MASARYK UNIVERSITY
FACULTY OF INFORMATICS

Improving Quality of Content-Based Image Retrieval

DISSERTATION PROPOSAL

Mgr. Petra Budíková

Supervisor: Prof. Ing. Pavel Zezula, CSc.

Brno, January 2010

Supervisor

Contents

Chapter 1

Introduction

The management of digital information has always been one of the key tasks of computer science. In the early decades, when most of the data consisted of text and numbers, relational databases handled the storage and searching well. However, with the rapid growth of more complicated data types, such as images, sounds or video, new approaches to searching are needed that suit our changing needs better. There are two major differences from traditional searching: first, many of the new domains are not linearly sortable, which puts standard indexing methods such as B-trees out of question. Second, users' approach to searching has changed. In the huge amounts of complex data, a search for exact matches to a given query is often not meaningful, whereas proximity concepts (similarity, dissimilarity) provide a useful tool for browsing and searching in the dataset.

To address these requirements, a new approach to data management has been intensively studied in the last two decades. The *content-based* or *similarity searching* has become a fundamental computational task in a variety of application areas, including multimedia information retrieval, data mining, pattern recognition, biomedical databases, data compression and statistical data analysis. The development of this research area proceeds on many levels, from techniques tailored to a very specific data and for a specific application to solutions based on general models and solid theoretical grounds that are applicable to a variety of data types.

1.1 Challenges of Similarity Searching in Image Retrieval

Content-based image retrieval is probably the most rapidly developing application of similarity searching. Evaluation of visual similarity is a natural process for people, which makes image search a perfect candidate for testing of content-based retrieval performance. Image searching is also interesting for a wide range of common users, which is reflected by the popularity of web image search systems (Google, Yahoo) and web galleries (e.g. Flickr, Facebook). However, these tools are mainly based on text search in image annotations, which has two substantial disadvantages. The first is that a considerable amount of human labor is required for manual annotation. Second, annotations are subjective and may not respond to other users' perception of the image.

Content-based approach seeks to overcome the disadvantages of text-based retrieval systems by exploiting the visual content of the image. It stands on a crossroads of many disciplines – computer vision, machine learning, information retrieval, human-computer interac-

tion, database systems, Web and data mining, information theory, statistics, and psychology. All of these participate in solving the problems of large collection management, suitable image representation, efficient indexing, intuitive searching and results presentation. Let us now briefly characterize the main issues of content-based image search and current research challenges, as defined in recent surveys [18, 45, 30, 20].

First of all, it is necessary to describe the data objects. It is important to notice that content-based retrieval does not rely on describing the content in its entirety. The description needs to be suited to the retrieval methods, which are based on similarity. The key problem is therefore to understand what people perceive as similar. In case of images, two main approaches exist. Either an image is described as a global object, or as a set of segments, eventually points of interest. Color, shape and texture properties are the most common features for extraction from the chosen area. The corresponding similarity measure then evaluates global similarity or similarity of certain image parts. The advantage of local descriptors is the ability to detect subimages, the disadvantage lies in increased complexity of extraction and higher computation costs during retrieval.

To enable efficient searching, extracted image descriptors need to be suitably organized and stored. Many existing technologies restrict the data domain to a vector space, using the space-partitioning or data-partitioning principles to organize the data. Typical examples of such structures are the *k-d tree* and *R-tree*, an overview of techniques that exploit the properties of vector space can be found in [9]. However, a more general approach to similarity searching is based on the metric space model. It is suitable for vector as well as non-vector data (e.g. DNA sequences) and does not suffer from problems common to vector data structures, such as crosstalk between vectors or problems with vector space dimensionality. The formal definition of metric searching as well as basic index structures are provided in [55].

The basic search model for the similarity retrieval is the query-by-example paradigm, where a set of objects with the smallest distance from a given query point is returned as the query result. The example may be either chosen from the dataset or provided externally (photo, drawing, etc.). When text annotations are available for images, these can be also used for query definition (which is then transformed to standard text search) or as an additional condition to the similarity query.

A number of image search systems have been created in the last decade (some of them can be found in [51]). However, experience has shown that the content-based search is not yet mature enough to be used by general public. Even though the search results are promising in many cases, in other situations they do not meet user expectations. "The need of the hour is to establish how this technology can reach out to the common man in the way text retrieval techniques have. [18]"

The main challenge recognized by many authors is the *semantic gap* problem. The existing descriptors and similarity measures are not sufficient for capturing the concept of similarity as perceived by users, who evaluate similarity with respect to the semantic meaning of the image. Moreover, the understanding of similarity is individual and context-dependent (e.g. in [45] authors distinguish search by association, aimed search and category search, each of which requires a different similarity measure). The two main research directions

aimed at solving the semantic gap problem are (1) learning the semantics of the image via machine learning and categorization, and (2) enabling users to tune the search settings to suit their needs using relevance feedback or other advanced search settings. Alternative solutions exploit the intelligence of crowds and game-like approaches for obtaining rich and precise image annotations.

Other research topics related to providing easy-to-use searching include the need for intuitive system interface, results postprocessing and presentation. A new research field is the security of content-based image search, which deals with image copy protection and forgery detection. As for the system architecture, parallel and distributed computing, hardware accelerations and approximative algorithms are being studied to enable interactive searching in very large data collections. To evaluate and compare existing approaches, standard benchmark sets and performance measures need to be established.

1.2 Dissertation Objectives

One of the main challenges of the image search identified in the previous section is the semantic gap problem. In our research, we intend to address several aspects of this problem for large-scale content-based image retrieval systems. The proposed methods will be verified over the existing similarity search system *MUFIN* [36, 37], developed at the Faculty of Informatics, Masaryk University. To ensure general usability of the proposed approaches, we base our research on the metric space model.

A promising approach to solving the semantic gap problem is based on involving the user into the search process. To achieve this, users need tools that allow them to specify their individual preferences. Many such tools have already been proposed for both text-based and content-based retrieval. However, in case of the content-based searching the research has been focused on small specialized collections rather that on general web searching. For the large-scale retrieval, efficient algorithms for the evaluation of advanced search operations yet need to be provided. Also, the existing methods have been proposed for various systems and without any systematic organization or regard to cooperation. Therefore, a uniform organization of the search methods is needed, which can be provided in the form of a query language. Specifically, the dissertation objectives are:

- *Efficient algorithms for query result refinement in large-scale similarity searching*. To enable scalability and fast retrieval, distributed architectures, parallel query evaluation and approximate searching are employed in the processing of very large datasets. We shall mainly focus on efficient evaluation of multi-point queries and feedback iterations in the distributed environment, considering both approximate and exact searching. We are also interested in result postprocessing methods as a way of fast interactive result refinement.

- *Query language for similarity searching.* We aim at defining a similarity query language that will support a wide range of advanced search options. First, we will study

and categorize the existing approaches to searching, which will be later formalized through a general query language. The motivation is to help users to customize the search settings according to their individual needs.

The thesis is organized as follows. First, we provide the basic principles of metric searching and system architectures in Chapter 2. Chapter 3 studies the existing methods of bridging the semantic gap using advanced search settings and iterative searching. In addition, we review the existing proposals of similarity query language. In Chapter 4, we select two promising strategies of bridging the semantic gap—the postprocessing methods and multi-point queries—and propose several algorithms for efficient evaluation of these operations in the distributed environment. We also outline our concept of the similarity query language.

Chapter 2

Metric Searching

In this chapter, we briefly introduce the similarity searching based on the metric space model. First, the mathematical concept of a metric space is defined and its fundamental properties are explained. We provide a few examples of similarity measures and present the basic query types that can be evaluated in the metric model. Next, we discuss several approaches to efficient organization of metric data and explain how parallelism can be utilized to create scalable systems. Finally, we present a particular similarity search system for content-based retrieval over large-scale metric datasets.

2.1 The Metric Space Model

The metric space is considered to be the most general data structure of similarity which can still be indexed and searched efficiently. It treats data as unstructured objects together with a function that measures distance (or dissimilarity) between pairs of objects. Formally, a metric space $\mathcal{M} = (\mathcal{D}, d)$ is defined [55] as a pair of domain of objects $\mathcal{D}$ and a total (distance) function d. The distance function d must satisfy the following properties: (i) non-negativity: $\forall x, y \in \mathcal{D}, d(x, y) \geq 0$; (ii) symmetry: $\forall x, y \in \mathcal{D}, d(x, y) = d(y, x)$; and (iii) triangle inequality: $\forall x, y, z \in \mathcal{D}, d(x, z) \leq d(x, y) + d(y, z)$. Such function is called a *metric*.

The distance functions express similarity of objects from a given domain. Different functions are used for various types of data and specific applications so that such characteristics of the data that are important for the application are reflected by the distance measure. Images are typically represented by a set of vectors that describe the low-level features such as color histogram, texture, etc. The most common similarity measures for vectors are the *Minkowski* (also denoted as L_p) distance functions, especially the *Euclidean distance*. Other well known functions are the *Edit distance* for strings or the *Jacard's coefficient* applicable to sets.

A query in the metric model is usually defined by a *query object q* and conditions that must be satisfied by qualifying objects, typically expressed as a constraint on their distance from the query object. The basic queries used in similarity searching are:

- *Range query:* This query is used to acquire objects within a certain distance from a given query object. Range query $R(q, r)$ is specified by a query object $q \in D$ and some query radius r that restricts the distance. The query retrieves all objects found with

distance to q at most equal to r. A specific type of the range query is the $R(q, 0)$ query called a point query or exact match.

- *Nearest neighbor query:* This query retrieves a fixed number of objects that are closest to the query object q. Such objects are called the nearest neighbors of q. Depending on the number k of neighbors to be retrieved, we define the $kNN(q)$ query. Compared to the range query, the kNN query does not require any knowledge of the distance function or the dataset.

In a simple case, queries deal with just one aspect of similarity, e.g. the distance of vectors which describe image color. However, this may not be satisfactory, since such query cannot distinguish a red circle from a red square. Therefore, more low-level similarity measures are often combined to evaluate the overall similarity of objects. The combination can be predefined and fixed, or set by a user in time of query specification. The substantial difference between these approaches is that the fixed combination can be precomputed and used for data indexing. On the other hand, user-defined combination can reflect the individual understanding of similarity. The evaluation of user-defined queries is more complicated and will be discussed in more detail in Chapter 3.

2.2 Indexing and Searching

To evaluate similarity queries efficiently, we need a quick access to the relevant data. This is provided by specialized index structures that organize objects from the given domain. The organization is based on the distances between objects under the given distance measure. Typically, the structure is designed as some kind of a tree, where objects are divided into subtrees with respect to their distance from the object in the parent node. All similarity searching structures use properties of distance function to navigate between substructures (e.g. the triangle property is often employed to decide whether a substructure can contain qualifying objects).

Small data collections can be organized in centralized structures on a single computer. The dataset is recursively divided using a set of designated objects (pivots) and a ball partitioning, hyperplane partitioning or excluded-middle partitioning principle (for more details see [55]). Some of the existing index structures are static and main memory structures, such as the *Vantage Point Tree* or *Generalized Hyperplane Tree*. The two most significant dynamic disk-oriented structures are the *M-Tree* and the *D-Index*. More information about these structures is provided in [55].

However, the centralized structures are not sufficient for real-time processing of large data collections. To enable more efficient data management, distributed architectures have been introduced. The use of more computing centers (nodes) provides easily enlargeable storage capacity and the possibility of exploiting parallelism during query processing. *GHT**, *Skip-Graphs* or *M-chord* are possible implementations of distributed index structure (their description and comparison is provided in [8]).

2.3 The *MUFIN* Search System

The *MUlti-Feature Indexing Network* (*MUFIN*) [36, 37] is a general tool for effective and efficient similarity searching in large and quickly growing data collections, developed at the Faculty of Informatics of Masaryk University. It is capable of indexing and searching any metric data and supports definition of several search indexes that describe different features of data objects. The system is built over distributed peer-to-peer (P2P) network so it can manage large volumes of data and provide acceptable response times. The *M-Chord* [38] algorithm is used to distribute the data among individual peers and to evaluate similarity queries. Each peer organizes the data locally in a metric-based index structure *M-Tree* [14].

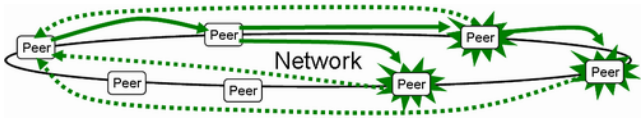


Figure 2.1: A typical query processing in P2P network

As the system is based on P2P architecture, a user can issue a query at any peer. Steps which are executed to answer the query are depicted in Figure 2.1. The query is forwarded to peers that may hold qualifying objects (solid arrows). Since the network changes in time as objects are added or deleted, navigation can be imprecise and query has to be forwarded several times until it reaches all relevant peers. Each peer with a promising data partition executes local search and returns all qualifying objects to the initiatory peer (dotted arrows) where the final answer is merged and presented to the user.

The *MUFIN Image Search* is the first prototype application of our system for image retrieval. Currently it indexes more than 100 million images from the Flickr[1] web gallery. Each image is described by five visual descriptors defined by the MPEG-7 standard [1]. The Image Search is publicly available through a web interface[2] shown in Figure 2.2.

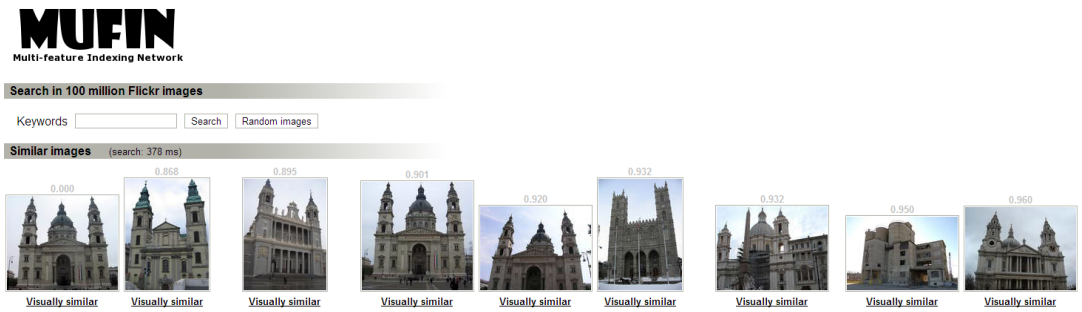


Figure 2.2: The *MUFIN Image Search* interface

1. http://www.flickr.com/
2. http://mufin.fi.muni.cz/imgsearch/

Chapter 3

Content-based Image Retrieval Strategies

The fundamental problem of content-based retrieval lies in capturing the concept of similarity. While users understand the meaning (semantics) of an image and evaluate similarity with respect to it, the search systems work with the low-level visual descriptors. The discrepancy between these two perspectives is referred to as the *semantic gap*. "The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation. [45]" In [26], the authors declare that there are several semantic levels between the raw image representation and human understanding of the image content, which include extraction of descriptors, identification of objects, object labeling, and full semantics with object relationships. The challenge is to overcome the misunderstandings that arise from the fact that current retrieval systems are very often on the lowest level, at best trying to identify the basic objects and their labels.

An excellent overview of existing approaches to the semantic gap problem is provided in [33]. The authors identify five categories: (1) using object ontology to define high-level concepts, (2) using machine learning tools to associate low-level features with query concepts, (3) introducing relevance feedback into retrieval loop for continuous learning of users' intention, (4) generating semantic template to support high-level image retrieval, (5) making use of both the visual content of images and the textual information obtained from the Web for WWW image retrieval. Most of these approaches try to learn the semantic meaning of images. However, this requires expert annotated training data, which can be accomplished for specialized data collections such as art images [24] or X-ray shots [31], but is difficult for very large and diverse datasets.

In case of wide, non-specific data collections for common use, another dimension of the semantic gap appears as a single object can often represent more semantic concepts; e.g. an image of a car in front of a building may represent both the car and the building. Understanding of similarity in such cases is individual and context-dependent, as different users at different times may have different interpretations or intended usages for the same image. Therefore, there cannot exist any universal best measure of similarity. "The challenge for image search engines on a broad domain is to tailor the engine to the narrow domain the user has in mind via specification, examples, and interaction. [45]" In other words, the only way to provide satisfactory search results in such conditions is to involve the user in the search process. The basic strategy is to employ the relevance feedback mechanism, which tries to learn the users' intentions on the fly.

In our research, we focus on search in general large collections of images, such as data from web galleries. The annotations of such images are typically sparse and inaccurate, which causes low precision of the text-based retrieval. From the above mentioned content-based search strategies, semantics-learning and ontological approaches are not suitable as they require training data. For the large and diverse collections, it is hardly possible to provide a training data set as the range of potential semantic concepts is too large. This implies that the searching has to rely on the visual data properties. Therefore, we restrict our study to approaches based on the query-by-example paradigm and explore how the basic model can be extended to enable users to specify their preferences more precisely and thus improve the quality of search results. We are mainly interested in *top-k queries*, which return the k most relevant objects. This kind of query is the most typical for general searching as it can be used without any knowledge of the collection properties (in contrast to other query types, e.g. the *range query*, where some knowledge of dataset distribution is needed to define reasonable search parameters).

In the following sections, we provide an overview of existing strategies of content-based query definition, presentation of results and iterative response refinement as the three aspects of retrieval which the user can influence. We base our analysis on the metric space model, which is the most general structure for similarity searching that can be efficiently indexed. For each part of the retrieval process, we analyze the implementation topics concerning efficient evaluation over very large data sets.

The definition and evaluation of a search task consists of many steps. As stated in [33], a query language for similarity searching would provide a highly useful tool to organize the search options. Therefore, we devote the last section of this chapter to the study of research in this area.

3.1 Search Task Definition

The search task is a particular search problem that is submitted to a retrieval system for evaluation. It can either by defined by the user, or result from previous actions (e.g. in case of iterative searching). The search task consists of the query definition, output specification and specification of target data collection or its subspace. In this section, we deal mainly with the query definition, which is the key part of the search process, and we briefly mention the selection of target data. The output specification will be discussed in the next section.

In the metric model, the query is defined by an example object and a distance (dissimilarity) function that is used to evaluate the similarity of objects. The basic questions to be solved are therefore the example object selection and the definition of required similarity measure. The first part of this section focuses on the specification of a query object. We compare single-point and multi-point queries and mention the keyword search as a possibility for selecting initial images. In the second part, we analyze several types of distance functions that are used for searching. These can be classified as fixed, user-specified, or dynamic, each of which has its special characteristics.

3.1.1 Query Object Selection

To provide the query object, users can select either some image from the searched collection or external object such as photo, sketch or computer graphics. If the user doesn't have the desired image at hand, it can be found by random browsing or using existing tools for keyword search in image collections. In case of external object, the search system has to extract the visual features from the object before starting the search.

When the keyword search is used in the beginning, the system usually forgets about the textual information in the next query evaluation and executes standard content-based retrieval over the whole dataset. However, for extremely large collections, another approach has been developed, that only searches for similar images within the subset of images found by the keyword search. This approach has been adapted by the *Google VisualRank* [28] technology (used in *Google Images*[1]) or *Microsoft Bing*[2] search system, where the content-based evaluation is in fact only used to rank results of the text search. In collection of billions of images, this method returns good results, even though it skips objects with incorrect or missing annotations.

Real content-based approach uses only the selected query image for searching. However, it may be difficult for users to express their expectations by a single example as most images contain more than one object and often express more than one concept. For example, a single photograph of a red rose may be used to specify a search for red roses, red flowers, or roses of any color. Therefore, a multi-object query has been proposed to enable more precise specification of the search task. User may select several query images, which can be processed in two ways: either the examples are pre-processed and combined to form a single query, or each query example is independently evaluated and the results are then combined. The combining of more images into an ideal query object was proposed in [27] for vector spaces, exploiting weighted averaging of visual features. The second approach, also called *Multiple expansion search* or *Aggregate query*, is generally agreed to be more effective. In [48], authors show that average performance of querying with multiple examples is significantly better than single-example querying. They also investigate the effect of the number of examples and performance of several primitive combination functions (min, max, sum). More advanced methods of results combination are proposed in [52] for vector spaces and in [41] for metric space model.

A possible extension of multi-point query is a query with both positive and negative examples. Negative examples are used to specify image aspects which are not desired in the result. As will be discussed in the section devoted to relevance feedback, the negative examples are difficult to work with, as they typically represent diverse concepts, while the positive images are related to one common concept. Still, it is possible to use negative examples in the pursuit of optimal query center, or to evaluate subqueries for both positive and negative examples and penalize objects similar to negative examples when combining the subresults [23].

1. http://images.google.com/
2. http://www.bing.com/images

Efficient implementation of single-example queries for large datasets has been the center of attention of many research groups. Building on basic index structures for metric space such as the M-tree [14], the most advanced existing techniques are based on distributed algorithms and approximate query evaluation [36]. As for the multi-object queries, earlier studies proposed to evaluate a standard query for each example and then combine the results. The first optimization for aggregate queries was introduced in [40], where a single query evaluation is used, based on novel pruning rules defined for aggregate queries over centralized metric index structures.

To conclude this section, the results of keyword search can be used to reduce the search space of content-based retrieval for extremely large collections. To enable more precise definition of a search task, it is advantageous to employ the multi-object queries. The best results are achieved by exploiting independent evaluation of several queries and combining their results using sophisticated combining function. As for the search costs, the most advanced techniques for single-object queries exploit parallel processing over distributed architectures and enable approximate evaluation to speed up the retrieval. The multi-object queries have only been studied on centralized architectures with the focus on sophisticated pruning of the search space.

3.1.2 Similarity Function Definition

The similarity function is used to measure the visual likeness of objects. As explained in Chapter 2, the *similarity* of objects can also be understood as their *distance*, the low distance representing high similarity and vice versa. In the following, we will use both notions interchangeably.

The distance function used for searching should reflect user's individual understanding of similarity in a given situation. Therefore, it is inherently user- and context-dependent. We shall denote this function as *query distance*. However, to enable efficient retrieval of relevant data objects, it is necessary to build index structures over the data collection in advance. These are created using a different similarity measure, *index distance*. To enable efficient searching, there has to be a strong relationship between the *index distance* and the *query distance* used for searching, so that the precomputed *index distances* can be utilized.

In the simplest case, the *index distance* and *search distance* are the same, which means that the user is given no choice and has to use the distance selected by the search engine creators as the optimal one. Such optimal distance is usually determined by supervised machine learning, as in [3].

However, as discussed in the chapter introduction, users may need to specify a different similarity measure that would reflect their need. Therefore, it is useful to provide them with a possibility to tune the distance function. Often, the similarity is described as a weighted sum of partial distances corresponding to the low-level features (e.g. color distance, shape distance, etc.). Then a straightforward idea is to allow users to set the weights, which can be done either explicitly, or the weights can be derived from the properties of query objects in case of multi-point queries, as presented in [27].

To enable efficient evaluation of such query, several approaches have been proposed. Fagin [21] shows how to compute the combined distance when sorted lists of objects under the partial distances are available. Using this algorithm and a set of indexes for each of the partial distances, any user-defined weighted sum distance can be processed, which is used e.g. in [7]. A different approach is employed in [11], where a specialized index structure called *Multi-Metric M-tree* is used to store the partial distances and estimate the weighted distances.

Even more general index structure called *QIC-M-tree* has been proposed by Ciaccia and Patella in [12]. Here the authors observe that when a lower-bounding function can be found for a class of distance functions, then it can be used to build an index that efficiently supports queries under all the distances from the given class. Therefore, users are not limited only to the weighted sums but can choose from a whole class of functions, e.g. the *Minkowski distances* for vectors. A more theoretical description of this approach as well as precise definition of pruning rules is provided in [13].

Apart from the efforts to enable users to tune the similarity, some systems also try to guess the suitable distance function for the given query object based on its similarity to some pre-selected representants of object classes with associated optimal distance (in such approaches, again mainly weights of sub-distances are adjusted).

Finally, the authors of [49] propose to use a dynamic function, where a fixed-size subset of visual features is selected for each distance evaluation such that the resulting distance is minimal. This method is suited for data with many extracted features and relatively high noise.

In this section, we have identified the following types of distance function: (1) static, determined by machine-learning as optimal for the given data set, (2) user-defined, entered either explicitly or using multi-point query and evaluated using specific algorithms and index structures, (3) defined by the system using some knowledge of object classes, (4) dynamic, with a different subset of features used for each distance evaluation. All these functions are applicable to the metric model.

3.2 Presentation of Results

When a query is issued to the search system, it gets evaluated and a set of objects most similar to the query image (images) is retrieved. The next task is to present the search results to the user. The experience of text retrieval systems shows that users expect the most relevant results to appear on the first page, containing about 20 top-ranking results. Results on next pages are much less likely to be visited. Therefore, the challenge is to get the most relevant objects (from the user's point of view) on the first positions. In a user-friendly search system, various settings of result presentation should be available to allow users to tune the search to their need.

The advantage of image retrieval, as compared to text search, is that more objects can be displayed on a single page since it is easier for people to capture the visual information. Still, a sophisticated organization of the results can help users to navigate in the images and find

the desired objects. In the first part of this section, we review the existing result visualization methods.

However, there are typically a lot more objects returned by the search system, than can be displayed on the first page. Therefore, result postprocessing techniques are used to select the most relevant images. We describe two classes of postprocessing methods, namely ranking and clustering.

3.2.1 Visualization

The visualization of results is the last step of image retrieval. After the query evaluation and postprocessing, the final set of objects needs to be shown to the user. Images are represented by thumbnails, which can be completed by keywords or semantic categories, when available. Here we explore the strategies of visual organization of results.

The basic approach adapted by most existing image search systems (both content-based and textual) is to show the results in a rectangular grid, with the relevance descending from left to right and from top to bottom. This provides users with clear information about the ranking of results, which may be useful for general browsing in the collection. However, it takes longer to find the desired object in case of targeted search.

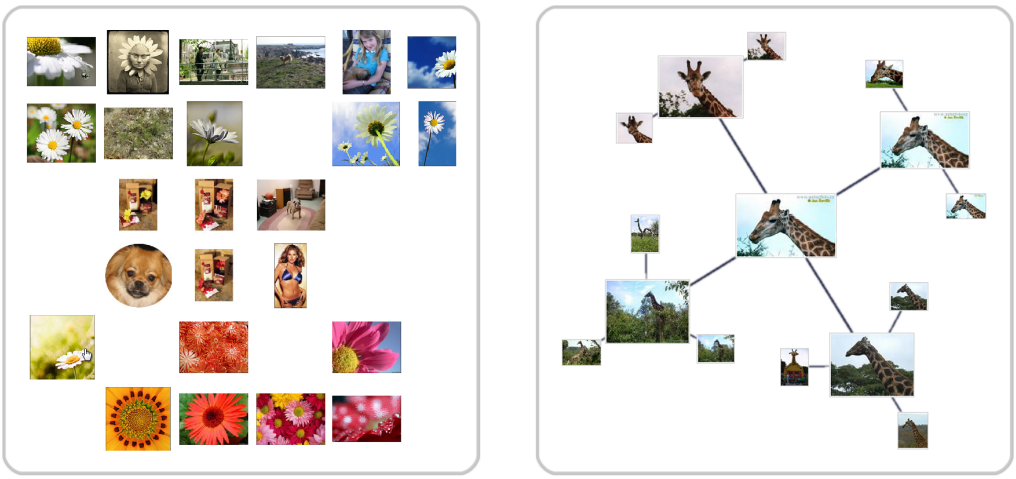
The authors of many studies (e.g. [19, 4]) argue that users would appreciate a different organization of results, reflecting the mutual relationships between the images. All the proposed methods are based on forming clusters of images based on their visual, textual or other properties (e.g. originating web site). Two examples of such organizations are shown in Figure 3.1. The screenshot from the *pixolu*[3] image search presents grid of images ordered by colors, the *Google Image Swirl*[4] technology uses graph representation, where the edges represent relationship between objects. When user chooses a different image, the graph structure changes to provide a closer view on objects related to the chosen one. In this way, dynamic visualization of images is provided. However, authors of [15] argue that static representations of results are more preferred by users, the dynamic ones being more attractive at first sight but less comfortable to use.

Altogether, presentation techniques can be classified as static or dynamic, the second class being less frequent and probably less suitable for common searching. Ranking-based grid organizations are sufficient for general similarity browsing, clustering of images may help targeted search.

3.2.2 Ranking

Ranking is the process of ordering objects in a search result so that the most relevant ones are displayed first (on the first, most important page). In a basic search model, ranking is defined by the distance function, the objects with the lowest distance being the highest-ranked ones. However, we have already discussed that such results are often not satisfactory. Thus, it may

3. http://pixolu.does-it.net/
4. http://image-swirl.googlelabs.com/



(a) (b)

Figure 3.1: Query result visualization: a) *pixolu* search, b) *Google Image Swirl*

be advantageous to employ some postprocessing steps to refine the result, especially as this is not expensive since the operations are only evaluated over a small subset of the dataset.

Ranking methods have mainly been proposed for text-based image searching, where the visual similarity is used only in the ranking phase. However, this technique can easily be used for content-based searching as well. The ranking of objects in the result set can be recomputed with respect to local, global or combined similarity, as proposed in [53]. The original rank from the initial query evaluation may also be taken into consideration.

The two main approaches are graph-based ranking and clustering-based ranking. In the first case, the result is represented by a graph, where objects form the vertices and weighted edges express the level of similarity between objects. The graph is then processed in different ways: in [57], the densest component is found that corresponds to the largest set of most similar subset of images, which receive the highest rank. In [29], the authors propose to employ the *PageRank* principle used in text retrieval, creating a *VisualRank*. Each vertex is weighted by some criterion, e.g. its cardinality, and using a random-walk approach, the weights are propagated through the graph. The final weights determine the ranking.

In the clustering-based approaches, objects in the result are first divided into clusters. Then, several methods can be used according to [39] to rank the objects: (1) majority-first method (largest cluster receives highest rank, objects within cluster are ordered with respect to their distance from the cluster centroid), (2) centroid-of-all method (overall centroid is found, objects are ordered with respect to distance from this centroid), (3) centroid-of-top-k method (the centroid is found between the k highest ranking objects based on the distance function), (4) centroid-of-largest-cluster method (centroid of largest cluster becomes the overall centroid).

To conclude, ranking enables to refine the query result using additional similarity evaluation over the objects found in the basic retrieval. When the text search is used to provide the first result set, any similarity measure can be used for ranking, combination of both global and local similarities being the most precise. In case of real content-based retrieval, it is advisable to use a similarity measure with lower computation costs for the initial retrieval and then to refine the result by the help of a more precise distance function.

3.2.3 Clustering

Clustering has already been mentioned in the previous parts as a method of results organization during visualization or an approach to result ranking. However, clustering can also be used to prune the result set, which is the topic of this section. There are two reasons for the use of this technique. First, for a given query image, there may be too many nearly-identical objects which fill the result without providing any new information. Second, the query image may represent several diverse concepts, which may get mixed in the standard ranking. In such case, objects in result set are classified into clusters and only cluster representants are provided to the user.

Several approaches have been proposed to determine the clusters and select the representants. In [44], the authors define a new type of similarity query, which they call the k *distinct nearest neighbors query*. Such query only returns distinct objects, which are objects with mutual distance greater than some predefined constant, denoted as *separation distance*. The choice of the *separation distance* is left to users, which assumes that they have some knowledge of the data distribution. Two possible implementations are proposed, retrieving either the first visited member of each cluster, or the centroid of each cluster. The same distance function is used both for initial retrieval and for the clustering.

Other solutions use different similarity measures for querying and clustering. The authors of [50] propose to use dynamic clustering, where the distance function for clustering is chosen with respect to the importance of individual features for the given query. Three methods of cluster representant selection are analyzed, the best being the *reciprocal election*, where each object in the cluster votes for a different object that represents it best and the object with most votes is chosen as the representant.

Altogether, the clustering principles can be used to provide more distinct results with representants of the diverse concepts, which are automatically identified in the initial result set. Users can easily explore the results and choose which concept is most relevant to them. Then the remaining images from the respective cluster can be provided.

3.3 Relevance Feedback

There are many situations when a user is not satisfied with the results of searching. Sometimes, the user has not been able to provide a good enough specification, in other cases users just browse through the collection without a clear idea of expected outcome. However, if there are at least some relevant images in the response set, these can be used to refine the

search. This idea is exploited in the relevance feedback technique used in traditional text-based information retrieval, which has been introduced to content-based retrieval in [42]. The relevance feedback attempts to capture the user's precise needs through iterative feedback and query refinement. The basic principle can be described in three steps [56]:

1. Machine provides an initial retrieval results, through query-by-keyword, sketch, or example, etc.
2. User provides a judgment on the currently displayed images as to whether, and to what degree, they are relevant or irrelevant to her/his request.
3. Machine learns and tries again. Go to step 2.

Relevance feedback algorithms have been shown to provide significant performance improvement in retrieval systems. This technique is also convenient for users since the judging of image content is easy and fast. However, there is still a lot of space for improvement. In the following parts, we analyze two important aspects of relevance feedback – the approaches to learning users' needs and the techniques of efficient evaluation of the feedback loops over large datasets.

3.3.1 Learning Strategies

The objective of the relevance feedback is to learn what a user asks for and provide it. To achieve this, it is necessary to obtain as much information as possible from the user and derive from it some better knowledge of users understanding of similarity and search target. In the following, we provide an overview of existing learning strategies based on [56, 42, 16, 17, 47].

The first topic that needs to be considered is what information can be provided by the user. The number of iterations and evaluated objects is usually very small so it is important to decide which objects should be presented to the user in each iteration. The user and the system often have conflicting goals. The user wants to see as many relevant images as possible in each iteration (in case of *category search*), or the best possible approximation of the searched object (in case of *target search* where the user is looking for one specific image). The system, on the other hand, needs to maximize the information learned in each iteration, which is achieved by presenting random (or uncertain) images for evaluation. A possible compromise is to combine these two strategies, providing n most relevant images and m random ones, where the number n is increasing with later iterations. Several strategies of object selection for fast convergence of relevance feedback are discussed in [32].

Concerning the user evaluation of presented objects, some systems only require marking positive examples while other work with both positive and negative ones. In the second case, the negative samples need to be treated in a different way, as they probably belong to different classes but at the same time are unlikely to be representative for all the irrelevant classes. A study of negative samples effect is provided e.g. in [23]. Some approaches even

consider levels of (ir)relevance for each of the evaluated objects, which are used as object weights in the learning process.

Early approaches to relevance feedback assume the existence of an ideal query point that, if found, would provide the appropriate answer to the user. These approaches belong to the family of *query point movement* methods, which in every iteration try to find a better query point. The selection of the optimal query point is usually based on the *Rocchio formula* [41]. Later works argue that the desired set of objects may consist of several classes with slightly different low-level representations and propose the use of multi-point queries. These methods have already been discussed in the search task definition section. This approach is often combined with *similarity metric modification* methods that seek to guess the optimal similarity measure. This is accomplished by re-weighting of the individual features in the distance function so that features more relevant (i.e. the ones with low variance) receive more weight.

Recent work on relevance feedback often relies on support vector machines [34], which are trained to distinguish between relevant and irrelevant objects. Other approaches are based on the probabilistic model and use the evaluated examples to estimate the density of feature spaces. For each object in the dataset, the probability that it will be judged as relevant is computed [47].

Some authors also consider long-time learning, accumulating the knowledge gained from relevance feedback. For example, each object can learn its optimal query point and distance function, which define the center and shape of the cluster of relevant objects. Other approaches try to classify objects according to the feedback. However, these strategies can only be used for some applications such as medical image databases with semantically meaningful static clusters, as discussed in the chapter introduction.

To summarize this section, we have presented several learning strategies for the relevance feedback, including query point movement and distance function modification, support vector machines and probabilistic models. Even though other approaches exist, these are most used in current research. All these strategies can be used to work with positive examples only or with both positive and negative examples, where the second possibility can provide better results but is also more prone to biasing the search. To enable quicker learning, random or most-information-providing images can be mixed into the result.

3.3.2 Implementation Aspects

The relevance feedback is based on interaction with the user. However, the user can only be expected to cooperate if it does not require much time and if rapid improvements can be seen. Therefore, it is necessary to provide interactive communication, which requires very fast evaluation of feedback loops. Also the number of feedback cycles should be kept as low as possible. The efficient evaluation of feedback process is in particular important for the large-scale searching.

The state-of-the-art techniques as presented in previous section are mainly interested in the accuracy of retrieval and do not study the efficiency of evaluation. The experiments are

typically evaluated only for relatively small datasets and scalability of the approaches is not considered. The query point movement and similarity metric modification methods are better suited for large datasets as they can be evaluated using metric indexes (more details are provided in Section 3.1). On the other hand, sophisticated learning strategies based on support vector machines or probabilistic modeling typically recompute the ranking of the whole dataset in each iteration. Such algorithms cannot be used for scalable searching.

Only a few research works are focused on the efficiency of the retrieval process. One of these is [54], where the authors provide a dimensionality-reduction technique to compress the data collection (modeled in Euclidean space) into a small in-memory database that can be used during the evaluation of feedback loops. The reduction exploits statistical properties of the dataset, in particular the mean and standard deviation of each dimension of the vector space. The resulting in-memory database allows approximate evaluation of queries and is used in combination with queries over the whole dataset to speed up the feedback evaluation.

Another approach is proposed in [43] to accelerate the feedback evaluation based on query point movement technique. It is based on gathering information during iterations and exploring their correlations. The key idea is that the range of objects relevant to the refined query in i-th iteration is likely to overlap with the range of objects relevant to the previous query. Therefore, some of the information from the previous evaluation can be stored and re-used. Several techniques are suggested for estimation of the candidate images that are likely to appear in next iterations. Moreover, the authors argue that it is possible to predict the target query point after a few iterations from the trend of changes and thus skip several iterations.

To conclude the survey of feedback techniques, only query point movement and distance modification strategies can be used for large-scale retrieval since their evaluation can profit from the underlying index structures. Other methods that require scanning of all objects can only be used as a postprocessing option over a small result set. The topic of efficient evaluation has not yet been studied much. Existing approaches propose to reduce the dataset to minimize disk accesses and to study relationships between iterations and reuse the results of previous feedback loops.

3.4 Query Language

While most of the research in the area of content-based retrieval is focused on index structures and search algorithms, it is also desirable to provide tools that allow users to formulate the queries. In the previous sections, we have demonstrated the importance of user participation in the query definition and the number of search options that help to express users' expectations (either explicitly, or through the relevance feedback). A structured query language allows to define the queries in a uniform way. From the system point of view, a standardized query format enables easy communication between different search engines.

The two main types of query language that have been proposed for similarity searching are SQL-based languages and XML-based languages. The first approach builds on the solid

Figure 3.2: Similarity query syntax: a) *MPEG Query Format* [46], b) *PostgreSQL-IE* [25], c) SQL extension proposed in [5]

theoretical model of relation algebra and well-established SQL query language. The second solution is motivated by the character of complex data, which is often described in the XML format, e.g. in the MPEG standard.

The first extension of relational algebra for similarity searching has been proposed in [2], defining a very general model of complex objects, their properties and similarity functions. Later works focus on the most commonly used search operations, such as the k-nearest neighbor query, the range query and similarity joins, and add new selection and join operators to the similarity algebra. The most recent theoretical works [22] study the algebraic properties of similarity operators and propose query optimization strategies for the evaluation of complex search conditions (e.g. nested range queries).

Naturally, the new functionality for similarity searching has also been added to the SQL language. Unfortunately, such extensions have been proposed by more research groups, resulting in different syntax and different search options that are available in various systems. Typically, the SQL extensions support the range and nearest neighbor operations and enable specification of basic search parameters (number of neighbors, range). In some implementations, users may choose from more similarity measures and index structures. The result postprocessing and presentation are not supported.

A different approach to similarity querying is based on the XML format frequently used for data representation. To avoid the collisions of different query language definitions, the MPEG standard group initiated a call for proposal for *MPEG-7 Query Format* standard. The standard shall define both the *Query Input Format* and *Query Output Format* to provide a general interface for multimedia searching. The current status of the standardization efforts is described in [46]. It defines the query and reply formats exchanged between clients and servers in a distributed multimedia search-and-retrieval system. The proposal enables detailed description of output and input data and query conditions, but does not support

user-specific distance functions or query result postprocessing. An extension of the *Query Format* is proposed in [35], enabling users to define weighted multi-object queries.

A brief example of a possible SQL-based and XML-based syntax is provided in Figure 3.2 for a simple search task *Find the 10 images most similar to the specified example image.* In fact, the first query returns some default number of nearest images, as the employed language does not provide means of specifying the result set size. On the whole, the existing query languages for content-based retrieval support the basic search operations, but use different syntax (even among the same class of languages) and provide different search options. None of the existing languages can be used to specify all the settings mentioned in previous sections (multi-object queries, user-defined similarity measure, postprocessing options). The current research in this area is focused on the standardization of query definition and the formulation of optimization strategies for the query evaluation.

Chapter 4

Thesis Plan

In this chapter, we would like to summarize the state-of-the-art of the content-based retrieval and formulate our research objectives. As we have anticipated in the beginning, we are mainly interested in large-scale searching with real-time interaction with users, similar to using a web search engine. Most of the approaches that have been analyzed in the previous chapter were proposed for and tested over small collections and did not consider the distributed environment necessary for searching in large datasets. It is therefore needed to provide efficient algorithms for such architectures. Also, the existing systems lack tools that would allow formulation of more complex queries. We have studied the existing proposals of query languages for similarity searching and discovered that none of them supports all the advanced search settings that are necessary to provide satisfactory searching.

Therefore, our global objective is to enable more precise and convenient searching in large data collections. We aim at bringing the proposed advanced functionality to users and providing efficient evaluation. In particular, we focus on (1) query results postprocessing, (2) efficient implementation of multi-point queries, and (3) extending the similarity query languages to support the advanced operations. We can capitalize on the existing *MUFIN* implementation and the *MUFIN Image Search* application, which provides both data and interface for user-satisfaction experiments.

In the following sections, we describe these three objectives in more detail. We conclude the chapter by the time schedule of our future research.

4.1 Postprocessing

In the previous chapter, we have discussed postprocessing only as a way of result refinement that is executed before presenting the results to the user. However, we can also consider working with the result set afterwards, using interaction with user to choose the postprocessing options. Thus, we can employ postprocessing similar to relevance feedback but only working with the initial result set instead of issuing a new query. This is particularly useful in case of the large datasets, where the full query evaluation is expensive.

At the same time, the distributed architectures allow to acquire larger result set than required by the user with little added costs (due to the overhead of network communication, etc.). Therefore, we propose to retrieve large result sets with each query evaluation (e.g. by one or two orders of magnitude, depending on the efficiency of the query processing). Only the top k required objects will be presented to the user, but he/she will able to re-rank the

whole result using the postprocessing options. There are many possible modifications that can be applied to the result, e.g. changing the similarity function, choosing different features for similarity evaluation, using modified query objects or involving other metadata such as annotations.

4.2 Multi-object Queries

The relevance feedback allows users to clarify their information needs by selecting several positive and negative examples from the result of the previous search. Alternatively, even the initial search definition can employ more query objects to specify the search task more precisely. This implies that the multi-point queries will become frequent in content-based retrieval. Recent research has shown that more precise results are obtained when the multi-object query is evaluated using more query objects, rather than by combining the examples into one ideal query object.

However, such processing is also more expensive. The naive evaluation, which executes a separate query for each example and then combines the results, is extremely expensive. An optimization has been introduced in [40] for centralized systems which enables to evaluate the multi-object query in one pass through the index structure. Our first objective is to generalize this algorithm for distributed architectures and offer further optimization for the relevance feedback evaluation. In particular, we plan to study the three strategies of multi-object nearest neighbor query processing: Full evaluation, Evaluation with prior knowledge and Approximate evaluation. The first one is suitable for new queries, the second and third enable precise and approximate evaluation of the feedback loops, respectively.

Full evaluation

This approach must be used when the multi-point query is issued as the initial query. The query objects are used to identify the peers that may hold relevant data and decide in what order they will be visited. The order is important, since the k-nearest-neighbor query is evaluated as a range query with dynamic range, which is initially set to infinity. The radius is reduced as better objects are found, being equal to the distance of the k-th best object seen so far. Therefore, it is important to try the most promising peers first. In the distributed environment, several peers can be searched in parallel. Each peer is examined using the aforementioned algorithm for centralized searching.

Evaluation with prior knowledge

When the multi-point query is issued during the relevance feedback loops, we already have some knowledge of the data. In particular, we can exploit the result set from the previous search which, with a high probability, contains relevant objects. Therefore, the new query will be first evaluated over this result set, which will provide good approximation of the query radius. This radius will then be used to search the database, enabling to apply more strict pruning and quicker evaluation of the query.

Approximate evaluation

The principle of the nearest-neighbor query is to order the database according to objects' growing distance from the query points and retrieve the k objects with the highest rank. In case of feedback iteration when the new query objects are very similar to the previous ones, it is probable that the previous and new ranking of the dataset will not differ much. Let us suppose that the k highest-ranked objects in the new ordering also have quite high rank in the old ordering, the worst of them having rank l. Then to evaluate the new query, it is sufficient to compute the new distances for the top l objects from the old ordering. We exploit this idea in our proposal of approximation strategy.

As already mentioned in the previous section, it is advantageous to acquire more objects than necessary in each query evaluation. Let us consider a number m of objects that are retrieved. Then, in the feedback loop, we only recompute the ranking of these objects (i.e. postprocess the result). In case the number m was higher than the (unknown) number l, we obtain precise result of the new query, otherwise we obtain approximate result. This strategy can be used when the expected number l is not much higher than m.

The numbers l, m need to be determined in experiments depending on the required result size k and the distance of old and new query objects. Whether to evaluate a new search or to use the approximate (postprocessing) approach can be decided with respect to this distance.

The latter two above mentioned strategies can be also used for the evaluation of single-point queries when some previous result is available. This may occur during the relevance feedback or in the course of simple browsing through the image collection.

4.3 Query Language for Metric Searching

Even though several proposals of similarity query languages exist, none of them is satisfactory for our purpose. The languages usually support some of the advanced search settings, but not all of them. Especially, the result postprocessing options are not included in those proposals. In addition, the existing proposals do not consider the possibility of approximate searching.

To choose one of the existing languages to build on, first we have to decide whether to use XML syntax or SQL syntax. Since our motive is to provide users with a tool for query definition, we need an easy-to-use language with clear syntax. This is better satisfied by SQL, the XML format being too complicated (compare the simple query definition in Figure 3.2). On the other hand, there are efforts to create a standardized similarity language in XML, which should not be ignored.

Since the user view is more important to us and we have a long-term experience with various databases, we prefer the SQL-syntax solution. We might even be able to create a plugin for a classical relational database like [25]. In the future, when the *MPEG-7 Query Format* becomes a real standard, we can provide a conversion mechanism between our syn-

tax and the *MPEG-7 Query Format*. We expect that the XML query description will be used for communication between various search systems rather than for direct formulation of a query.

In our query language, we intend to propose new operations to support the advanced query definition (multi-object queries, user-defined distance, time or precision preferences) and postprocessing options (additional ranking, clustering). We also plan to implement a parser and integrate the query language into the user interface of the *MUFIN* search system.

4.4 Study Plan

In this section, we provide a rough schedule of the future research:

Spring 2010:	query result postprocessing – design, implementation, evaluation query language – formal definition
Autumn 2010:	query language – parser, execution strategy multi-object queries – algorithm design
Spring 2011:	multi-object queries – prototype implementation, experimental evaluation
Autumn 2011:	multi-object queries – full implementation of the relevance feedback mechanism
Spring 2012:	extensive testing, evaluation completing the dissertation thesis

Bibliography

[1] MPEG-7. Multimedia content description interfaces. Part 3: Visual., 2002. ISO/IEC 15938-3:2002.

[2] Sibel Adali, Piero Bonatti, Maria Luisa Sapino, and V. S. Subrahmanian. A multi-similarity algebra. *SIGMOD Rec.*, 27(2):402–413, 1998.

[3] Giuseppe Amato, Fabrizio Falchi, Claudio Gennaro, Fausto Rabitti, and Pasquale Savino. Improving image similarity search effectiveness in a multimedia content management system. In *MIS 2004: Proceedings of International Workshop on Multimedia Information Systems*, pages 139–146, 2004.

[4] Paul André, Edward Cutrell, Desney S. Tan, and Greg Smith. Designing novel image search interfaces by understanding unique characteristics and usage. In *INTERACT '09: Proceedings of the 12th IFIP TC 13 International Conference on Human-Computer Interaction*, pages 340–353, Berlin, Heidelberg, 2009. Springer-Verlag.

[5] Maria Camila Nardini Barioni, Humberto Luiz Razente, Caetano Traina Jr., and Agma J. M. Traina. Querying complex objects by similarity in SQL. In *Proceedings of 20 Simpósio Brasileiro de Bancos de Dados*, pages 130–144. UFU, 2005.

[6] Michal Batko, Petra Kohoutkova, and David Novak. CoPhIR image collection under the microscope. In *SISAP '09: Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, pages 47–54, Washington, DC, USA, 2009. IEEE Computer Society.

[7] Michal Batko, Petra Kohoutkova, and Pavel Zezula. Combining metric features in large collections. In *Proceedings of the 24th International Conference on Data Engineering Workshops, ICDE 2008, April 7-12, 2008, Cancún, México*, pages 370–377. IEEE Computer Society, 2008.

[8] Michal Batko, David Novak, Fabrizio Falchi, and Pavel Zezula. On scalability of the similarity search in the world of peers. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, pages 1–12, New York, NY, USA, 2006. ACM.

[9] Christian Böhm, Stefan Berchtold, and Daniel A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33(3):322–373, 2001.

[10] Paolo Bolettieri, Andrea Esuli, Fabrizio Falchi, Claudio Lucchese, Raffaele Perego, Tommaso Piccioli, and Fausto Rabitti. CoPhIR: a test collection for content-based image retrieval. *CoRR*, abs/0905.4627v2, 2009.

[11] Benjamin Bustos and Tomáš Skopal. Dynamic similarity search in multi-metric spaces. In *MIR '06: Proceedings of the 8th ACM international workshop on Multimedia infor-*

mation retrieval, pages 137–146, New York, NY, USA, 2006. ACM.

[12] Paolo Ciaccia and Marco Patella. Searching in metric spaces with user-defined and approximate distances. *ACM Trans. Database Syst.*, 27(4):398–437, 2002.

[13] Paolo Ciaccia and Marco Patella. Principles of information filtering in metric spaces. In *SISAP '09: Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, pages 99–106, Washington, DC, USA, 2009. IEEE Computer Society.

[14] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In *VLDB '97: Proceedings of the 23rd International Conference on Very Large Data Bases*, pages 426–435, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[15] Katy Cooper, Oscar de Bruijn, Robert Spence, and Mark Witkowski. A comparison of static and moving presentation modes for image collections. In *AVI '06: Proceedings of the working conference on Advanced visual interfaces*, pages 381–388, New York, NY, USA, 2006. ACM.

[16] Michel Crucianu, Marin Ferecatu, and Nozha Boujemaa. Relevance feedback for image retrieval: a short survey. In *State of the Art in Audiovisual content-based retrieval (DELOS2 Report FP6 NoE)*, 2004.

[17] Gita Das and Sid Ray. A comparison of relevance feedback strategies in CBIR. In *ICIS 2007: 6th IEEE/ACIS International Conference on Computer and Information Science*, pages 100–105, 2007.

[18] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 40(2):1–60, 2008.

[19] Haoyang Ding, Jing Liu, and Hanqing Lu. Hierarchical clustering-based navigation of image search results. In *MM '08: Proceeding of the 16th ACM international conference on Multimedia*, pages 741–744, New York, NY, USA, 2008. ACM.

[20] John P. Eakins and Margaret E. Graham. Content-based image retrieval. Technical report, Institute for Image Data Research, University of Northumbria, Newcastle, 1999.

[21] Ronald Fagin. Combining fuzzy information: an overview. *SIGMOD Rec.*, 31(2):109–118, 2002.

[22] Mônica Ribeiro Porto Ferreira, Agma J. M. Traina, Ires Dias, Richard Chbeir, and Caetano Traina Jr. Identifying algebraic properties to support optimization of unary similarity queries. In *Proceedings of the 3rd Alberto Mendelzon International Workshop on Foundations of Data Management*, 2009.

[23] Annalisa Franco, Alessandra Lumini, and Dario Maio. A new approach for relevance feedback through positive and negative samples. In *ICPR '04: Proceedings of the*

17th International Conference on Pattern Recognition, pages 905–908, Washington, DC, USA, 2004. IEEE Computer Society.

[24] Haiying Guan, Sameer Antani, L. Rodney Long, and George R. Thoma. Bridging the semantic gap using ranking SVM for image retrieval. In *ISBI'09: IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pages 354–357, 2009.

[25] Denise Guliato, Ernani V. de Melo, Rangaraj M. Rangayyan, and Robson C. Soares. PostgreSQL-IE: An image-handling extension for PostgreSQL. *Journal of Digital Imaging*, 22(2):149–165, 2009.

[26] Jonathon S. Hare, Paul H. Lewis, Peter G. B. Enser, and Christine J. Sandom. Mind the gap: another look at the problem of the semantic gap in image retrieval. In *Multimedia Content Analysis, Management, and Retrieval 2006*, volume 6073. SPIE, 2006.

[27] Yoshiharu Ishikawa, Ravishankar Subramanya, and Christos Faloutsos. MindReader: Querying databases through multiple examples. In *VLDB '98: Proceedings of the 24th International Conference on Very Large Data Bases*, pages 218–227, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[28] Yushi Jing and Shumeet Baluja. VisualRank: Applying PageRank to large-scale image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1877–1890, 2008.

[29] Yushi Jing and Shumeet Baluja. VisualRank: Applying PageRank to large-scale image search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1877–1890, 2008.

[30] Michael S. Lew, Nicu Sebe, Chabane Djeraba, and Ramesh Jain. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 2(1):1–19, 2006.

[31] Qingyong Li, Siwei Luo, and Zhongzhi Shi. Fuzzy aesthetic semantics description and extraction for art image retrieval. *Computers & Mathematics with Applications*, 57(6):1000 – 1009, 2009.

[32] Danzhou Liu, Kien A. Hua, Khanh Vu, and Ning Yu. Fast query point movement techniques for large CBIR systems. *IEEE Transactions on Knowledge and Data Engineering*, 21(5):729–743, 2009.

[33] Ying Liu, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. A survey of content-based image retrieval with high-level semantics. *Pattern Recognition*, 40(1):262–282, 2007.

[34] Apostolos Marakakis, Nikolas P. Galatsanos, Aristidis Likas, and Andreas Stafylopatis. Relevance feedback for content-based image retrieval using support vector machines and feature selection. In *ICANN 2009: Proceedings of 19th International Conference on Artificial Neural Networks*, pages 942–951. Springer, 2009.

[35] Yosi Mass, Benjamin Sznajder, Jonathan Mamou, Michal Shmueli-Scheuer, Sigmund Akselsen, Nicola Orio, and Massimo Melucci. Query language definition for querying combinations of multi-media data, 2008. Deliverable 5.1, SAPIR project.

[36] David Novak, Michal Batko, and Pavel Zezula. Web-scale system for image similarity search: When the dreams are coming true. In *CBMI 2008: Proceedings of International Workshop on Content-Based Multimedia Indexing*, pages 446–453, 2008.

[37] David Novak, Michal Batko, and Pavel Zezula. Generic similarity search engine demonstrated by an image retrieval application. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 840–840, New York, NY, USA, 2009. ACM.

[38] David Novak and Pavel Zezula. M-Chord: a scalable distributed similarity search structure. In *InfoScale '06: Proceedings of the 1st international conference on Scalable information systems*, page 19, New York, NY, USA, 2006. ACM.

[39] Gunhan Park, Yunju Baek, and Heung-Kyu Lee. Web image retrieval using majority-based ranking approach. *Multimedia Tools and Applications*, 31(2):195–219, 2006.

[40] Humberto L. Razente, Maria Camila N. Barioni, Agma J. M. Traina, Christos Faloutsos, and Caetano Traina, Jr. A novel optimization approach to efficiently process aggregate similarity queries in metric access methods. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 193–202, New York, NY, USA, 2008. ACM.

[41] Humberto L. Razente, Maria Camila N. Barioni, Agma J. M. Traina, and Caetano Traina, Jr. Aggregate similarity queries in relevance feedback methods for content-based image retrieval. In *SAC '08: Proceedings of the 2008 ACM symposium on Applied computing*, pages 869–874, New York, NY, USA, 2008. ACM.

[42] Yong Rui, T.S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: a power tool for interactive content-based image retrieval. *Circuits and Systems for Video Technology, IEEE Transactions on*, 8(5):644–655, 1998.

[43] Heng Tao Shen, Shouxu Jiang, Kian-Lee Tan, Zi Huang, and Xiaofang Zhou. Speed up interactive image retrieval. *The VLDB Journal*, 18(1):329–343, 2009.

[44] Tomáš Skopal, Vlastislav Dohnal, Michal Batko, and Pavel Zezula. Distinct nearest neighbors queries for similarity search in very large multimedia databases. In *WIDM '09: Proceeding of the eleventh international workshop on Web information and data management*, pages 11–14, New York, NY, USA, 2009. ACM.

[45] Arnold W. M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(12):1349–1380, 2000.

[46] John R. Smith, Mario Döller, Ruben Tous, Matthias Gruhne, Kyoungro Yoon, Masanori Sano, and Ian S. Burnett. The MPEG Query Format: Unifying access to multimedia retrieval systems. *IEEE MultiMedia*, 15(4):82–95, 2008.

[47] Zhong Su, Hongjiang Zhang, S. Li, and Shaoping Ma. Relevance feedback in content-based image retrieval: Bayesian framework, feature subspaces, and progressive learning. *IEEE Transactions on Image Processing*, 12(8):924–937, 2003.

[48] Seyed M. M. Tahaghoghi, James A. Thom, and Hugh E. Williams. Multiple example queries in content-based image retrieval. In *SPIRE 2002: Proceedings of the 9th International Symposium on String Processing and Information Retrieval*, pages 227–240, London, UK, 2002. Springer-Verlag.

[49] Anthony K. H. Tung, Rui Zhang, Nick Koudas, and Beng Chin Ooi. Similarity search: a matching based approach. In *VLDB '06: Proceedings of the 32nd international conference on Very Large Data Bases*, pages 631–642. VLDB Endowment, 2006.

[50] Reinier H. van Leuken, Lluis Garcia, Ximena Olivares, and Roelof van Zwol. Visual diversification of image search results. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 341–350, New York, NY, USA, 2009. ACM.

[51] Remco C. Veltkamp and Mirela Tanase. Content-based image retrieval systems: A survey. Technical report, Department of Computing Science, Utrecht University, 2002.

[52] Khanh Vu, Hao Cheng, and Kien A. Hua. Image retrieval in multipoint queries. *International Journal of Imaging Systems and Technology*, 18(2-3):170–181, 2008.

[53] Li Wang, Linjun Yang, and Xinmei Tian. Query aware visual similarity propagation for image search reranking. In *MM '09: Proceedings of the seventeen ACM international conference on Multimedia*, pages 725–728, New York, NY, USA, 2009. ACM.

[54] Ning Yu, Khanh Vu, and Kien A. Hua. An in-memory relevance feedback technique for high-performance image retrieval systems. In *CIVR '07: Proceedings of the 6th ACM international conference on Image and video retrieval*, pages 9–16, New York, NY, USA, 2007. ACM.

[55] Pavel Zezula, Giuseppe Amato, Vlastislav Dohnal, and Michal Batko. *Similarity Search - The Metric Space Approach*, volume 32. Springer, 2006.

[56] Xiang S. Zhou and Thomas S. Huang. Relevance feedback in image retrieval: A comprehensive review. *Multimedia Systems*, 8(6):536–544, 2003.

[57] H. Zitouni, S. Sevil, D. Ozkan, and P. Duygulu. Re-ranking of web image search results using a graph algorithm. In *ICPR 2008: 19th International Conference on Pattern Recognition*, pages 1–4, 2008.

Current study results

I became familiar with the principles of content-based retrieval during the last year of my master studies, when I studied and implemented one type of similarity query as a diploma project. The results of this work were presented at the First International Workshop on Similarity Search and Applications (SISAP 2008) in Cancun (Mexico) [7]. In the following three semesters of my doctoral study at the Faculty of Informatics, Masaryk University, I focused on three aspects of searching in metric data: (1) efficient techniques of retrieval using approximation, (2) properties of large data collections, (3) query refinement strategies.

Approximate searching

In the first months, I studied the *locality sensitive hashing* approach to data management and its possible application for similarity searching. Later I proposed a theoretical probability model of the *M-index* structure to evaluate the benefits of combining several such indexing structures. We intend to use this model in future to enable dynamic tuning of search performance.

Properties of large data collections

To prepare for the research of query postprocessing strategies, I studied the properties of large datasets in the spring semester. In particular, I evaluated the influence of the descriptors used in the *CoPhIR* [10] data collection on the search results and analyzed some of their properties (correlation, level of significance, suitability for various image types). This work was presented at the Second International Workshop on Similarity Search and Applications (SISAP 2009) in Prague (Czech Republic) [6].

Query refinement

The refinement of the query results has been the main topic of my study. I summarized the main strategies in a poster *Metric Query Processing* presented at the Seminar of Informatics. As a first step, I implemented several reordering methods for the *MUFIN* similarity search system. I also proposed three possible implementations of relevance feedback mechanism for the distributed architecture of *MUFIN*, exploiting approximation to cut the evaluation costs. At present, I am preparing user-satisfaction experiments to evaluate the performance of these methods. In December 2009, I have succesfully defended a proposal for the *Query Language for Similarity Searching Paradigm* project and have been awarded a scholarship of the South Moravian Centre for International Mobility to support this project.

Apart from the research activities, I also participate in the education process. In Autumn 2008, I conducted the course IB102 Automata and Grammars as a seminar tutor. In Spring 2009, I assisted in teaching of IB108 Algorithm Design II. In Autumn 2009, I conducted the course IB102 Automata and Grammars as a seminar tutor and the PB154 Database Systems course as a lecturer for hearing-impaired students of Teiresias center.

Appendix A: Publications

Here we provide a full text of research papers written by Petra Budíková (née Kohoutková):

- Michal Batko, Petra Kohoutková, and Pavel Zezula. Combining metric features in large collections. In *Proceedings of the 24th International Conference on Data Engineering Workshops, ICDE 2008, April 7-12, 2008, Cancún, México*, pages 370–377. IEEE Computer Society, 2008.
- Michal Batko, Petra Kohoutková, and David Novák. CoPhIR image collection under the microscope. In *SISAP '09: Proceedings of the 2009 Second International Workshop on Similarity Search and Applications*, pages 47–54, Washington, DC, USA, 2009. IEEE Computer Society.