</td>
<td>90</td>
<td>100</td>
<td>85</td>
<td>98</td>
<td>72</td>
<td>88</td>
</tr>
<tr>
<td>V + D₁ + D₂</td>
<td>88</td>
<td>99</td>
<td>79</td>
<td>94</td>
<td>59</td>
<td>82</td>
</tr>
<tr>
<td>D₁ + D₂</td>
<td>89</td>
<td>100</td>
<td>79</td>
<td>97</td>
<td>65</td>
<td>84</td>
</tr>
<tr>
<td>H + D₁</td>
<td>89</td>
<td>100</td>
<td>86</td>
<td>99</td>
<td>75</td>
<td>89</td>
</tr>
<tr>
<td>H + D₂</td>
<td>90</td>
<td>100</td>
<td>85</td>
<td>99</td>
<td>72</td>
<td>88</td>
</tr>
</tbody>
</table>
Table 3.3: Retrieval accuracy for different filter combinations, for character ‘o’, ‘q’ and ‘s’. The best results are printed in **bold**. (From testdb2. PM=Perfect Match, T5=Top 5)

<table>
<thead>
<tr>
<th>Filter</th>
<th>PM</th>
<th>T5</th>
<th>PM</th>
<th>T5</th>
<th>PM</th>
<th>T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>79</td>
<td>97</td>
<td>91</td>
<td>100</td>
<td>92</td>
<td>100</td>
</tr>
<tr>
<td>V</td>
<td>81</td>
<td>99</td>
<td>87</td>
<td>99</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>D₁</td>
<td>81</td>
<td>97</td>
<td>91</td>
<td>100</td>
<td>92</td>
<td>100</td>
</tr>
<tr>
<td>D₂</td>
<td>84</td>
<td>99</td>
<td>92</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>H + V</td>
<td>85</td>
<td>99</td>
<td>95</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>H + V + D₁ + D₂</td>
<td>82</td>
<td>98</td>
<td>93</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>H + D₁ + D₂</td>
<td>83</td>
<td>98</td>
<td>93</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>V + D₁ + D₂</td>
<td>84</td>
<td>98</td>
<td>89</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>D₁ + D₂</td>
<td>85</td>
<td>98</td>
<td>93</td>
<td>100</td>
<td>92</td>
<td>100</td>
</tr>
<tr>
<td>H + D₁</td>
<td>80</td>
<td>97</td>
<td>93</td>
<td>100</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>H + D₂</td>
<td>83</td>
<td>97</td>
<td>91</td>
<td>100</td>
<td>90</td>
<td>100</td>
</tr>
</tbody>
</table>

The contribution of eigen-images corresponding to low eigenvalues is usually negligible. The retrieval performance as a function of number of eigenimages, for scanned and printed versions of character ‘a’, is given in Fig. 3.20. Image size is 24 × 24 pixels, and character images are pre-processed with edge filtering. The figure shows that 30 to 40 eigenimages are appropriate for character ‘a’. Preliminary tests were carried out with other image sizes, and other characters, and most of them show that using 40 eigenimages is sufficient.

A question that arises is whether all 40 eigenimages are needed if we only want to perform classification? An example is the classification into different font styles, like Regular, **Bold** and **Italic**. Experiments show that classification

![Figure 3.20: Retrieval performance for different number of eigenimages.](image-url)
or clustering of font styles can be achieved with only a few of the first eigenimages, corresponding to high eigenvalues. Eigenimages with low eigenvalues are more or less useless for classification, but as can be seen in Fig. 3.20, they are needed to obtain high accuracy in the recognition. Those eigenimages are probably more useful for distinguishing finer details and small variations between characters. A visualization of some of the experimental results can be seen in Fig. 3.21 and Fig. 3.22. In both figures, character ’a’ from the first 200 fonts in the database are plotted in a 2-D coordinate systems. The position of each character image is determined from the projection of the character image onto a subspace spanned by two eigenvectors. In Fig. 3.21 the coordinates are obtained from the second and third eigenimages ($u_2$ and $u_3$), both corresponding to high eigenvalues. Different font styles are gathered in different clusters, or regions, in this 2-D space. In Fig. 3.22 the coordinates are obtained from eigenimages corresponding to the two lowest eigenvalues ($u_{39}$ and $u_{40}$). Here the characters seem to be almost randomly distributed. Classification properties are further investigated in section 3.7.

### 3.5 Choosing Components

In the previous section we discussed the basic design of the search engine, and the most important design parameters. We now continue with components for fine tuning the system. First we investigate the role of image scaling and in-
Figure 3.22: Character 'a' from the first 200 fonts in the database plotted in the space spanned by the eigenimages corresponding to the two lowest eigenvalues.

Table 3.4: Retrieval accuracy for different scaling of character 'b' (from testdb1).

<table>
<thead>
<tr>
<th>Image size (rows x columns)</th>
<th>PM</th>
<th>T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>24 × 24</td>
<td>77</td>
<td>99</td>
</tr>
<tr>
<td>24 × 20</td>
<td>81</td>
<td>99</td>
</tr>
<tr>
<td>20 × 24</td>
<td>71</td>
<td>98</td>
</tr>
</tbody>
</table>

terpolation techniques. Then we test if features not derived from eigenimages could improve retrieval performance. Finally, methods for similarity measurements are evaluated.

3.5.1 Image scaling and interpolation

It was shown in Section 3.4, that a suitable image size for character 'a' is 24×24 pixels. However, square images are not suitable for characters like 'l' and 'i', therefore we use rectangular eigenimages for "rectangular characters". As an example, the retrieval accuracy for character 'b' (considered as a "rectangular character") can be seen in Table 3.4.

Scaling requires interpolation, and the influence of the interpolation method on the search performance was evaluated for three common interpolation techniques: nearest neighbor, bilinear and bicubic interpolation. The experiments
3.5 Choosing Components

Table 3.5: Retrieval accuracy for features not derived from eigenimages (for character 'a' from testdb1).

<table>
<thead>
<tr>
<th>Image size</th>
<th>Extra feature</th>
<th>PM</th>
<th>T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>24 × 24</td>
<td>none</td>
<td>70</td>
<td>97</td>
</tr>
<tr>
<td>24 × 24</td>
<td>Area/Box</td>
<td>71</td>
<td>98</td>
</tr>
<tr>
<td>24 × 24</td>
<td>Height/Width</td>
<td>80</td>
<td>99</td>
</tr>
<tr>
<td>24 × 24</td>
<td>Centroid</td>
<td>70</td>
<td>97</td>
</tr>
<tr>
<td>24 × 24</td>
<td>All three above</td>
<td>80</td>
<td>99</td>
</tr>
</tbody>
</table>

revealed that the interpolation method is of minor importance as long as something more advanced than nearest neighbor is used.

3.5.2 Features not derived from eigenimages

We also evaluated if features not derived from eigenimages could improve the retrieval performance. We evaluated the influence of the ratio between character height and width (before scaling), the ratio between the area of the character and the area of the surrounding box, and center of gravity (centroid) values. The result is shown in Table 3.5. The only extra feature that resulted in significant improvements is the ratio between character height and width. However, in the final implementation, it is important that the value is weighted properly. Combinations of different extra features did not produce higher accuracy compared to the case when only ratio between character height and width is used.

3.5.3 Measuring similarity

The similarity between feature vectors $x_i$ and $y_i$ of length $n$ is calculated with the $L_2$ norm, or Euclidean distance, given by

$$d(x, y) = \left( \sum_{i=1}^{n} |x_i - y_i|^2 \right)^{\frac{1}{2}}$$

(3.7)

We also tested other distance metrics ($L_1$, and Mahalanobis distance with different covariance matrices), but the Euclidean distance gave the best result. This might be related to the fact that eigenimages are calculated with Principal Component Analysis, which is defined as the minimizer of the $L_2$ approximation error.
3.6 Results

In this section the overall results are presented. We also describe our experiments investigating the effects of different pre-processing methods and noise sensitivity. Discussions about computational complexity, scalability, memory consumption, and other topics related to processing large amount of data, will not be given here because it highly depends on individual implementations and the system in use. Since the presented recognition method is based on well known and frequently used standard methods it should be possible to implement the search engine using almost any software and hardware. And a lot of different methods, like tree-structures or other multi-dimensional indexing techniques, can be used for maintaining scalability.

3.6.1 Method composition

Below is an overview of the final combination of different methods and settings. For some choices there are tradeoffs between accuracy and computation time. These settings and parameters are used for both the training set (character images from the original font database) and query images:

* **Image scaling:** We use square images, size $24 \times 24$ pixels, for "square characters" like a, e, and o, and rectangular images, for instance $24 \times 20$ pixels, for "rectangular characters" like l, i, and t. Characters are aligned and scaled to fill the whole image. We use bilinear interpolation for scaling.

* **Edge filtering:** Three Sobel filters, one horizontal and two diagonal.

* **Number of eigenimages:** 40

* **Extra features:** Ratio between character height and width before scaling.

* **Distance metric:** $L_2$ norm (Euclidean distance)

3.6.2 Font databases

The original database contains 2763 different fonts for the English alphabet. Single characters are represented by images, with a typical height of approximately 100 pixels. Fig. 3.2 shows some examples of character 'a' in different fonts. For evaluation, three test databases were created. The first contains images of characters (from the original database) that are printed in 400 dpi with an ordinary office laser printer, and then scanned at 300 dpi with an ordinary desktop scanner (HP Scanjet 5590, default settings). As a first step 100 randomly selected characters of 'a' and 'b' were scanned (testdb1). These 200
Table 3.6: Overview of test databases.

<table>
<thead>
<tr>
<th>Name</th>
<th>Number of fonts</th>
<th>Characters</th>
<th>Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>testdb1</td>
<td>100</td>
<td>a, b</td>
<td>200</td>
</tr>
<tr>
<td>testdb2</td>
<td>100</td>
<td>a-z</td>
<td>2600</td>
</tr>
<tr>
<td>notindb</td>
<td>7</td>
<td>a</td>
<td>7</td>
</tr>
</tbody>
</table>

images are used in the majority of the experiments. For the second database (testdb2) the same 100 fonts were used, but this time all small characters were scanned, giving totally 2600 test images. These images are used in the evaluation of the final search engine. The third test database also adopts the print and scan procedure, as mentioned above, but with fonts that are not included in the original database. Only seven fonts were used, all of them downloaded from dafont (www.dafont.com). Both, fonts with an “ordinary look”, and fonts with unusual shapes were used. An overview of all three test databases can be seen in Table 3.6.

3.6.3 Overall results

The final search engine is evaluated with testdb2, containing 100 different fonts with 26 characters in each. In total 2600 characters, first printed and then scanned to resemble a real life situation. The search performance is measured on performance of single character queries. For different characters the results are shown in Table 3.7. The mean values for a perfect match and a top 5 result is 88.2 and 99.1%. The poor result for characters ‘l’ and ‘i’ decrease the mean values rather significantly. The reason is quite obvious; those characters contain relatively few lines and details that can be used for distinguishing fonts from each other. Without ‘l’ and ‘i’, the mean values increase to 89.2 and 99.8%. Since the input to the search engine is a text line, tricky characters like ‘l’ can be removed or weighted down, and usually the remaining characters will be sufficient for producing an accurate result.

3.6.4 Image quality influence

Experiments based on image resolution/scaling, JPEG compression and character rotation are presented in this section. Results are reported from experiments made with character ‘a’ from testdb1, but similar results where found for other characters. Fig. 3.23 shows the relationship between query image size and search performance. We can observe that a query image size below 40 pixels (height) will decrease the retrieval performance significantly. In Fig. 3.24, the correlation between different JPEG compression rates and search performance is shown. In JPEG compression, the quality can be set between 0 and 100, where 100 corresponds to the best quality (lowest compression). As shown in
Table 3.7: Search performance for different characters (from testdb2, PM=Perfect Match, T5=Top 5)

<table>
<thead>
<tr>
<th>Character</th>
<th>PM</th>
<th>T5</th>
<th>Character</th>
<th>PM</th>
<th>T5</th>
<th>Character</th>
<th>PM</th>
<th>T5</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>94</td>
<td>100</td>
<td>j</td>
<td>85</td>
<td>99</td>
<td>s</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>b</td>
<td>90</td>
<td>100</td>
<td>k</td>
<td>86</td>
<td>100</td>
<td>t</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>c</td>
<td>89</td>
<td>99</td>
<td>l</td>
<td>71</td>
<td>88</td>
<td>u</td>
<td>91</td>
<td>100</td>
</tr>
<tr>
<td>d</td>
<td>90</td>
<td>100</td>
<td>m</td>
<td>91</td>
<td>100</td>
<td>v</td>
<td>89</td>
<td>99</td>
</tr>
<tr>
<td>e</td>
<td>91</td>
<td>100</td>
<td>n</td>
<td>91</td>
<td>100</td>
<td>w</td>
<td>91</td>
<td>99</td>
</tr>
<tr>
<td>f</td>
<td>89</td>
<td>100</td>
<td>o</td>
<td>83</td>
<td>98</td>
<td>x</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>g</td>
<td>88</td>
<td>100</td>
<td>p</td>
<td>88</td>
<td>100</td>
<td>y</td>
<td>87</td>
<td>100</td>
</tr>
<tr>
<td>h</td>
<td>88</td>
<td>100</td>
<td>q</td>
<td>93</td>
<td>100</td>
<td>z</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>i</td>
<td>82</td>
<td>94</td>
<td>r</td>
<td>86</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>MEAN</td>
<td>88.2</td>
<td>99.1</td>
</tr>
</tbody>
</table>

Figure 3.23: The relationship between query image size and search performance. The dashed line corresponds to a perfect match, the solid line corresponds to a top 5 result.

In this section the complete process from input image to the font name output is illustrated. An example of an input image can be seen in Fig. 3.26. The

3.6.5 An example of a complete search

In this section the complete process from input image to the font name output is illustrated. An example of an input image can be seen in Fig. 3.26. The
3.6 Results

Figure 3.24: The relationship between JPEG compression rate and search performance. The dashed line corresponds to a perfect match, the solid line corresponds to a top 5 result.

Figure 3.25: The relationship between query image rotation and search performance. The dashed line corresponds to a perfect match, the solid line corresponds to a top 5 result.
first step is to ensure that the text line is horizontal. This is done using the Hough transform on the edge filtered image (see Section 3.3), producing the result seen in Fig. 3.27.

Then the text line is segmented into sub images, each corresponding to one character (see Section 3.3). The first 12 sub images are shown in Fig. 3.28. The user will then assign letters to sub images that are to be used in the database search. Since the segmentation algorithm did not manage to segment 'k' and 'j', the user should not assign a letter to this sub image. The character segmentation can be improved, but here we are primarily interested in font recognition and therefore we did not implement a more sophisticated segmentation method.

Assigned images will be used as input to the search engine. Results from individual characters are weighted and combined to a final result, presenting the most similar fonts in the database. Although the combination of different characters is very important for the final result, we leave this discussion for upcoming publications (see Section 6.1). Fig. 3.29 shows the query image and the five most similar fonts in the database. In this case the first seven characters were assigned letters and used as input. Even when the correct font can not be found in the database a match list will still be presented to the user, showing the most similar fonts. The user can then continue the search by using a font from the match list as next query. In the public implementation of the search engine, search results are color coded according to how similar they are to the query (we believe that for most users a color will be easier to interpret than the exact similarity score).

http://media-vibrance.itn.liu.se/fyfont/

Figure 3.28: First 12 sub images after character segmentation.
3.7 Visualizing the Font Database

Denvita Denvita Denvita
(a) Query images (b) Second best match (c) Fourth best match
Denvita Denvita Denvita
(d) Best match (e) Third best match (f) Fifth best match

Figure 3.29: (a) Query images (b)-(f) The five most similar fonts in the database.

3.6.6 Fonts not in the database

It is always difficult to know how the search engine will respond when query images contain fonts that are not in the database. The retrieval accuracy can’t be measured in the same way as in previous sections, instead we need to evaluate the result visually. Fig. 3.30 shows seven query images (character ’a’) from fonts not present in the database, together with the five best matches. The problem is that for some query images it might be feasible that we can not find anything similar at all in the database. However, if this happens, we do believe that showing a match list to the user is beneficial since fonts from the match list can be used as next query. As mentioned in the previous section, search results are color coded according to how similar they are to the query, so if a font can not be found in the database the user will be aware of the situation.

3.7 Visualizing the Font Database

Visualizing the content of a font database can be of importance in many situations. For persons who are managing and selecting fonts in their daily work (e.g. various people employed in the graphic design industry) a visual overview over all fonts in the database might be helpful. Moreover, in relation to the font search engine, we can assume some users want to find a particular font, without using a query image. If the example images on the front page can’t accomplish the task, we need to find other ways of visualizing the database content. This is a challenging task, especially if the size of the database is large. We describe how to use the findings from the font recognition study in a visualization tool for large font databases. The proposed method should be interpreted as a pilot study and inspiration for future research.

The basic idea with the method is to order all images of a given character in the different fonts in a transparent way, giving the user an opportunity to effectively interact with the database. As starting point we use the high-dimensional
vectors (41 dimensions) obtained from the font search engine. Then linear and non-linear mappings are used to project the 41-D space of all images of a given letter to a two-dimensional manifold. Finally we describe a refinement process that maps these raw two-dimensional representations to a regular 2-D grid to avoid overlapping images and provide a better overview for the user.

The database is almost the same as the one used earlier. The only difference is that 8 files were now found to be corrupt, leaving 2755 fonts for the English alphabet. Here we use images from the original font database only and do not consider image degradation due to printing and scanning since now our goal is the visualization of the database and not the retrieval from the database.

3.7.1 Related work

We mention a few examples regarding visualization of image database content, not particularly focusing on font databases. Nguyen and Worring [59] present a method for similarity based visualization based on three criteria: 1) The displayed images should show a general overview of the whole dataset, 2) the original structure of the data should be preserved, and 3) reduce image overlap as much as possible. They use the SNE (Stochastic Neighbor Embedding) and the ISOMAP methods for preserving neighbor identities and redefine the dissimilarity matrix.
3.7 Visualizing the Font Database

Moghaddam et. al. [53] visualize retrieved images on a 2-D screen not only in order of their decreasing similarities, but also according to their mutual similarities. Images are converted to feature vectors based on color moments, wavelet based textures, and water-filling edge features. They use PCA for reducing the dimensionality of the feature space to only two dimensions (using the two eigenvectors corresponding to the two largest eigenvalues), and then, to minimize the overlap between images, a constrained nonlinear optimization approach is used.

For non-conventional browsing in image databases we mention Torres et. al. [89], where images are placed on concentric rings or spirals. The motivation for using such shapes is that they make it easier to keep user focus on both the query image, and the most similar retrieved images. The query image is placed in the centre of the spiral or rings, and retrieved images are placed on surrounding rings depending on similarity (images with high similarity close to the center). Also Heesch and Rüger [32] propose some interesting ways of browsing in image databases. Three different interfaces are presented and evaluated. First an interface where images are displayed in the form of a spiral, with an images’s distance from the center being proportional to the distance to the query image in feature space. The second interface uses basically the same approach, but images in the periphery are displayed at a smaller size. By placing images from the center and outwards, the method avoids overlapping images. In the final approach a pre-selection task is utilized where the user is shown a set of nodes (images) that are representative for different collections of images. Clicking on one of the nodes recovers the set of nearest neighbors from the database, which are then displayed such that their distances to the center are proportional to their dissimilarity to the selected node. Within a query the user can easily move between interfaces for a better understanding of the search result.

3.7.2 Dimensionality reduction

After filtering and feature extraction, described earlier in this thesis, every character in the font database is represented by a 41-dimensional feature vector. If we want to visualize the content of the database we have to reduce the dimensionality of the descriptor space further. Since we want to use a conventional monitor as display device we have to map the 41-dimensional space to a two-dimensional surface. To be meaningful this mapping must of course preserve the intrinsic geometrical structure of the original representation space.

In the following we will illustrate two examples of such mappings, one linear, the other non-linear. The linear mapping is simply a selection of the second and third component in the PCA coordinate space (the first coordinate has very little visual information related to font appearance). In many applications linear mappings are too restrictive since they cannot take into account
that the actual data vectors may be located on a lower dimensional manifold embedded into a higher dimensional space. In the simplest example of such a situation the data vectors are all located on a two-dimensional circle. Traditional PCA will always indicate that the data is two-dimensional and will need two principal components to represent the data. These two-dimensions are, however, only a consequence of the coordinate system used to describe the data. Selecting a polar coordinate system with origin at the center of the circle will reveal that the radial value for all data vectors is constant and that the data can be represented by the angle without information loss. In this simple example it is clear how to choose the coordinate system in which the lower-dimensionality of the data becomes obvious. In the general case it is however very difficult to estimate the intrinsic dimensionality of the data and to construct non-linear mappings to achieve this reduction. This problem is known as manifold learning in the machine learning community. We don’t give a detailed discussion of these methods here but refer the reader to http://www.math.umn.edu/ wittman/mani/ for an overview over some of the methods. There one can also find implementations of eight popular non-linear methods. In our experiments we mainly worked with the ISOMAP algorithm described in Tenenbaum et. al. [86].

The ISOMAP (isometric feature mapping) is based on the observation that the data vectors can locally be fitted by linear approximations but that traditional methods like PCA fail to recognize the global structure since they apply the Euclidean distance measure also to far-away points for which this approximation is no longer valid. ISOMAP avoids this error by estimating the true distance between two data points by first linearly fitting neighboring points. Then these local approximations are patched together and the shortest paths connecting any two points is computed. In the final step this table of approximations of the geodesic distances between point pairs is used to embed the data points into a low-dimensional subspace such that the intrinsic geometry is preserved as much as possible. The result is in our application a two-dimensional representation of the characters in the database that tries to preserve the intrinsic geometry in the 41-dimensional feature space.

Many powerful tools for both linear and non-linear dimensionality reduction have been presented in the past. Well-known examples are SOM, or Self-Organizing Maps (see http://www.cis.hut.fi/projects/somtoolbox/), also known as the Kohonen Maps, Locally Linear Embedding (LLE) [69], and Laplacian Eigenmaps [4]. Preliminary experiments with Self-Organizing Maps reveal no improvements compared to the PCA or ISOMAP approach. However, the performance of the SOM algorithm seems to improve if rather few of the dimensions included in the original dataset (totally 41 dimensions) are used as input. Possibly, the explanation is related to the conclusion made by De Backer [16], that SOM performs better for very low number of dimensions. The method might be more suitable in a classification task, where the obtained code vectors
(also known as model vectors or prototype vectors) provide the classes. In general, the use of different dimensionality reduction methods should be further investigated in future research.

### 3.7.3 Grid representation

The visualizations produced by the PCA or ISOMAP approach aims at preserving the relative distance between data points in the high- and low-dimensional spaces. This provides a representation of the relative density in different regions of the font space but it also leads to visualizations in which large parts of the space are relatively empty while others are rather crowded. For many purposes such geometrical similarities are less important while a more ordered presentation will provide a better overview over the database content. Our solution is to evenly distribute images on a square grid, while at the same time trying to maintain mutual relations. The following algorithm was used to compute such an ordered display.

Let $M$ be the matrix containing the font indices: $k = M(r, c)$ means that font number $k$ is presented at position $(r, c)$ in the display. We use an almost square-formed display leading to a matrix of size $52 \times 53$ for the 2755 fonts. The matrix $P$ contains the coordinates of the characters as computed by PCA or ISOMAP: the vector $(P(i, 1), P(i, 2))$ contains the coordinates for character images $i$ in the two-dimensional space obtained by the PCA or ISOMAP transformation. The coordinates in $P$ are first scaled to fit the size of $M$. Each element in $M$ will be assigned an image index $i$ corresponding to the nearest neighbor in $P$.

The algorithm used is as follows: Starting with the top row, for every position $(r, c)$ we do the following:

1. Find the index $i$ giving the minimum Euclidean distance between $M(r, c)$ and $P(i)$, where $i = 1 \ldots 2755$.
2. Move the image at coordinates $P(i)$ to $M(r, c)$.
3. Mark the index $P(i)$, so it can not be relocated.
4. If we have not reached the end of the row increase the column index by one and go to step 1., if we have reached the end of the row, reset the column index to 1 and go to the next row.

### 3.7.4 An example: Visualizing character ‘a’

In Fig. 3.31 and Fig. 3.32 we show all the font images for character ‘a’ at coordinates obtained from the PCA respectively ISOMAP approach. It can be seen that the ISOMAP reduction seems to better divide different types of fonts into different clusters, and fewer images are overlapping. In Fig. 3.33
Figure 3.31: A visualization of all characters ‘a’ in the database based on the second and third PCA coefficients.

and Fig. 3.34 the PCA and ISOMAP results for character ‘a’ are converted to a grid representation, as described in previous section. Also here the best performance is shown for the ISOMAP reduction. In the final application, the idea is that the user can zoom in on parts of the space for studying a few fonts more closely, and then obtain names, and possibly more characters, for specific fonts.

We also tested if the ordered representations could be improved by further optimizations. For this purpose we selected 2 points in an ordered representation described above. If we could improve the presentation by swapping these two points then the accumulated distances between these two points and their neighbors would decrease after the swapping compared to the same accumulated distance before the swapping. We created a large number of random pairs for swapping and systematically tested swapping points located near each
Figure 3.32: A visualization of all characters 'a' in the database based on ISOMAP manifold learning.
Figure 3.33: A grid representation of all characters ‘a’ in the database based on the PCA approach.
3.7 Visualizing the Font Database

Figure 3.34: A grid representation of all characters 'a' in the database based on the ISOMAP representation.
other. In none of the cases we could observe a reduction of the distances after swapping. This indicates an optimal solution of the selected ordering method.

A visual evaluation of the image grid obtained from the ISOMAP reduction reveals that different font styles are clearly separated. In the upper left corner mainly bold characters are located. The upper right corner is dominated by the regular style. Similar regions can be found for italic, "small caps", etc. We believe the proposed interface can be used by, for instance, people in the graphic design industry, as an efficient tool for browsing in large font databases. The user can relatively quickly find a group of interesting fonts, and study these in more detail.

3.8 Online Implementation

The font recognition system presented earlier is implemented in a font search engine, called FyFont, publicly available on the Internet\(^4\). The search engine is currently hosting 2364 non-commercial fonts. This section will briefly describe the system setup, followed by user statistics gathered during a period of 20 months.

3.8.1 System setup

Interaction with the user is carried out through a web front page implemented in the scripting language PHP. Through a socket connection, PHP is communicating with MATLAB\(^5\), which is running as an internal server in the background\(^6\). The MATLAB environment is hosting all components dealing with calculation and comparison of features, image pre-processing and data storage. The main reason for using MATLAB in the recognition unit is to save time in the implementation process (since we are only creating a demo). However, all methods can easily be implemented in most programming languages.

The communication between the user and the search engine is illustrated in Fig. 3.35. First, the user can either upload an image containing a text line, or submit an image URL. The received image is sent to MATLAB and the pre-processing unit, where the text line is rotated to a horizontal position, and segmented into single characters. Segmented images are presented on the front page, and the user is asked to assign letters to characters that have been segmented correctly. Characters with corresponding letters are submitted to

\(^4\)http://media-vibrance.itn.liu.se/fyfont/

\(^5\)MATLAB is a high-level technical computing language and interactive environment for algorithm development, signal and image processing, data visualization, data analysis, and numeric computation. See: http://www.mathworks.com/

\(^6\)Running MATLAB as a server can be accomplished by using for instance the "TCP/UDP/IP Toolbox" by Peter Rydesäter, available through the MATLAB Central: http://www.mathworks.com/matlabcentral/fileexchange/345
3.8 Online Implementation

Figure 3.35: A flow chart describing the interaction between the user and the search engine. The front page, which is responsible for all communication with the user, is implemented in PHP. The search engine running in the background is implemented in MATLAB.

MATLAB and the recognition unit. Feature vectors are calculated for each character, and resulting features are compared to features for all fonts in the database. The match result is sent back to the front page, and presented to the user. The user can continue browsing the database by submitting font images from the obtained result, or return to the start page and submit another input image. The Graphical User Interface for each phase described above is demonstrated in Fig. 3.36-3.38.

Offline tools have been created for building the feature database. When adding new fonts to the database, corresponding feature vectors are simply added to new rows in the database. Since no tree-structure (or other multi-dimensional indexing technique) is used for sorting and saving the data, the recognition time \(O(N)\) will increase linearly with new fonts added to the database. However, this is not believed to be a major problem since the total
Figure 3.36: The start page. The user can upload an image, submit an image URL, or search with one of the example images.
Figure 3.37: The second page. The submitted text line has been segmented into characters, and the user is asked to assign letters to characters that have been segmented correctly.
Figure 3.38: The final page, displaying the search result. The top row shows the query (here the user has chosen to search with four letters). Subsequent rows correspond to the best matches in the database.
Figure 3.39: Examples of images that failed in the segmentation process. A common source of failure is that characters are too close to each other, or even connected as in the lower right image. Otherwise, images containing a lot more than merely plain text are also more challenging for the segmentation algorithm.

number of fonts (compatible with the English Alphabet) nowadays publicly available worldwide is approximately 50 000 - 60 000 (both commercial and non-commercial fonts), and the feature vectors describing the font characteristics are very compact. Consequently, on an ordinary server or desktop computer, a search in the database can be accomplished within a few milliseconds.

3.8.2 User statistics

This section will review some search engine statistics gathered during a period of 20 months (March 2007 - October 2008). Statistics that can help improving the search engine are of particular interest. However, we start with some general measurements. The front page has in total collected 834 visits, from 62 different countries/territories. Out of 834 visits, the unique number of visitors is 543. On average, a user continues using the search engine for 4 minutes and 21 seconds. The main traffic source is search engines (43%), followed by referring sites (34%) and direct traffic (23%).

Many users are only testing the search engine with one of the test images provided on the start page. However, 379 external images were uploaded. 17 were not segmented at all, mainly due to the fact that they did not contain any distinct characters. See some examples in Fig. 3.39. Furthermore, 106 images were segmented, but the user did not use any of the segmented characters, indicating that the segmentation process failed in some way. In other words, 256 images, or 68% of the uploaded content, were successfully segmented and used in a search query.

The recognition accuracy highly depends on the image size/resolution or
quality, as shown in Section 3.4. Moreover, the segmentation unit also prefers images of rather high quality, with images containing characters that have a certain minimum height. The distribution of input image heights are shown in Fig. 3.40. Here we assume that character height is equivalent to image height. The average image height is 108 pixels. However, we notice that rather many images are of a smaller size than the recommended minimum (80 pixels) specified in the instructions on the start page. The figure also shows how many of the submitted images that failed in the segmentation process, and how many that were segmented but not used in a query. One conclusion is that smaller images (below recommended minimum size) more often generate a failure than larger images.

For those 256 images segmented correctly and used in a query, the average number of assigned characters is 4.1. The relationship between number of segmented characters, and number of assigned characters, is shown in Fig. 3.41 (the maximum number of segmented characters is 20, and the user can assign at most 10 of those characters). An unexpected finding is that most users are querying the search engine with single characters, even though the accuracy will increase if more characters are submitted. However, the figure shows that in general, users tend to assign characters to every segmented image.

An interesting question is whether users are satisfied with the search result.
3.8 Online Implementation

In other words, if their particular font, or a similar font, is included in the search engine’s database. Unfortunately, users were not asked that particular question, so the ground truth is unknown. However, recognition score boundaries marked with green (“A good match. It might be the correct font”), blue (“A rather good match, but probably not the correct font”), and red (“Not a good match”) were set empirically. Fig. 3.42 shows the distribution of recognition scores for all queries, and which boundary or category they belong to. 10% of the queries resulted in an excellent score (green), suggesting that the correct font or a very similar font was found. In 66% of the queries, the result is a rather good match (blue), but for 24% of the queries, it was not possible to find anything in the database similar to the query font (red).

Another statistical measure that might indicate if users are satisfied or at least attracted by the search result is whether they continue the search by clicking on one of the fonts in the search result. Then the chosen font is used as the next query. It turns out that as much as 48% of the users (who ended up with a search result) continue to search within the database. Among those users, 2.1 is the average number of times the user click on results within the retrieved font list. The all time high record for the number of clicks by a single user in one session is 26 clicks!

A topic that was discussed early in the project was whether users will mainly submit very strange looking fonts, or fonts with a more regular appearance. If mainly regular fonts are processed it might be possible to connect a character
Figure 3.42: The distribution of recognition scores. Green (here printed as white) corresponds to a really good match between the query font and a font in the database. Blue (here printed as gray) corresponds to a rather good match, but probably not the correct font. For a red (here printed as black) recognition score, nothing in the database is similar to the query.

Figure 3.43: Examples of images submitted to the search engine. Typically, fonts have a very regular appearance, like the examples above. Very strange looking fonts have not been encountered.
recognition system to the search engine, and help users to assign characters. Visual evaluation of submitted images shows that fonts with a regular appearance are dominating. A few examples of input images can be seen in Fig. 3.43. No extreme looking fonts could be found among submitted images.

In summary, for readers interested in developing their own font search engine, following issues should be taken into consideration. The concerns might influence both how to design the system, and how to evaluate retrieval results.

* A vast majority of the users are searching with a single character.

* The height of the input image is often rather small. Images below 100 pixels are common, probably containing even smaller character heights.

* Many users continue to search within the database by clicking on retrieval results. One should make use of this and implement a more sophisticated "within-search" or feedback mechanism. For instance, one can let the user select several appealing fonts from the retrieved result, and search for a new font based on weighting of selected fonts.
Chapter 4

Color Emotions in Image Retrieval

This chapter will present a novel image retrieval method based on color emotions. Extensive psychophysical experiments are used for evaluating the method.

4.1 Introduction

Most of us live in a world full of colors, usually perceived in an infinite number of multi-colored combinations. Our colorful environment affects us in many ways. For instance, the relationship between colors and human emotions has a strong influence on how we perceive our environment. Naturally, the same holds for our perception of images. All of us are in some way emotionally affected when looking at a photograph or an image. One can often relate some of the emotional response to the context, or to particular objects in the scene, like familiar faces, etc. Simultaneously, as part of the color perception process, also the color content of the image will affect us emotionally. Such emotions, generally called color emotions, can be described as emotional feelings evoked by single colors or color combinations. They are typically expressed with semantic words, such as “warm”, “soft”, “active”, etc. Color emotions, together with color memory, color harmony, color meaning, etc., belong to the cognitive aspects of colors. The original motivation for this research was to include high level semantic information, such as emotional feelings, in image classification and image retrieval systems. Emotional responses based on objects, faces etc. are often highly individual, and therefore one has to be careful when including them in classification of general image databases. However, the emotional response evoked by color content is more universal. Consequently, the question dealt with in this thesis is whether the emotional response related to colors
in ordinary multi-colored images are similar between persons, like previous research has shown for color emotions related to single colors and two-color combinations. If so, one can combine color emotions with Content Based Image Retrieval and discover new ways of searching for images in semantic image search. One can mainly think of the method as a tool for sorting or pre-selecting images. For instance, a popular keyword-query may result in a huge set of images that are impossible for the user to review. New methods that can help the user by selecting a subset of the query result, or grade the images found, are therefore highly desirable.

Most of the popular search engines, like Google, are mainly text or keyword based. Others are using features from traditional image processing, and some are specialized to specific tasks like face recognition. Some of these systems are also combining a wide variety of techniques like computer vision, speech recognition and understanding and automatic translation to assist in searching huge video databases. Most of these systems are however using objects as their prime descriptors of image content. A typical query could thus be the problem to find all images showing a car or a given person or a scene with a mountain. Instead of focusing on objects etc., we propose to use emotion-related properties of images when searching in image databases. The presented method can be used standalone, or in combination with other methods. Regardless, the natural starting point for such a new type of search criteria is to use color as a bearer of emotional information.

The color emotion metric used is obtained from Ou et. al. [61][62][63]. From psychophysical experiments Ou et. al. derived color emotion models for single colors and two-color combinations. By factor analysis they identify three color-emotion factors: activity, weight, and heat. We demonstrate how to use Ou’s models together with ordinary RGB-histograms of images to obtain compact but efficient image descriptors. Images can be emotionally classified by positioning on different scales related to color emotion words. We also derive emotion histograms, describing the distributions of vectors in the space spanned by the emotion scales, and use them for image retrieval. Similarity measurements between histograms are used to retrieve images with similar emotional appeal as the query image. Since the method only involves transformations on ordinary RGB-histograms, usually already computed in many Content Based Image Retrieval systems, the method is very time efficient, which is essential when dealing with very large image databases containing millions or billions of images. Initial experiments with a small database of 5000 images show that color emotion properties are well suited for semantic image classification and retrieval.

Research on color emotions for single colors and two-color combinations has received considerable attention in the past. However, similar investigations for multi-colored images are less common, which probably is one of the explanations why color emotions are seldom used in image retrieval systems. In
4.2 Background

Background material for color emotions in image retrieval can be recovered from two different research areas. Color emotion research is usually carried out within domains like cognitive psychology, product design, etc. Research concerning semantic image retrieval is typically published by researchers from computer science, possessing some special interests in semantic concepts. The gap between those research areas is sometimes huge, but with this thesis we hope to narrow the gap.

Research on color emotions for single colors and two-color combinations is by now a well established research area. In a series of papers Ou et. al. [61][62][63] investigated the relationship between color emotions and color preference.
Color emotion models for single colors and two-color combinations are derived from psychophysical experiments. Observers were asked to assess single colors on ten color emotion scales. It is then shown that factor analysis can reduce the number of color emotions scales to only three categories, or color emotion factors: activity, weight and heat. Ou et. al. conclude that the three factors agree well with studies done by others, for instance Kobayashi [40] and Sato et. al. [68]. In this study we will use those emotion factors when investigating color emotions for multi-colored images. In another study of human’s emotional response on colors Gao and Xin [24] selected twelve pairs, or scales, of color emotion words which were considered fundamental. They also show that most of the variance in the data can be represented by fewer factors than twelve. By maximum likelihood factor analysis they group scales into three categories or indexes, called activity, potency and definition. One of their conclusions is that color emotion connotations are mainly connected to lightness and chroma, and less connected to hue. One important question is whether color emotions are influenced by different regional or cultural backgrounds. In an extensive study by Gao et. al. [25] it was concluded that the influence of cultural background on color emotions is very limited. In psychophysical experiments totally 214 color samples were evaluated on 12 emotion variables by subjects from seven different regions worldwide. Using factor analysis they show that a smaller number of factors are needed for the representation, which corresponds well to other studies. Another conclusion, common with others studies, is that lightness and chroma are the most important factors in color emotions, whereas the influence of hue is limited. Similar results about regional and cultural backgrounds were earlier found in cross-regional comparisons by Xin et. al. [99][100]. Also age-related differences were investigated. A recent example is Beke et. al. [3], where color preference of aged observers are compared to young observers. The results indicate important differences, both depending on neuro-physiological changes, and other aspects such as cultural implications.

Related to the problem of color emotions is the concept of color harmony. Harmonic color combinations are colors that are said to generate a pleasant effect when seen in neighboring areas. Research about harmony has usually been carried out on rather restricted combinations of colors, for instance two-color combinations. An extensive study by Ou and Luo [60] investigates harmony in two-color combinations in order to develop a quantitative model for prediction. In psychophysical experiments color pairs were assessed on a category scale related to harmony. According to the authors the model shows satisfactory performance for prediction of two-color combinations. However, they point out that the model was only developed for two-color combinations, consisting of uniform patches on a gray background. They conclude that harmony of more complex images containing more than two colors, etc. needs to be addressed in future research. Another extensive study on harmony, using the Coloroid color system and including many thousand participants, was presented by Neme-
In the first paper the harmony content of different scales with similar hue was investigated. The paper ends with an extensive list of detailed conclusions. However, it seems rather complicated to express them in general terms, applicable in, for instance, other color spaces. The second paper investigates the harmony content of different monochrome color pairs. Some of the conclusions are that a harmony function can describe the variation of harmony content as a function of brightness- and saturation-intervals, and that the variation of harmony content depending on brightness- and saturation-intervals is not being influenced by the hues of colors of the color pair in the composition. As mentioned earlier, a similar conclusion about the limited influence of hue, but for color emotions, was made by Gao et al. [25].

Harmony has also been used in computational imaging to beautify images. Cohen-Or et al. [9] present a method that enhances the harmony among colors of a given photograph or image. Using a notion of harmony originating from Itten [36], they define different harmonic templates on the hue wheel. Then the user selects a template, and hue values in the image are shifted towards template sectors. The process tries to preserve the original colors as much as possible, and at the same time avoid splitting large regions of uniform colors. A segmentation algorithm is used for finding those color regions. Another important research area related to color harmony is interior and product design. Examples of papers discussing how to select colors in interior design based on harmony and other semantic concepts can be found in Shen et al. [73], Nakanishi et al. [56], and Shen et al. [72]. Similar ways of using color harmony, but in product design, can be found in Tsai and Chou [90], Tokumaru et al. [87], and Hsiao et al. [34].

There are few papers addressing the problem of including color emotions in image retrieval. The methods presented are often focusing on semantic image retrieval in a more general way. Wang and Yu [96] propose an emotional semantic query model based on image color semantic descriptors. Images are segmented by clustering in the CIELAB color space. Then images are converted to the CIELCh color space (the cylindrical version of CIELUV), and segmented regions are converted to semantic terms through a fuzzy clustering algorithm. Both regional and global semantic descriptors are extracted. The user is able to query the image database with emotional semantic words, like "sad" and "warm", and also with more complex sentences. One interesting detail to notice is that they use Gaussian low-pass filtering to remove edges prior to segmentation, with the motivation that the capability of the human visual system to distinguish different colors drops rapidly for high spatial frequencies. However, what is not discussed in the paper is that one should probably be careful with the amount of filtering. As an example we can consider an image containing stripes, where the semantic response may change rapidly with different amounts of low-pass filtering. Also Corridoni et al. [10] use clustering in the color space to segment images into regions with homogenous
Then Itten’s formalism together with fuzzy sets are used to represent intra-region properties (warmth, hue, luminance, etc.) and inter-region properties (hue, saturation, luminance contrast, etc.). Properties are gathered to a color description language based on color semantics. Querying the database is accomplished through a rather complex user interface including sketches and dialog boxes.

In a paper by Hong and Choi [33] a search scheme called FMV (Fuzzy Membership Value) Indexing is presented. It allows the user to retrieve images based on high-level semantic concepts, using keywords such as ”cool”, ”soft”, ”romantic” etc. Emotion concepts are derived from color values in the HSI color space. Cho and Lee [7] developed an image retrieval system based on human preference and emotion by using an interactive genetic algorithm (IGA). Image features are created from average colors and wavelet coefficients. Yoo [103] propose an emotion-based image retrieval method using descriptors that are called query color code and query gray code. The descriptors are based on human evaluation of color patterns on 13 emotion pairs or scales, most of them related to color. The image database is queried with one of the emotions, and a feedback method is utilized for dynamically updating the search result. However, combining the result from relatively simple image descriptors with complex emotion words like ”beautiful” and ”ugly” should be investigated further.

Wang et. al. [97] use a three-dimensional emotional space (with some similarities to the emotion space used in this thesis) for annotating images and perform semantic queries. The space is based on psychological experiments with 12 pairs of emotion words. Image properties are described with different kinds of histograms, and from histogram features emotional factors are predicted using a support vector machine. They create a search interface where the user can create semantic queries based on one of the emotion words. A disadvantage with the presented work is that emotion scales are derived from category scaling without, for instance, anchor images, which is not the most reliable scaling method (discussed later in this thesis). The method was developed and evaluated for paintings only. It would be interesting to evaluate a similar method with a broader set of images. In Lee et. al. [44] they show how rough set theory can be used to build an emotion-based color image retrieval system. Emotion data is extracted by letting people observe different random color patterns in category scaling experiments. Three different emotion scales are incorporated: warm - cool, dynamic - static, and heavy - light. The primarily field of application seems to be different color patterns, like wall papers etc. But the authors mention that the method can also be applied in image retrieval.

We conclude with three recent papers discussing emotion based image retrieval. Datta et. al [12] use a computational approach for studying aesthetics in photographic images. They use photos from an online photo sharing
4.3 Fundamentals

Website, peer-rated in two qualities, *aesthetics* and *originality*. Methods for extracting numerous visual or aesthetical features from the images (like Exposure, Colorfulness, Depth of field, etc.) are developed, and the relationship between observer ratings and extracted features are explored through a statistical learning approach using Support Vector Machines and classification trees. For certain visual features, the results are promising. In [14] they use the same data, but a weighted linear least squares regressor and a naive Bayes’ classifier is utilized, which increases the performance considerably. In a recent paper, Datta et. al. [15] discuss the future possibilities in inclusion of image aesthetics and emotions in, for instance, image classification. They introduce the phrase ”aesthetic gap” (compare to ”semantic gap”), and report on their effort to build a real-world dataset that can be used for testing and comparison of algorithms related to this research area.

The approach presented by Datta et. al. is correlated to observer judgments. However, for other image retrieval methods mentioned above, the evaluations are limited and user studies are missing. It is hard to know whether the models agree with observer judgments. Unique for the retrieval method presented in this thesis is that a comprehensive user study is incorporated in the presentation.

### 4.3 Fundamentals

This section will present some phrases and concepts that are frequently used in research concerning color emotions and other psychophysical experiments. We start with a vocabulary, mainly adopted from Engeldrum [18].

Vocabulary of scaling:

* **Observers**: Subjects, respondents, judges, participants.

* **Samples**: Images, stimuli, objects.

* **Psychometrics**: Measuring human response to a ”ness” or attribute, or image quality.

* **Psychophysics**: The human response to a stimulus specified by physical parameters.

* **Scaling**: Generation of a scale or a ruler of the observers response to an attribute under study.

* **Psychometric scaling**: Generation of scales or rulers of the attributes by human measurement.

* A **judgment** is objective, a **choice** has a personal preference.
Color emotions: Emotional feelings evoked by single colors or color combinations, typically expressed with semantic words, such as "warm", "soft", "active", etc.

Engeldrum uses the phrase *Psychometric scaling*, roughly translated to "mind measuring", when describing how to assign numbers to different sample/image attributes, or "nesses". The problem is basically linking a (sometimes multidimensional) physical stimulus to a subjective perception. One of the most difficult parts is sample selection. Engeldrum lists the following four key factors (see [18] for more details):

1. *What categories should the samples represent?* Random, represent different classes, represent the population, etc.
2. *Range and distribution of attributes or “nesses”?* A common solution is to use samples that span the range of “ness” of interest.
3. *Image size and spatial sampling/resolution?*
4. *Image content?* Sometimes important to avoid critical objects, colors, etc. However, with many samples, spatial configurations will average out.

Before observers start judging samples they need instructions, for instance, what is the “ness” or attribute, and what is the judgment task? Other important instructions should include the context, for instance in what situation the result will be applied, and a definition of the attribute in words that can easily be understood by the observer.

Next, observers will judge samples on different scales. We can mainly think of four types of scales, as listed beneath. Every scale type listed below a particular type has the properties of that scale type plus all the types above.

1. **Nominal scaling**: Names or labels are assigned to objects or events, with a one-to-one relationship. These scales are not so useful alone, but can form the basis of scales of higher types.
2. **Ordinal scaling**: Labels (usually adjectives) or numbers are used to represent the order of the samples. The scale has a greater than, or less than, property. A popular version of ordinal scaling is category scaling, where observers place samples in categories, labeled with names or numbers. A drawback with this approach is that observers tend to use all categories equal number of times (according to Engeldrum [18]). Another drawback with ordinal scaling is that we don’t know the distance between samples, only the order. However, the main advantage is that the method is rather easy for the observer to understand.
3. **Interval scaling**: Adds the property of distance to the ordinal scale, which means that equal differences in scale values represent equal differences in the sample attribute. Interval scaling is a widely used scaling method, and a frequent approach is to let observers place a set of samples on an attribute scale or ruler ranging from one "ness" to another. A consideration is "the rubber band effect", which means that observers are using the scale in different ways. For each observer we have two constants, a multiplier and an additive constant. The solution is to put each observer on a common scale, either by adjusting sample positions to have the same mean and variance, or beforehand placing "anchor-images" on the scale.

4. **Ratio scaling**: Adds an origin to the distance property of the interval scale. Usually one assumes that the scale has a zero point, and the observer gives a numerical response in proportion to the strength of the attribute and the ratio compared to a reference sample. This scaling method is more demanding for the observers, therefore more suitable for experienced observers or experts.

A large number of different methods can be used for evaluating observer judgments. The following methods and concepts are frequently used within color emotion research:

**Factor analysis**: A statistical method used for reducing the space of attributes from a larger number of variables to a smaller number of factors. It uncovers and describes the variability among observed variables, modeled as linear combinations of the factors, plus error terms. Factor analysis and the widely known method of Principal Components Analysis (PCA) are related, and often confused. One difference is that in Factor analysis we don’t need to assume the errors to have the same variance. Another difference is that in PCA we assume that all variability in an item should be used, while in Factor analysis we only use the variability in an item that it has in common with other items. For interested readers, a mathematical definition can easily be found in appropriate literature or on the Internet. When selecting between component analysis and factor analysis, the paper by Velicer and Jackson [92] might be used for guidance. Algebraic similarities and differences are discussed, along with a number of theoretical and practical issues. PCA and Factor analysis often produce very similar results. However, PCA is usually preferred in data reduction, while Factors analysis is usually preferred for detecting structures.

In psychometrics, factor analysis is frequently used to identify factors that explain a variety of results on different observer studies. The number of attributes can be reduced, and it is possible to find structures, such as groups of inter-related attributes, and investigate how they are related to each other.

**Inter-observer agreement**: Measures the consistency of observer judgments. The well known Root Mean Square (RMS), also known as Root Mean
Square Deviation (RMSD) or Root Mean Square Error (RMSE), is used for measuring the agreement by averaging the RMS between each observer’s judgment and the overall mean, defined as

\[ RMS = \sqrt{\frac{\sum (x_i - \hat{x}_i)^2}{N}} \quad (4.1) \]

where \( x_i \) represents the scale position given by an observer for sample \( i \), \( \hat{x}_i \) represents the mean scale position of all observers for sample \( i \), and \( N \) represents the number of samples. If one wants to compare RMSs with different units (different scale, different amounts of categories, etc.) the RMS should be normalized, for instance by calculating the Normalized Root Mean Square

\[ NRMS = \frac{RMS}{x_{max} - x_{min}} \quad (4.2) \]

where \( x_{max} \) and \( x_{min} \) are maximum and minimum of possible scale values.

The NRMS can also be used for calculating the **Intra-sample agreement**, a measure of how judgments concerning specific samples are spread along the scale. Eq. (4.1) and (4.2) are used, but with \( i \) representing different judgments instead of samples, and \( N \) represents the number of observers.

**Correlation coefficient**: For evaluating the correlation between observer judgments and for instance predicted values, the *Pearson product-moment correlation coefficient* \( r \) is frequently used. The coefficient is defined as

\[ r = \frac{1}{N-1} \sum_{i=1}^{N} \left( \frac{x_i - \hat{x}}{s_x} \right) \left( \frac{y_i - \hat{y}}{s_y} \right) \quad (4.3) \]

where \( x_i \) and \( y_i \) are positions on the emotion scale, \( \hat{x} \) and \( \hat{y} \) are mean positions, and \( s_x \) and \( s_y \) are standard deviations of all positions for observer judgments and predicted values respectively. \( N \) represents the number of samples. The correlation coefficient measures the strength and direction of a linear relationship between sets of variables, for instance judgments and predictions. For a perfect increasing (or decreasing) linear relationship the correlation is 1 (or -1). If variables are independent then the correlation is 0.

### 4.4 Color Emotions

Color emotions can be described as emotional feelings evoked by single colors or color combinations, typically expressed with semantic words, such as "warm", "soft", "active", etc. Color emotions together with color meaning, color harmony, color memory etc., belong to the cognitive aspects of color. Psychophysical experiments are frequently used for emotional classification of colors or for creating color emotion models that can be used for predicting color
emotions. This thesis investigates the possible usage of a color emotion model in image retrieval. In a series of papers [61][62][63], Ou et. al. investigated the relationship between color emotion and color preference. Color emotion models for single-colors and two color-combinations are derived from psychophysical experiments. Observers were asked to assess single colors on ten color emotion scales. They show that factor analysis can reduce the number of color emotions scales to only three categories, or color emotion factors: activity, weight and heat, defined as

\[
\text{activity} = -2.1 + 0.06 \times \left[ (L^* - 50)^2 + (a^* - 3)^2 + \left( \frac{b^* - 17}{1.4} \right)^2 \right]^{\frac{1}{2}} \tag{4.4}
\]

\[
\text{weight} = -1.8 + 0.04(100 - L^*) + 0.45 \cos(h - 100^\circ) \tag{4.5}
\]

\[
\text{heat} = -0.5 + 0.02(C^*)^{1.07} \cos(h - 50^\circ) \tag{4.6}
\]

\[
h = \arctan\left( \frac{b^*}{a^*} \right) \tag{4.7}
\]

\[
C^* = \sqrt{a^{*2} + b^{*2}} \tag{4.8}
\]

$L^*$, $a^*$ and $b^*$ are CIELAB coordinates, $h$ is CIELAB hue angle and $C^*$ is CIELAB chroma. For readers not familiar with the CIELAB color space, a detailed description of CIE color spaces can be found in Fairchild [20]. Ou et. al. conclude that the three factors agree well with studies done by others, for instance Kobayashi [40] and Sato et. al. [68]. The presented color emotion model was not developed for multi-colored images of potentially very complex structure. It is therefore easy to construct images were it will fail. We will, however demonstrate that the results obtained show that these techniques provide useful statistical characterizations for Content Based Image Retrieval.

### 4.5 Color Emotions for Images

Since the intended usage is retrieval or classification in very large image databases, two special requirements are present. Apart from fast feature extraction, the derived image features needs to be saved in compact descriptor vectors. Using RGB-histograms for measuring statistical properties of color images is very common in Content Based Image Retrieval. We make use of this and propose a method that will transform ordinary RGB-histograms of images to emotion descriptors. Typically the histograms consist of 512 entries with eight quantization levels (bins) per color channel. We collect RGB-histograms for all images in the database and save them in a matrix $H$ of size $N \times 512$ (rows $\times$ columns), where $N$ is the number of images in the database. For
each bin in the RGB-histogram we calculate the corresponding color emotion vector, using the equations in section 4.4. These 512 three-dimensional vectors are collected in the color emotion matrix $E$ of size $512 \times 3$ (three emotion values for each bin). In Fig. 4.1 the locations of the 512 RGB-bins are shown in the three dimensional space spanned by the three emotion factors. We can observe that although some colors have very different RGB-values they generate similar emotions. Since results obtained in this paper will be implemented in a public search engine, we have no control of the users viewing conditions, and are consequently forced to make some assumptions. In all calculations we assume images are saved in the commonly used sRGB color space, and we use the standard illumination D50 when transforming sRGB values to CIELAB values.
4.5 Color Emotions for Images

4.5.1 Retrieval by emotion words

Multiplying the histogram matrix $H$ with the color emotion matrix $E$ we obtain a matrix $C$

$$C = H \cdot E$$  (4.9)

of size $N \times 3$. The $n$-th row in the histogram matrix $H$ describes the probability distribution of the RGB vectors in image $n$. We denote this row by $h_n$ and see that the $n$-th row (denoted by $c_n$) in the matrix $C$ is obtained by the scalar product $c_n = h_n \cdot E$. The vector $h_n$ is a probability distribution and $c_n$ is thus an expectation vector describing the expected value of the color emotion vectors of the pixels in image number $n$ in the database. This vector contains thus the mean score for each of the emotion factors: activity, weight and heat.

In Fig. 4.2, 30 images are plotted according to their emotion coordinates in the three-dimensional emotion space. This figure shows that the emotion coordinates bring together images of similar emotional impact and we can therefore use the emotion coordinates as search interface. A query is constructed by selecting a position on one, two or three emotion scales (activity, weight and heat). For a given query vector of emotion scores we retrieve the images whose emotion vectors are nearest neighbors of the query vector in the $L_2$ norm. One consideration is that an image containing for instance warm colors in the left half, and cool colors in the right half, will be given a neutral position on the heat scale. A possible question to be dealt with in future work is whether users find this acceptable or not. However, by using retrieval by query image, presented in the next section, the problem can be avoided.

4.5.2 Retrieval by query image

Instead of creating a query by selecting values on three emotion scales, the user can submit a query image to obtain images from the database with similar emotional appearance. Since Fig. 4.1 demonstrates that the RGB space is unevenly spaced in emotion space, we try to avoid the problems encountered by using more than a single emotion coordinate. We use a kd-tree decomposition to obtain a more balanced decomposition of the 512 coordinates in the emotion space given by the 512-bins RGB histogram. We split emotion space perpendicular to one of the axes (activity, weight or heat). Cycling through the axes, splitting each new region by the median, the result will be a distribution with equal number of entries in each region, generally called a balanced tree with equal number of entries in each leaf. We continue splitting until we get 64 leaves, each containing 8 entries. This is the color emotion histogram with 64 bins. In other words, 8 different bins from the original RGB histogram will be included in each bin in the emotion histogram. The splitting procedure is illustrated in Fig. 4.3, and the result from splitting the 512 bins RGB histogram can be seen in Fig. 4.4. Each color represents a leaf, which corresponds to a
Figure 4.3: A simple illustration of the decomposition of histogram bins. For viewing purpose, only two color emotions are included in this illustration. In the upper left image the initial data, this time 16 dots (bins), are displayed. In the upper right image we perform the first decomposition by splitting the data perpendicular to the Color Emotion 1 axis. In bottom left image we perform the second decomposition by splitting each new data region perpendicular to the Color Emotion 2 axis. Then we continue cycling through the axes and split new regions until we get the preferred amount of leaves. In this example eight leaves, each containing 2 bins, as shown in the bottom right image.

Figure 4.4: Visualization of the color emotion histogram. Same points as in Fig. 4.1, but groups of 8 are coded with different colors. Each group represents a bin in the color emotion histogram.
4.6 Psychophysical Evaluation

Knowing which color emotion value belongs to which leaf, or bin in the color emotion histogram, we can create a matrix $T$ transforming 512 bins RGB-histograms to 64 bins color emotion histograms. $T$ will have the size $512 \times 64$ ($rows \times columns$), where each column, denoted $t_n$, will generate bin number $n$ in the emotion histogram. Each column $t_n$ contain zeros except for the eight positions that will be included in bin $n$ in the emotion histogram. After multiplying the histogram matrix $H$ with $T$

$$R = H \cdot T$$

we obtain a matrix $R$, where each row is a 64 bins color emotion histogram, describing the distribution of emotion vectors in the space spanned by the three emotion scales. The $L_2$ norm is used for calculating the similarity between histograms in the database and the emotion histogram derived for the query image.

4.5.3 Results

A test database was created, containing 5000 images. These are randomly selected from a much larger image database used in previous research. The test database contains different image categories, both photos and graphics. The first search mode: retrieval by emotion words, is illustrated in Figure 4.5. A search interface is used, where the query vector is specified with the help of 1-3 sliders, corresponding to different emotions. The second search mode: retrieval by query image, is illustrated in Figure 4.6.

Objective evaluations (like measuring Precision and Recall) are difficult to design since there is no ground truth. However, psychophysical experiments are used for evaluating the method, as described in the next section.

4.6 Psychophysical Evaluation

In this section psychophysical experiments are used for evaluating the emotion based retrieval method.

4.6.1 Limitations

The main goal with our research is to develop methods that can be implemented in search engines for large image databases, especially search engines that are publicly available on the Internet. In other words, the findings of this study will be used in a situation rather different from the situation encountered in traditional color emotion experiments. Consequently, conditions completely different from previous experiments should be considered, as described below.
Figure 4.5: Search using emotion words. The sliders show the values of the emotion words, the images are the five best matches for each query.
The main concern is that, since results will be used on the Internet, we have no control over the user environment. An unknown adaptation is present, which may influence the result in many ways. For instance, users may have completely different monitor settings. Also the viewing environment, like illumination settings, background colors, etc. may vary significantly. Moreover, the psychological state of the user is unknown. We might have a different response from a relaxed user than from someone who is stressed. Since related research about color emotion models is usually carried out in controlled environments, it is of course questionable if the same models can be applied in an uncontrolled environment. However, we believe that if the entire evaluation process is based on user tests on the Internet, where we have no control over environmental settings, and still the results are successful, then we can also apply the models in an uncontrolled environment.

Another factor to be aware of is that image content in general, not only the color content, will probably affect the user’s emotional response. Some images may even have strong emotional content closely related to those emotions evoked by the color content. For instance, some may feel that a photo of a sun-drenched beach brings a warm feeling, even though the color of the water can be perceived as rather cold. And observers familiar with cold climates may perceive a photo containing snow and ice as cold, even if the actual color content is rather neutral. But as long as the user is aware of, that the emotional search is purely based on color content, we believe most users will both notice and accept images containing other emotional content. Finally, since tests are carried out anonymously on public web pages, we have to face the risk that some users will deliberately submit answers disagreeing with their own emotional feelings.
4.6.2 Methods

When designing user tests, we followed to a large extent the methodology presented in Engeldrum [18]. The study consists of two tests, following each other in time. First a pilot study is conducted, that will indicate if we are aiming in the right direction. The study will also help us to select images, from now on called samples, for the second test. The pilot study is designed to be as simple as possible for the observer to perform. The second test, here called the main study, uses samples earlier judged in the pilot study. It is slightly more demanding for the observer. The advantage though is that the results are more reliable, and easier to evaluate. Our final conclusions are mainly based on the main study. Both tests are using web pages for communicating with the observer. The reason for describing both the pilot study and the main study is that the main study is to some extent based on results and experience gained in the pilot study.

The Pilot Study

In this study observers are judging one sample at a time, using a category scaling method. Samples are random and independently selected from the database of 5000 images, containing both photos and graphics. For those samples that are selected, the observer is asked to judge the sample on different emotion-related scales. For simplicity, each scale is divided into five equal-appearing intervals, or categories, and the observer can only pick one of them. The entire graphical user interface, including user instructions, can be seen in Fig. 4.7. The maximum sample size, height or width, is 200 pixels. Advantages with this method are that observer instructions are rather easy to understand, and observers are not forced to judge an entire set. They can quit judging samples whenever they want. However, the simplicity will bring some negative aspects as well. Foremost, the observer criterion may drift (move along the emotion scale), especially for the first few images that are judged. The reason is that observers are not familiar with the full range of the scale until they have seen relatively many samples. Another drawback is that we only obtain a rough estimation of visual distance between samples since the interval spanned by each category is rather large. And finally, when judging samples, observers tend to use all categories equal number of times (according to Engeldrum [18]), which may not resemble the reality since samples are displayed randomly.

The Main Study

An interval scaling method is utilized in this study, which, compared to the pilot study, will increase the accuracy of the emotional distance. Observers are shown a set of samples, and then asked to place all samples on a ruler ranging from one emotion attribute to another, for instance cool to warm. Observers
4.6 Psychophysical Evaluation

Figure 4.7: The Graphical User Interface for the pilot study. (Results from the harmony factor will be presented in future publications)

Figure 4.8: The Graphical User Interface for the main study. Observers are asked to place each set of samples on the emotion ruler below. They are instructed that the distance between samples should correspond to the emotional difference.
are also instructed to place the samples so that the distance between samples corresponds to the emotional difference. Samples with the same emotion response can be positioned above, or on top of each other. Totally 10 samples are used for each emotion scale. The selection of samples is based on judgments obtained in the pilot study. As mentioned earlier, in the pilot study each emotion scale was divided into five intervals or categories, and observers were assigning samples to those categories. From the set of samples assigned to each category, two samples are randomly drawn, giving us totally 10 samples for each emotion scale. Using this selection should provide a more homogeneous coverage of the emotion space than a random selection. Even if a full coverage is not necessary (observer judgments can still be evaluated for parts of the emotion space), it is of course desirable to be able to evaluate the entire emotion space within the current study. The selection procedure is an important and challenging task that is further discussed in section 5.2.

The user interface for the main study, including selected samples, can be seen in Fig. 4.8. As shown, by coincidence one sample was actually drawn for both heat and weight. Examining the samples visually, it looks like none of them contains a strong emotional content not related to color (the author’s opinion), as discussed in previous section. Using the mouse pointer, observers can click on images and drag them to desired positions. Here the maximum sample size, height or width, is 70 pixels. The reason for using smaller samples than in the pilot study is only to make sure that all samples and the entire scale fits within the observer’s browser window. This method demands from the observer that a complete set of samples should be judged, and the observer also needs to consider the emotional distance between samples.

An important consideration in the design of psychophysical experiments is the "rubber band effect". This means that most interval scales involve two arbitrary constants, a multiplier and some additive constant. To overcome this problem and enable easier comparisons between judgments from different observers, the best-known method (according to Engeldrum [18]) is to adjust judgments until they have the same mean and variance. In other words, observer judgments are calibrated to a common scale. For each judgment \( j \), containing \( i \) samples, the mean is subtracted, and the result is divided by the standard deviation, defined as

\[
a_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}
\]

where \( x_{ij} \) is the position of sample \( i \) in judgment \( j \), \( \bar{x}_j \) is the mean position for all samples in judgment \( j \), \( s_j \) is the standard deviation for all samples in judgment \( j \), and \( a_{ij} \) is the resulting position for sample \( i \) in judgment \( j \). An alternative approach is to select a sample from each end point of an emotion factor (e.g. most cool and most warm) and use them as anchors. Then each judgment is normalized based on anchor positions. However, as mentioned
above, two samples were randomly selected from each end point category. Using only one of them as anchor will probably influence the result depending on which one we choose. But more important, at this stage in the study we cannot know if the selected sample is a representative observer judgment suitable to be used as anchor sample. Moreover, judgments made in the pilot study cannot be fully trusted since the study was conducted on a public web page. Therefore the normalization by mean and variance is adopted.

4.6.3 Results

The Pilot Study

The web address of the page containing the pilot study was sent to a number of recipients, including colleagues, students and friends, both males and females ranging from approximately 20 to 65 years old. The address was also displayed on a public web page, probably generating a few more unknown participants. Totally 52 observers participated in the study, judging in total 567 samples (as described earlier, these are random and independently selected from a larger database containing 5000 images). The number of observers is based on a measurement of the number of unique computers used when judging images. Several observers may have used the same computer, or a single observer may have used more than one computer. However, we assume that the consequences on the overall result are of minor importance and can be ignored. In Fig. 4.9, observer judgments for heat, weight and activity are plotted against emotion values derived using the method described earlier in this thesis. For both heat and weight a clear relationship between judgments and derived values can be observed. For activity the relationship is weaker, only samples that are judged to have maximum activity are somewhat distinguishable from remaining ones. The figure also shows how many samples were assigned to each category. Apparently, “neutral” samples (close to the center of each scale) are more common than “extreme” samples (end points).

The Main Study

The address to the web page containing the main study was also sent to a number of recipients, but not displayed on a public web page. The group of recipients was comparable to the group in the pilot study: colleagues, students and friends, both males and females ranging from approximately 20 to 65 years old. However, most of the recipients did not earlier participate in the pilot study (although we cannot guarantee that they didn’t visit the public web page for the pilot study). Since the presumed numbers of observers were lower than in the pilot study, participants were asked to judge samples only once to avoid single observers influencing the data too heavily. Totally 35 observers participated. The response from two observers were removed because they
Figure 4.9: Observer judgments for heat, weight and activity from the pilot study are plotted against predicted emotion values. The x-axis represents judgment categories, and the y-axis represents predicted values. For each category, the short horizontal line corresponds to the mean of the predicted values, the vertical line corresponds to the standard deviation, and dots correspond to the minimum and maximum among predicted values for samples in that category. The numbers along the dotted baseline shows how many samples that were assigned to each category.
4.6 Psychophysical Evaluation

pushed the submit button before moving any of the images (we assume that was a mistake). This leaves 33 observer judgments for the evaluation process. In Fig. 4.10 all judgments are shown, for heat, weight and activity respectively. The mean and variance are used for scaling each judgment. As mentioned earlier, samples are selected based on judgments obtained in the pilot study.

For measuring inter-observer agreement we adopt the method and terminology used by Ou and Luo [60]. The inter-observer agreement measures the consistency of observer judgments, in other words how well observers agree with each other. The agreement can be derived by averaging the Root Mean Square (RMS), also known as Root Mean Square Deviation (RMSD) or Root Mean Square Error (RMSE), between each observer’s judgment and the overall mean. The RMS is derived by

\[ RMS = \sqrt{\frac{\sum (x_i - \bar{x})^2}{N}} \]  
(4.12)

Figure 4.10: Observer judgments for heat, weight and activity. Judgments are scaled to have the same mean and variance.
where $x_i$ represents the scale position given by an observer for sample $i$, $\hat{x}_i$ represents the mean scale position of all observers for sample $i$, and $N$ represents the number of samples. However, what is not mentioned by Ou and Luo [60] is that if one wants to compare RMSs with different units (different scale, different amounts of categories, etc.) the RMS needs to be normalized, for instance by calculating the Normalized Root Mean Square

$$NRMS = \frac{RMS}{x_{max} - x_{min}}$$

(4.13)

where $x_{max}$ and $x_{min}$ are maximum and minimum of possible scale values. Inter-observer agreements obtained are 0.099 for the heat factor, 0.079 for weight, and 0.160 for activity. As expected from earlier results, the agreement is better for heat and weight than for activity. Notice that a good agreement corresponds to a low value, and vice versa. The agreement values obtained are similar or even lower than results presented in related studies (for instance Ou and Luo [60]), indicating that the agreement is satisfactory. Since each observer only judges one set of images we do not need to consider the intra-observer agreement.

The next step is to compare the observers emotional response to the proposed predictive model used in image retrieval. In Fig. 4.11, judgments are plotted against emotion values derived using the predictive model. Diamonds correspond to mean positions for each sample. For each diamond there is a distribution of dots (on the same horizontal level) showing individual judgments. The evaluation of the predictive model uses the Pearson product-moment correlation coefficient $r$ between observer judgments and predicted values, for both mean positions (diamonds) and all positions (dots) in each emotion scale. The correlation coefficient is defined as

$$r = \frac{1}{N-1} \sum_{i=1}^{N} \left( \frac{x_i - \bar{x}}{s_x} \right) \left( \frac{y_i - \bar{y}}{s_y} \right)$$

(4.14)

where $x_i$ and $y_i$ are positions on the emotion scale, $\bar{x}$ and $\bar{y}$ are mean positions, and $s_x$ and $s_y$ are standard deviations of all positions for observer judgments and predicted values respectively. $N$ represents the number of samples. The correlation coefficient measures the strength and direction of a linear relationship between sets of variables, this time judgments and predictions. For a perfect increasing (or decreasing) linear relationship the correlation is 1 (or -1). If variables are independent then the correlation is 0. Heat and weight resulted in correlation coefficients as high as 0.97 and 0.98. For activity the correlation is 0.86. The results obtained can also be found in Table 4.1. The last column contains $R^2$ values (the square of the multiple correlation coefficient), which can easily be obtained from $r$ since the correlation is attained from a simple linear regression ($R^2 = r \times r$). $R^2$ values indicates how much
4.6 Psychophysical Evaluation

Figure 4.11: Observer judgments for heat, weight and activity from the main study are plotted against predicted emotion values. The x-axis represents positions on the emotion ruler, and the y-axis represents predicted values. Diamonds correspond to mean positions for each sample. For each diamond there is a distribution of dots (on the same horizontal level) showing individual judgments. For increased readability, the plot area is larger than the possible value interval ([0 1] for both the x- and y-axis)
Table 4.1: Pearson $r$ and $R^2$ correlation between observer judgments and predicted values.

<table>
<thead>
<tr>
<th>Emotion factor</th>
<th>Pearson $r$</th>
<th>$R^2$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean pos.</td>
<td>All pos.</td>
</tr>
<tr>
<td>HEAT</td>
<td>0.97</td>
<td>0.90</td>
</tr>
<tr>
<td>WEIGHT</td>
<td>0.98</td>
<td>0.93</td>
</tr>
<tr>
<td>ACTIVITY</td>
<td>0.86</td>
<td>0.68</td>
</tr>
</tbody>
</table>

of the variation in the judgments that can be predicted by the model (e.g. $R^2 = 0.7$ means that the model explains 70% of the variance). Using the rather simple correlation coefficient $r$ for measuring correlation can probably be questioned. However, in this study an evaluation method invariant to scaling and translation of variables is desired, and consequently we find the correlation coefficient suitable. Remember that we want to measure the association between two variables, not the exact agreement. In addition, considering previously published papers within this research area we notice that using the correlation coefficient enables us to compare results with others. Still, it is not obvious how to draw conclusions from single correlation values. Anyway, by comparing our results with results reported in related studies ([61][24][99][60] to mention a few) we conclude that obtained correlation values lie within the range of what in general is considered to be a good correlation. Especially the results for heat and weight show a very good correlation.

For comparison, in the pilot study the correlation coefficient was 0.58 for heat, 0.52 for weight, and 0.14 for activity. This confirms that the method of category scaling is much less reliable than interval scaling. In addition, we suspect that some of the data in the pilot study is corrupt since the user test was conducted on a public web page.

The relationship between the inter-observer agreement and the Pearson correlation for heat, weight and activity can be seen in Fig. 4.12. The result shows that there is a connection between poorer inter-observer agreements and lower correlation coefficients. We may even suspect that a large amount of the discrepancy in the correlation coefficient, especially for activity, is due to observers varying opinions, and not due to limitations in the predictive model.

Intra-sample agreement (for finding problematic samples)

Knowing samples with observer judgments that are highly inconsistent can be of importance if one wants to improve the predictive model. The Normalized Root Mean Square is used for calculating the intra-sample agreement, a measure of how judgments concerning specific samples are spread along the emotion scale. Eq. (4.12) and (4.13), presented earlier, are used for calculating the intra-sample agreement, but with $i$ representing different judgments instead of
samples. Consequently, $N$ represents number of observers. Obtained results for sample 1-10 in all three emotion scales are plotted in Fig. 4.13. The mean intra-sample agreements are 0.10 for heat, 0.08 for weight, and 0.17 for activity. The result corresponds well with values obtained for the inter-observer agreement, reinforcing that the activity factor is harder for observers to comprehend and apply on multi-colored images than the heat and weight factors. On the other hand, the result may also show that observers have slightly varying opinions about activity for samples.

The three samples from each emotion factor that obtained the worst intra-sample agreements are plotted in Fig. 4.14. We can observe a mixture of both highly abstract graphical images and ordinary photos. Samples with poor intra-observer agreement cannot be visually distinguished from the remaining ones (all samples can be seen in Fig. 4.8). However, more research is needed to find out if those samples can be distinguished by some currently unknown statistical or computational properties.

### 4.7 Online Implementation

The color emotion based image retrieval system presented in this thesis is implemented in a public demo search engine called ImEmo - Color Emotions for Image Retrieval\(^2\). Since the search engine is a demo, only 5000 images are included in the public version. But the same methods can easily be applied in

\(^2\)http://media-vibrance.itn.liu.se/imemo/
Figure 4.13: Intra-sample agreement for different samples and emotion factors. Notice that even if samples from different emotion factors have the same sample number, the image content is not the same. See all samples in Fig 4.8 (sample number 1 to 10 from left to right).

Figure 4.14: The three images from each emotion factor that obtained the worst intra-sample agreements.
4.7 Online Implementation

much larger databases, containing millions or billions of images. This section will briefly describe the demo system setup, followed by some user statistics.

4.7.1 System setup

Interaction with the user is carried out through a web front page implemented in the scripting language PHP. Through a socket connection, PHP is communicating with MATLAB\(^3\), which is running as an internal server in the background\(^4\). The MATLAB environment is hosting all components dealing with calculation and comparison of features, and data storage. The main reason for using MATLAB is to save time in the implementation process (since we are only creating a demo). However, all methods can easily be implemented in most programming languages.

The communication between the user and the search engine is illustrated in Fig. 4.15. A query can either be an image, or emotion values on 1-3 emotion scales. If a query image is used, the user submits an image URL. The socket connection is used for sending the address to MATLAB, where the recognition unit computes emotion features for the image, and compares the feature vector to vectors in the database. The search result is returned to the PHP front-page where a list of images is presented. When emotion scales are used for searching, the approach is very similar, except that slider positions on selected emotion scales are submitted to MATLAB and compared to emotion values in the database. A rendering of the PHP frontpage (the Graphical User Interface) can be seen in Fig. 4.16. Notice that retrieved images can be used in the next query, enabling browsing of the database.

Offline tools have been created for building the feature database. When adding new images to the database, corresponding feature vectors are simply added to new rows in the database. Consequently, the recognition time (\(O(N)\)) will increase linearly with new images added to the database. If the methods are utilized in a much larger image database one should consider using a tree-structure or other multi-dimensional indexing techniques for saving and searching the database.

4.7.2 User statistics

This section will review some general search engine statistics gathered during a period of 14 months (October 2008 - November 2008). The front page has in total collected 403 visits, from 31 different countries/territories. Out of 403

\(^3\)MATLAB is a high-level technical computing language and interactive environment for algorithm development, signal and image processing, data visualization, data analysis, and numeric computation. See: http://www.mathworks.com/

\(^4\)Running MATLAB as a server can be accomplished by using for instance the "TCP/UDP/IP Toolbox" by Peter Rydesäter, available through the MATLAB Central: http://www.mathworks.com/matlabcentral/fileexchange/345
visits, the unique number of visitors is 152. On average, a user continues using the search engine for 4 minutes and 28 seconds, performing on average 7 queries. The main traffic source is direct traffic (43%), closely followed by referring sites (42%), and search engines (15%). Fig. 4.17 tries to give an overview over what parameters each query was based on. The bar graph reveals for instance that 27% of the queries are based on an emotion value from a single emotion factor (activity, weight or heat). For 12% of the queries, values from all three emotion factors are combined in the query. As much as 51% of the queries are based on internal images (images that are included in the demo search engine and displayed in search results). A possible explanation is that people tend to "browse" the database by clicking within search results.

Figure 4.15: The communication between the user interface (implemented in PHP) and the search engine (implemented in MATLAB).
Figure 4.16: The Graphical User Interface for the search engine. The user can either select retrieval by emotion words, or retrieval by query image.
Figure 4.17: On overview over what parameters different queries were based on. Bar 1-7 correspond to different selections of emotion sliders. For instance, bar 4 reveals that 3% of the queries were based on a combination of emotion values defined by the activity and weight scales. Internal images are images included in the demo search engine, and external images are image URLs pointing at images on other web pages.
Chapter 5

Conclusions

5.1 Font Retrieval

A search engine for very large font databases has been presented and evaluated. The search engine operates on fonts for the English alphabet. The input is an image with a text from which the search engine retrieves the name of the font used when printing the text. Here we focused on recognition and retrieval from very large font databases, but the search engine can also be used as a pre-processor to OCR. The method, based on eigenimages calculated for different characters and fonts, can be classified as a local approach since features are calculated for individual characters. Prior to calculating eigenimages, character images are scaled and filtered with one horizontal and two diagonal edge filters. Even for the very large font database, containing 2763 fonts, the retrieval accuracy is very high. For individual characters, the mean accuracy for a perfect match is 88.2%, and the probability to find the correct font name within the five best matches is 99.1%. These values are for single characters. In practice, the overall accuracy can increase since the search engine is capable of working with text lines, with the opportunity to combine the result from many characters. To resemble a real life situation, retrieval experiments were made with printed and scanned text lines and character images from the original database. The method has proven to be robust against various noise levels and image quality settings, and rotation/scaling of the input image. Moreover, it can handle both overall shape and finer details. However, the main advantages with the proposed method are that it is simple to implement, features can be computed rapidly, and descriptors can be saved in compact feature vectors, which is essential when working with very large databases.

The recognition accuracy obtained is higher than the accuracy achieved with the best performing region-based method evaluated in Larsson [41]. In that thesis, a four level kd-tree decomposition, with second order moments,
achieve a mean perfect match accuracy for characters 'a' and 'b' of 81%, and a top five accuracy of 96%. The corresponding mean scores for characters 'a' and 'b', for the proposed eigenimage method using exactly the same evaluation process, is 92% and 100% respectively.

The proposed method was implemented in a public search engine holding 2364 non-commercial fonts. User statistics gathered during a period of 20 months reveal, for instance, that many users are submitting single characters to the search engine, and many users are using the search engine as a browser by utilizing retrieval results as the next query. Obtained retrieval results together with the online implementation show that the method is appropriate for font recognition in very large font databases.

In addition, initial investigations concerning a search and visualization tool for large font databases were described. Proposed methods use shape descriptors combined with linear and non-linear dimensionality reduction methods and a post-processing ordering algorithm to handle the visual content of the characters in large font databases. It was shown that this tool is useful for a systematic visualization of the database content and we expect it to be a powerful design tool for publishing and graphic arts applications.

5.2 Color Emotions in Image Retrieval

The findings of this study show that using color emotions metrics in content-based image retrieval results in new and interesting methods for image retrieval and classification. The color emotion metric, derived from psychophysical experiments, is based on three scales: activity, weight and heat. It was originally designed for single-color combinations, but is also capable of predicting color emotions for color pairs by averaging the color emotion scores of individual colors in each pair. Experiments demonstrate that the same approach can be used for images. The color emotion of a whole picture is derived as the sum or average of all color emotions from its pixels. A modified approach for statistical analysis of color emotions in images, involving transformations on ordinary RGB histograms, proves to be powerful enough to provide a useful tool for image retrieval. The method is both very fast in feature extraction, and the resulting descriptors are very small, which is essential since we are interested in retrieval from very large image databases containing millions or billions of images. Two retrieval methods were presented: one method based on the selection and weighting of color emotion words and the other using color emotion histograms. These are distributions of vectors in the space spanned by the emotion scales. Similarity measurements between histograms are used to obtain images with similar emotional appeal as the query image.

One of the novelties in this study is that an extensive user study complements the retrieval model. In a psychophysical evaluation it was shown that
people in general perceive color emotions for multi-colored images in similar ways, and that observer judgments highly correlate with the predictive model used in image retrieval. Images were judged by observers on the three emotion factors: heat, weight and activity. Obtained inter-observer agreements are 0.099 for the heat factor, 0.079 for weight, and 0.160 for activity. Comparing the results with other studies involving single colors or two-color combinations shows that the agreement is good, especially for heat and weight. Notice that a good agreement corresponds to a low value, and vice versa. For measuring the correlation between observer judgments and predicted values the Pearson product-moment correlation coefficient $r$ was used. Heat and weight obtained correlation coefficients as high as 0.97 and 0.98. For activity, the correlation is 0.86, but still within the range of what in general is considered a good correlation. A poorer inter-observer agreement for the activity factor has a negative influence on the correlation. Also the intra-sample agreement for heat and weight is better than for activity, indicating both that the activity factor is harder for observers to comprehend, and that observers may have slightly more varying opinions about the activity factor than the other two factors. No visual difference could be established between images with high and low intra-sample agreement.

The selection of samples in the main study is an important and challenging question. However, since this is the first study within this subject, it is difficult to decide how to select an ideal set of samples. One attempt could be to create a set that is representative for the database used in the target application. But the subsequent question is then: Representative in what way? The selection can be based on many different measurements, like emotional properties, the color content, etc. As shown in Fig. 4.9, the results obtained in the pilot study indicate that emotionally "neutral" samples (located in the middle of the emotion scale) are more common than "extreme" samples (located at end points).
That this is a realistic assumption is confirmed in the following preliminary experiment where we used 249,587 sample images from the image search engine Picsearch (www.picsearch.com). For these images we computed emotion values and histograms describing the distributions of emotion values. The histograms are plotted in Fig. 5.1. Also for this database, collected from the Internet, it is obvious that "neutral" samples are more common than "extreme" samples. A set of samples emotionally representative for the entire target database would thus contain more "neutral" samples than "extreme" samples. However, if the findings are implemented in an image retrieval system, we expect that users utilizing the emotion search are mainly interested in samples with strong emotional content. Consequently it might be beneficial to develop and evaluate the method with a sample set containing equal number of "neutral" and "extreme" samples, or even more "extreme" samples than "neutral" ones. With this in mind we conclude that the sample selection method used in this initial study was appropriate, and that the obtained knowledge can be used for refining the selection method in upcoming research.
Chapter 6

Future Work

6.1 Font Retrieval

Even if the proposed method is already suitable to be used in a public search engine, some ideas on how to improve the system should be considered in future work. For instance, how individual characters are weighted and combined to a final result is an unsolved issue. The easy solution would be to let the weighting depend on the search performance for different characters (see Section 3.6). However, a more delicate solution would be to take into consideration which characters users prefer to search with, or which characters users think are most important. Moreover, since users tend to submit rather ordinary looking fonts one can also think of utilizing an OCR system for helping the user assigning characters to segmented images. Finally, the possibility to use the search engine as a browser (by re-using retrieval results) should be developed further. For instance, one can let the user select several appealing fonts from one or many retrieval results, and search for a new font based on a weighting of selected fonts.

The results from the visualization of the font database can be considered a starting point for future research. For instance, it is desirable to extend the method to utilize more than one character in the visualization. Moreover, the proposed method should be evaluated in user tests to clarify if people find the system useful, and how they interact with it. Another topic do be dealt with in future work is the method for dimensionality reduction. A lot of powerful tools for both linear and non-linear dimensionality reduction have been presented in the past. Preliminary experiments with for instance Self-Organizing Maps (also known as the Kohonen Maps), reveal no improvements compared to the PCA or ISOMAP approach. Nevertheless, the use of different dimensionality reduction tools should be further investigated in future research. A fine-tuning of training parameters or other ways of pre-processing the data might produce
visualization results comparable to the ISOMAP approach. One can also think of using other methods as a pre-classifier, and then for instance apply ISOMAP on individual classes to obtain a final visualization grid.

6.2 Color Emotions in Image Retrieval

Today, the focus of image retrieval research is on scene and object recognition. However, we predict that one of the upcoming great challenges in Content Based Image Retrieval will be the understanding of emotion-related attributes, like "warm", "cold", "happy", "harmony", etc. The presented study is the first step towards a broader use of emotion related properties in Content Based Image Retrieval. Upcoming research will focus on including additional emotion properties, and how to efficiently communicate those properties in a user interface for image retrieval. One of several emotion properties that will be investigated further is harmony. Preliminary experiments indicate that it is possible to create models that can predict if an image will be perceived as harmonic or not, and even predict the amount of harmony. For the presented study in particular, one thing to consider is whether the activity factor should be altered to some factor that users more easily comprehend, resulting in better inter-observer agreement.

An important research topic for future research is the sample selection task. Results obtained in this study can help to improve the selection in upcoming experiments. Related to the sample selection is the detection of images with problematic content. It is obvious that for certain samples the accuracy of the predictive model will decrease. Examples are images containing a strong emotional content not related to colors (as discussed in section 4.6), or images containing a mixture of emotions (for instance warm colors in the left half, and cool colors in the right half). The question is whether these samples can be distinguished beforehand. One approach is to explore potential relationships between statistical properties of images and the intra-sample agreement obtained from observer judgments. One can also, for instance, try to find samples with a strong emotional content not related to colors, or create custom made images with a mixture of emotions, and try to approach the problem by comparing observer judgments for those samples with other more common samples.

Classic image retrieval based on numerical values without the semantic interpretation, popular in the beginning of the CBIR era, can according to Gentner and Grudin [26] be called the "engineering model" of retrieval. Some users (engineers?) will find the model useful, others will not. Consequently, fundamental investigations on how people would like to search for images need to be addressed in future research. Meanwhile, the presentation and evaluation of all kinds of image retrieval models are of importance.
Bibliography


