THE FACULTY OF SOCIAL SCIENCES,

DEPARTMENT OF INFORMATION SCIENCE AND MEDIA STUDIES

Utilizing context in ranking results from distributed image retrieval – the CAIRANK Prototype

THESIS

Submitted in partial fulfilment of the requirements for a

Master's degree in information science

By Christian Hartvedt

March 2007

Acknowledgements

Writing this master's thesis has been an interesting and educational process. It has provided me with the possibility of pursuing and investigating some aspects within the field of information retrieval that I personally find very interesting. Several people have contributed to the work presented in this thesis, and they all deserve the appropriate acknowledgements.

First and foremost, I would like to thank my supervisor, Associate professor Joan C. Nordbotten. Her guidance, support, and patience, have proved invaluable throughout this project. Secondly, I would like to thank Lars-Jacob Hove, who in Joan's absence did his utmost to support and guide me in my work.

I would also like to thank Nina Bergli, a fellow student at the time I started my project, for giving me an introduction to the properties of the IBM DBS as well as providing me with AIV and NSE extenders. Kjetil Johnsen also deserves credit for his insightful pieces of advice on how to solve challenges when using Oracle Text. Thanks also goes to Mannaf Haque for interesting discussions, and valuable input regarding programming solutions when using JDBC with the Oracle database system. I'm grateful to Weiyi Meng, professor at the Department of Computer Science of Binghamton University, for taking the time to explain to me the nature of the constant presented in his formula used to describe and review the ProFusion approach.

In addition, I would also like to thank my fellow student Odd Arne Maridal for many interesting discussions throughout this project. Good luck in Oslo!

Table of contents

List of figures

List of Tables

1 The Challenge of Distributed Image Retrieval

Imagine writing an article or paper on bridges, honouring these majestic engineering marvels that bear witness to the ingenuity and impressive achievements from thousands of dedicated construction workers around the world. In order to portray different bridges and illustrate different aspects of the various bridges, images may be an enriching supplement to the text. If you are unable to specify an image query using text, e.g. if you do not know the name of more than a few bridges, retrieving images using the depicted image content is an alternative.

If using an image of a bridge as a starting point for a query, you may be interested in a result set showing various bridges in a given circumstance, or context, rather than images very similar to the one you already have. These contexts could be different weather conditions, different hours of the day, bridges under construction, or bridges collapsing. If this is the case, the relevance criterion for the returned results is not only that they resemble the original image, but also that they depict bridges in the desired conditions.

This thesis focus on how to improve the process of merging and ranking retrieved result sets from queries submitted to distributed image collections by utilizing stored context descriptions. The work will highlight some difficulties associated with similarity-based ranking of multiple result sets in image retrieval, and present an alternative approach to the merging and ranking process. This approach, represented by the *Context Aware Image Ranking* (CAIRANK) prototype, draws from both text-based query methods and content-based query methods in order to improve the result.

Here, images are retrieved from various databases and merged by combining the use of content-based retrieval algorithms and text-based retrieval algorithms. This approach is considering both image content and image context when processing result sets. The results from both algorithms form the basis for the ranking of the query results.

1.1 Information Retrieval and Image Retrieval

The theoretical foundation for this thesis stems mainly from the field of information retrieval. This field is according to Baeza-Yates and Ribeiro-Neto(1999), concerned with *representation, storage, organization of, and access to information items* (Baeza-Yates & Ribeiro-Neto, 1999). Although information retrieval traditionally meant retrieving documents containing text from a single source, it has evolved to also including retrieval of information in the form of audio, video and images from various sources.

This thesis will draw from work done by both the *Database Management and Information Retrieval* research communities, as well as the *Computer Vision* research community as a theoretical foundation. The former two are primarily focused on text-based image retrieval, and have been a part of the information retrieval field since the 1970s. The latter, although being a wide research area, is also focusing on image retrieval using the visual content of an image, such as colour and texture, rather than relying on text-based keywords for retrieval purposes (Rui *et al.*, 1999).

1.1.1 Text-based Image Retrieval and Content-based Image Retrieval

Retrieving images through text-based queries has a long tradition within the field of information retrieval. The annotations or text-descriptions that many text-based retrieval techniques rely upon are valuable in describing both the semantic content of images as well as

the objects portrayed in them. However, without accessible annotations, the text-based approach to image retrieval is not feasible (Lu, 1999).

The process of adding annotations to all images can be a quite time-consuming task when maintaining large image collections. It is also possibly both biased and error prone in that the semantic information added about the images and the objects portrayed, may be a result of the annotator's own interpretation. Any mismatch between keywords provided by the annotator, and terms specified by a user, will result in failure to retrieve a given image even if the image in fact satisfies the query criteria.

Although keyword searching is still very common in image retrieval, *Content Based Image Retrieval* (CBIR), utilizing the global and local low-level features of an image as search criteria, emerged during the 1990s as an alternative to text-based queries. To achieve this form of image retrieval, feature extraction techniques gather visual features from the images for indexing and later retrieval purposes. These features include colour, texture, shapes and spatial placement (Rui et al., 1999). After extracting these features, the DBMS generates a signature of the image, providing a possibility of comparing visual features between images. Drawing on this technique, *Query by Example* (QBE) methods, make use of a seed image in order to locate other images resembling this image based on syntactical similarity (Xiangyu & James, 2003).

The overall degree of success using CBIR and QBE is rather limited, perhaps due to a lack of semantic interpreting abilities in the retrieval system. This deficiency often results in retrieved images with a strong syntactical resemblance, but having little or no actual information value, as illustrated in figure 1 adapted from Hove (2004). Both images have a distinct curved shape, and they are both without colour, giving the retrieval system less data to draw from when calculating similarity. From a structural or syntactical point of view, these two images have many similar features, despite the fact that they do not look alike to a human eye.

Thus, a system executing a QBE query may return images based solely on the extracted feature data available in indices, and if not considering semantic information, the result can be somewhat haphazard. This is an example of the problem called the *semantic gap* in image retrieval.

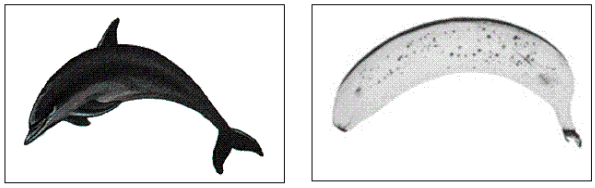


Figure 1 - The problem with syntactical resemblance vs. semantic relationship

1.2 Ranking Results in Information Retrieval Systems

The results obtained through queries submitted to an *Information Retrieval System* (IRS) most often appear as lists, organized according to some criteria like similarity, either determined by the user or set as default by the *database management system* (DBMS) responsible for operating the database.

In order to provide the user with a ranked list, an IRS uses statistical and mathematical methods to calculate the similarity of items stored in the database against criteria given in the form of a query. Based on these calculations, every item retrieved from the collection is arranged according to a score or rank position showing how similar the DBMS calculates the item to be, given a specific query.

1.3 The Challenge of Ranking Distributed Image Queries

The way in which different database systems represent images digitally is perhaps the most serious challenge when retrieving images from several different systems. This is because retrieval and ranking of query results from distributed image queries is based on calculations done by local ranking algorithms in the different sources (Steidinger, 2000), i.e. these problems occur because different systems use different algorithms for feature extraction, signature generation and image retrieval. Signatures are also system dependent, and ordinarily one DBMS cannot retrieve images directly from a database developed by another manufacturer, leaving a similarity score generated by the different DBMS' as the only piece of data available. These data may not be compatible, which can result in the displacement of relevant images by images less relevant because different DBMS' are using different measurement scales for assigning similarity scores.

Even with the problems associated with retrieval algorithms and different representation approaches solved, the fundamental problems of information retrieval, described in section 1.1.1, prevail. The most likely outcome for the user is ending up with images evaluated as relevant by the DBMS, which in fact are useless in satisfying the user's information need. These problems affect all who retrieve images through both text-based image retrieval and content-based image queries, but perhaps even more so in a distributed environment containing DBS' from different producers.

Submitting queries to several databases simultaneously thus creates the need for a retrieval system able to handle and sort through multiple result sets in order to rearrange them into a global list. The purpose of this process is to take the retrieved results from the query and merge them into one result set, presenting the most relevant items first regardless of source (Voorhees *et al.*, 1994). Common solutions used are specialized middleware systems acting as a layer between the user and the participating DBS. However, these middleware systems are usually quite expensive to develop, in that they are custom made for each area of application (Fan *et al.*, 2004b).

There exist several other approaches to this merging process using different methods for ranking multiple query results into one merged list. Some methods interleave the scores by always picking the items with the highest similarity score, whereas others merge result sets using normalized scores or utilize statistical meta-information about the content in the local result sets in order to normalize them.

According to Charles Peirce, relying on theory from the field of semiotics, images lack the ability to provide users with information on their own (Hausken & Larsen, 1999; Kjørup, 1978). The photographer or the author will always have to specify the semantic information regarding both the content and the context of an image. According to this line of thought, the context information associated with an image increases in value, because it captures the context seen from the author's point of view. A common way of presenting context information is through annotations in the form of keywords or short descriptions (Lu, 1999).

There has been some previous work done on image retrieval using both keywords and image content in the retrieval process. The slightly different approach to recording context information taken in this project is to use full-text documents consisting of information stubs related to the depicted image content. Using this approach may ease the burden of providing all images with context information in that different images may use the same information stubs in the image description. In addition, this approach may utilize the functionality developed for full-text search when evaluating similarity.

The main motivation behind this thesis has been to evaluate if the utilization of existing DBMS functionality in a new way could contribute in narrowing the Semantic Gap. The approach proposed in this thesis seeks to combine some of the techniques and methods of text-based information retrieval with those of content-based retrieval when merging and ranking results from distributed image retrieval.

1.4 Proposed framework – Ranking Distributed Results Utilizing Context

Access to distributed databases can prove valuable in satisfying users information needs if the databases share their contents in a collaborative manner. In this form, the various sources can complement each other, thus provide a collective result greater than the sum of its parts.

One area of application where sharing of information recourses can occur, is if several museums or libraries cooperate in presenting their stored material, e.g. with an organizational structure like that of a consortium. Here, multiple entities can participate in a common activity by pooling their recourses in order to achieve a common goal[1]. Thus, different museums and libraries can present digital versions of their material, for instance in joint virtual exhibitions or as a collaborative media library[2].

Important tasks for these collaborating organizations would be to present their material to the public through cooperation, while simultaneously being able to control the contents of their exhibits, thus enforcing their proprietary rights. An integrated database solution might therefore not be feasible.

In addition, as different museums also could be interested in different aspects and contexts of the same physical items, these differences could manifest themselves in images taken of the same items. For instance, a historical museum, an art museum, and a science museum would probably take different perspectives when viewing the same item, and hence possibly store different images of it, as displayed in figure 2.

Figure 2 - Three different perspectives on the same structure

[1] http://www.m-w.com/dictionary/consortium
[2] http://images.fws.gov/

These images all depict the first bridge crossing the river Severn in the United Kingdom. In addition to storing different images, the different types of museums mentioned above, might also perceive different context information as being interesting. From a historical point of view, interesting facts would probably include that construction work on this bridge ended in 1966, and that it crosses the river Severn taking almost exactly the same route as the former ferry service crossing from Aust Cliff to Beachley Peninsula[3]. In an art museum, focusing more on aesthetic and artistic qualities, the middle image probably would be of greater value in a photo exhibition and described accordingly, whereas a science museum would probably focus more on the engineering aspect, illustrated by the image to the right. Interesting facts here, could be that this was the first bridge to use an aerodynamically shaped deck to avoid the weight and complexity of a truss[4], or that it vas the first suspension bridge with hollow box girder with an aerodynamic profile [5].

In order to make this context information available to help in answering queries, annotations could be associated with each image. Using this approach, each museum would probably annotate the stored images with keywords reflecting the domain ontology of the field the museum represents, i.e. history, art or science.

The aforementioned problems associated with annotations aside, this solution could ensure that valuable information was available to support context queries. Nevertheless, all context information given for the images in figure 2 could be equally important to a user since all the images are in fact displaying the same item. Some information should therefore be associated with all the images. Solving this task using annotations would probably be an all too daunting undertaking.

Alternatively, the cooperating museums could make use of external information sources to describe image context by using full-text descriptions. With this solution, they would help ensure a uniform context description across collections. In doing so, museums may provide a better chance to retrieve relevant contexts for images depicting the same item even if located in different collections.

Following from this, the assumption made in this thesis is that having several different domain specific databases available simultaneously - all providing a uniform context information for all images stored in the collections - could potentially provide a richer and more complete pool of information resources than databases in any one museum could achieve on their one.

If results from queries against these cooperating collections were to be based only on the similarity criteria present in CBIR, i.e. having queries retrieve images based solely on similarity scores obtained by comparing low-level features, utilizing the potential of this information pool could prove to be very difficult.

To illustrate this problem, we can think of a query using the middle image in figure 2 as a seed image when looking for images like that on the right-hand side. For the human eye, these two images are similar to the degree that we probably would not have major difficulties in identifying them as depicting the same bridge. In addition, we can also clearly see that some of the things that set them apart are that two of the images have multiple colours while the

[3] http://en.wikipedia.org/wiki/Severn_bridge
[4] http://www.brantacan.co.uk/suspension.htm
[5] http://www.ce.berkeley.edu/~filippou/Courses/L&S122/Suspension_Bridges.pdf

third is in black and white. The images also depict the bridge at different hours of the day, and the picture to the right show that the bridge in is not yet completed.

Drawing the same conclusions from comparing low-level feature signatures of these images would probably not be so simple, and this example illustrates some of the problems associated with the semantic gap in image retrieval, described in chapter 2.

On the other hand, the abovementioned interpretation of the images was mostly dependent of information about general contextual aspects, like time of day or the concept of construction work. This type of information could also be stored in the databases, and thus be made available for use by the user and the DBMS.

Having several photos of a given number of brides may also result in images representing several different contexts. Examples of different contexts in pictures of a bridge can be images taken at different stages in the construction period, images taken at different hours of the day, and images taken in different weather conditions. Figure 3 illustrates an example on how different context can be associated with various images of the same bridge.

Figure 3 - Three images of the same structure illustrating different contexts

Depending on the specific nature of the information need a user may have, the three images in figure 3 will probably vary in how relevant they are. Images of a foggy day on the bridge, or lightning striking the bridge, will most certainly have less relevance if the images needed are images depicting the construction stages of bridges.

A different challenge may occur when a given image contains several different contexts, as illustrated in figure 4. The same image may thus be relevant in different queries despite variations in the degree of resemblance to a seed image.

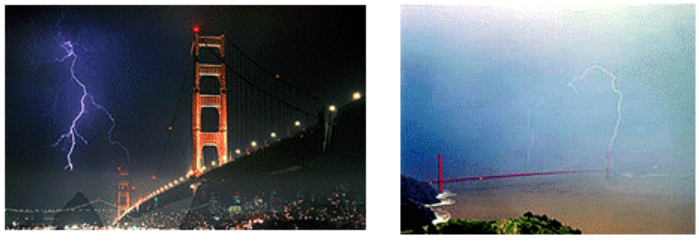


Figure 4 - Two images of the same structure illustrating multiple contexts

Both images in figure 4 illustrate as least two different contexts each. From a syntactical point of view, these bridges are not very similar, but with regard to semantics, both images could prove relevant if the user is interested in images depicting Golden Gate Bridge when being struck by lightning.

1.5 Research Project

This research project will focus on evaluating if image retrieval based on the utilization of image context can contribute to a higher degree of relevance when merging and ranking query results from distributed image retrieval.

1.5.1 Research Question and Hypotheses

In order to address the problem area described here, the research question forming the base for the project is:

> *Can combining similarity scores from both text- and content-based queries significantly improve the process of merging and ranking multiple result sets from distributed image retrieval compared to Raw Score merging using content-based similarity scores only?*

The following hypotheses form the base for investigating the research question:

> H1:
> *Combining similarity scores from both text- and content-based queries will significantly improve Precision when ranking results from an image database compared to ranking using scores from content-based queries only.*
>
> H2:
> *Merging and ranking query result sets from multiple database systems by combining similarity scores from both text- and content-based queries will significantly improve precision compared to ranking using a Raw Score merging approach relying on content-based similarity scores only.*

The first hypothesis, if verified, gives an indication on whether the combination of similarity scores from text and content-based queries improves the precision of the ranked sub-results provided by each participating database system. The second hypothesis, if verified, gives an indication on whether combining scores, normalization and weighted merging is capable of improving the precision of the results than using single similarity scores only.

Measuring recall and precision is a widely used method in evaluating the quality of a retrieval system (Baeza-Yates & Ribeiro-Neto, 1999; Lu, 1999). Measuring performance using recall/precision makes it relatively straightforward to compare different retrieval techniques. In evaluating the second hypothesis and investigating the research question put forth in this thesis, Precision was a suitable tool while Recall was omitted because both approaches acted on the same result sets, i.e. the relevance of the result sets, measured by Recall, would remain the same in both approaches.

In order to measure the degree of precision in the ranked result sets produced locally in each of the participating systems considered in this project, Distance was used as measure. A Distance diagram and a Distance table illustrated the distance measures for

the two approaches in each result set. The Distance diagram helped show the distance between the placement of a relevant image in the returned result set and its placement in an ideal response set. The Distance table displayed the dislocation in actual number of places.

1.5.2 Methodological approach

In order to test these hypotheses, an experiment was set up to test two alternative approaches to the process of ranking and merging multiple result sets and to the process of ranking results based on either image similarity scores alone versus a combination of image and text similarity scores.

As this thesis focus on image retrieval in a distributed environment, functionality for communicating with users, submitting queries to and retrieving results from different DBS' is required. A system environment thought capable of doing this is illustrated in figure 5.

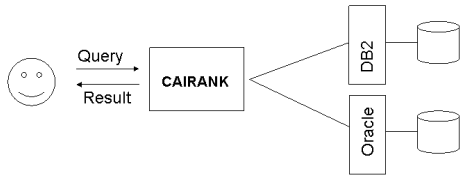


Figure 5 - Overview of the CAIRANK system environment

This approach distributes the workload between the different system components, i.e. the participating DBS' and the CAIRANK search engine. Here each of the participating DBS conducts the content-based image retrieval and the text query before storing the similarity scores from the two queries in a temporary table. Then the CAIRANK prototype retrieves and processes the collected similarity scores and calculates a new score. Then the results are sent to an application to be merged and ranked in order for the user to see the result.

The data used as base for comparison for results obtained with the alternative ranking method proposed in this thesis, were results collected using a Raw Score merging method for ranking results. This ranking method merged and ranked results based on a DBMS-assigned content-based similarity score alone.

Statistical analysis of the performance of the ranking approaches considered here was used to measure the significance of any differences in performance observed in the experiment.

2 Theoretical Framework and Literature Review

2.1 Information, Documents and Images – Fundamental Concepts

In this chapter and throughout this thesis, fundamental concepts important for the discussion will be presented and defined[6]. Many of these concepts also serve as a foundation for much of the work presented in this thesis.

2.1.1 Data, Information and Knowledge

Some of the most fundamental and central concepts within the field of information retrieval are those of data, information and knowledge.

> **Definition 1 – *Data*** *are symbols inscribed in formalized patterns, representing facts, observations and/or ideas, that are capable of being communicated, interpreted and manipulated by some human or mechanized process* (Nordbotten, 2006).

The definition above does not limit data to be facts only, and data may thus represent stories, messages, ideas, narrative blocks of text and other information:

> **Definition 2 – *Information*** *is the meaning that a human extracts from data by means of known conventions of the representation used* (Gould, in Nordbotten, 2006).

At the heart of this definition of information lies the notion that although data may be valuable for both man and machine as data, the transformation into information does require a human to process the data using previous knowledge, defined here as:

> **Definition 3 – *Knowledge*** *is the fact or condition of being aware of, or knowing, something with familiarity gained through experience or association, by acquaintance with or understanding of, a science, art, or technique, thus apprehending truth or fact through reasoning.*

Having the ability to process and store digital representations of data offers great possibilities, but it also creates the need for differencing between the source data and the digital version. This alternative version is commonly referred to as media data (Nordbotten, 2006), defined as:

> **Definition 4 – *Media data*** *is digital data used to record the information presented in a particular type of media object, f. ex. text, image, sound, or tables (of alphanumeric data)* (Nordbotten, 2006).

[6] See appendix A for a compiled list of all definitions. Where no references are given, definitions are put together by the author, based on various sources and lexicographical definitions.

2.1.2 Text, Images and Documents

Text and images are quite versatile in use, and offer great diversity regarding what types of source data it is possible to represent and record.

Concerning text, this is not merely a collection or set, of signs or symbols:

> **Definition 5 –** ***Text*** *is the vehicle of a communicative act when expressing something using written words in accordance with grammar.*

In addition to text as a means of representing data and information, images are also a viable alternative for this task. Through different forms of representation, images have a long tradition as information providers. Images represent a reality as seen in the eyes of the creator:

> **Definition 6 –** *An* ***image*** *is a visual representation of an entity or entities, produced on a medium.*

Converting images into media data creates new possibilities concerning both manipulation and area of use. By using specially developed mathematical methods for extracting the visual features from images and represent them digitally, it is now relatively straightforward to record them in a digital form. The following definition of digital images is proposed:

> **Definition 7 –** *A* ***digital image*** *is a photograph or graphic, composed of discrete pixels of digitally quantized brightness and colour, created or rendered on a computer from an ultimate input source such as a digital camera or a scan of an image.*

Digital images consist of a rectangular array, with a fixed numbers of pixels on a horizontal line and a fixed number of lines in the vertical dimension. Digital images can be in greyscale or in colour. Pixels in a greyscale image vary in their brightness and intensity, presented in a two-dimensional array. Pixels in a colour image only differ from greyscale images in that we need three colours – red, green and blue – to represent each pixel. Colour images are therefore represented by three two-dimensional arrays, each array corresponding to each of the primary colour components in the image (Lu, 1999).

When using text and images for communication purposes, it may be in the form of documents:

> **Definition 8 –** *A* ***document*** *is a representation of a unit of information, and may consist of plain or formatted text, images, inline graphics, sound, other multimedia data, and/or hyperlinks to other documents.*

2.1.3 Concerning Image Content

In order to get a clear understanding of some fundamental concepts concerning both the syntax and the semantics of image content, this thesis draws from the framework presented in Jaimes and Chang (2002) , summarized below.

Percept vs. Concept
Taking in an image through our visual senses by perceiving the different patterns of light only, gives us a neutral perception of the different elements in an image. The percept thus refers to what our senses perceive, without assistance from any higher mental processes. A concept refers to a representation, an abstract or generic idea, generalized from particular instances. This implies perceiving an image with the support of background knowledge and an inherent interpretation of the content. Viewing these terms in light of the definitions of data and information, presented in section 2.1.1, percept refers to image data, whereas concept refers to the information present in an image.

Syntax vs. Semantics
While percept is associated with the impressions we perceive, syntax refers only to the visual elements themselves and their arrangement or composition. The way in which we perceive an image may thus differ, but the syntax stays the same. Semantics refers here to the meanings of the elements themselves, as well as their arrangement or composition.

General vs. Visual Concepts
A general concept can include any kind of attribute, whereas a visual concept includes only visual attributes. An important aspect concerning general concepts is that different observers can have differences in their concepts and often see objects at different conceptual levels. Take the Golden Gate Bridge for instance. An engineer specialising in bridges, a tourist of foreign nationality, and a filmmaker shooting a documentary on the subject of suicides from the Golden Gate Bridge[7], may all have a different general concept of the bridge because of differences in their background knowledge and interpretation.

Visual vs. Non-visual Content
The visual content of an image corresponds to features directly perceived when observing an image, e.g. lines, shapes and colours. The non-visual content corresponds to information closely related to the image, but that is not present in it. Annotations like the name of the photographer or the location of the shot belong in the non-visual category.

One noteworthy approach to the study of visual image content is Panofsky's image analysis, called iconology, developed from his iconographic approach to the interpretation of motifs. Here, Panofsky classified images as having three different levels of meaning (Schwebs & Østbye, 1999):

- *The primary, or natural, meaning.* Here, image descriptions focus on the lines, shapes and colours used to portray the recognizable objects in the image, e.g. persons, items or landscapes. This is the pre-iconographic description level in Panofsky's classification. Thus, no special knowledge is required in order to describe images at this level.
- *The secondary, or conventional, meaning.* On this level, culture determined conventions form the basis for the interpretation of motives and symbols present in the image, referred to as an iconographic analysis. Interpreting images at this level requires knowledge on customs and traditions regarding both the time-period and the culture the image come from.

[7] Filmmaker Eric Steel caused an uproar with the film project proposed as a "monuments documentary" intended to "capture the grandeur" of the Golden Gate Bridge, but instead ended up as a suicide documentary. http://en.wikipedia.org/wiki/Eric_Steel

- *The internal meaning, or content, of an image.* Here, it is a quest for the inner or deeper meaning of an image. In addition to knowledge required in the iconographic analysis, a prerequisite for this iconological interpretation of an image is knowledge about both history and social conditions surrounding the time of creation. This is because the image content represents a manifestation of the spirit of the times. Thus, both intuition and special knowledge is required in order to interpret the image content properly.

From the frameworks presented above, we can now make a distinction between syntactic and semantic image content:

> **Definition 9 – *Syntactic image content*** *is the spatial arrangement of characteristics, like colour, shape and texture, associated with the visual elements perceived in an image.*

> **Definition 10 – *Semantic image content*** *is the meaning given to both the visual elements perceived in an image and the way in which they are arranged.*

The visual content of the image is thus specified and defined by the sum of the various image features present.

> **Definition 11 – *An image feature*** *is a distinguishing primitive characteristics or attribute of an image* (W. Pratt in Bergman *et al.*, 1997).

As discussed above, non-visual content can also contribute in describing the visual image content:

> **Definition 12 – N*on-visual image content*** *corresponds to information that is closely related to the image, but that is not necessarily explicitly given by its appearance.*

Jaimes and Chang (2002) present a simple structure, depicted in figure 6, providing general guidelines for indexing non-visual information.

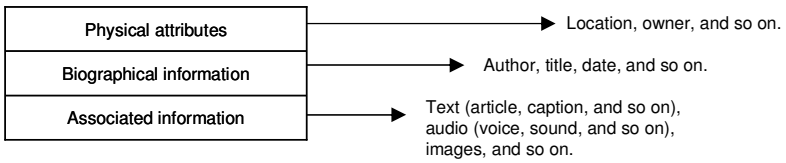


Figure 6 - Classification of non-visual information, from Jaimes and Chang (2002)

Biographical information

Images may have *Bibliographical information* associated with them and this information may repeat itself for different objects in the image, exist for the given image only, or not exist at all. Often, *Bibliographical information* is not directly related to the subject of the image, but rather to the image as a whole, e.g. author, date, title.

Associated information
This concept refers to information directly related to the image in some way, i.e. most often directly related to the subject of the image, serving as a supporting part at the semantic levels in the presentation since they require more knowledge that is often not present in the image alone.

Physical attributes
These attributes describe the characteristics of those that have to do with the image as a physical object.

A common way of providing users with non-visual information is the use of textual annotations in describing the visual content of an image. This approach has proved to work reasonably well for specifying semantic image information (Colombo & Del Bimbo, 2002).

2.1.4 Concerning Image Context

All information not directly derived from the visual properties, or low-level features, constitutes the context of an image (Westerveld, 2000). The Dublin Core metadata standard, used by librarians for documenting texts and images, refers to context as information about the document but not its semantic interpretation. Thus, *author* is considered to be context information, while the *title* is considered to be semantic metadata. Both author and title is referred to as bibliographical data by Jaimes and Chang (2002). Context, as defined in this thesis, is:

> **Definition 13 – *Context*** *is any information that can be used to characterize the situation of an entity* (Dey *et al.*, 1999).

Like in Karlsen and Nordbotten (2005), an entity is in this thesis first and foremost an image, but may also be a user searching for images to be used in a specific situation.

Dey et al. (1999) have developed a framework for capturing context in terms of four different categories:

- Activity
- Identity
- Location
- Time

Activity, answers a fundamental question of what is occurring in a situation. Examples are lightning, fog, construction etc. *Identity* provides us with the possibility of acquiring many pieces of information. Examples are or course name, but also other identifying features like aliases or nicknames, e.g. Big Apple and Big Easy for New York and New Orleans, which enable us to distinguish one entity from another. *Location* may be vague or concrete. America is less concrete than Boise, Idaho, which is less concrete than 1600 Pennsylvania Avenue, Washington DC, USA. *Time*, is of course possible to use in order to help distinguishing context in images for instance separating day from night.

2.1.5 Images as Information Providers

Two schools of thought contributing to the theoretical framework for the process of identifying the meaning of image content, are the fields of art-history and semiotics, or

semiology; the study of the function of signs and symbols in human communication. Erwin Panofsky, briefly discussed in section 2.1.3, represents here the first school of thought while Roland Barthes and Charles Peirce, discussed below, represent the latter.

From Panofsky's point of view, as presented above, images do provide users with information extracted through identification, analysis, and interpretation of the three levels of meaning. This reading of image content is dependent on having knowledge of different cultural norms and traditions (Schwebs & Østbye, 1999). Images thus present a readable snapshot of a reality, and as such, are informative.

Barthes theorises that messages found in images rely both on codes being dependent of socio-cultural knowledge, and on codes not dependent on this knowledge. Hence, images are consisting of two simultaneous announcements: The *denotated* announcement, which is codeless, and the *connotated* announcement built up by a combination of the image presentation and the image content (Schwebs & Østbye, 1999). The presentation element comes from the author's processing of the image, while the content element refers to the cultural understanding present in the society receiving the announcement.

According to Barthes, content captured in photographs thus consists of two separate levels of meaning, also known the primary and the secondary meaning of an image (Hausken & Larsen, 1999). Each level of meaning may provide the user with a different kind of information. As the second level consists of the two different types of announcements, this creates a three-level structure of meaning, somewhat similar to that of Panofsky.

Peirce, taking his viewpoints from the field of semiotics, has a somewhat different opinion whether images alone can serve as information providers. Peirce's ideas about semiotics distinguishes between three types of signs (Schwebs & Østbye, 1999), summarized below:

- *Likenesses, or iconic signs*, resemble the object. These signs serve to convey ideas of the entities they represent simply by imitating them. Examples are drawings, images and maps. Although iconic signs most often are visual, they may also be of a verbal character.
- *Indications, or index signs*, refers to a man made or natural cause-effect relationship between sign and object. They show something about things because their physical connection to them. Examples are smoke (There is no smoke without fire), or a guidepost giving directions.
- *Symbols, or general signs*, are signs where there is no resemblance or cause-effect relationship between sign and object. These signs are associated with their meanings by usage or conventions. Examples are words, phrases, speeches and numbers.

In order for something to serve as an information source, an identification, or indication, of what is referred to is necessary, primarily by specifying time or place. This identification points to the object the sign represents. According to Peirce, neither icons nor symbols can identify what is referred to on their own, and therefore cannot inform. Thus, as images in this view most often lacks the ability to identify the depicted motif on their own, only by understanding the contents captured in the image, the non-visual information and the image context, are we able to tell if, how, and what the images represent (Hausken & Larsen, 1999; Kjørup, 1978).

2.2 Recording and Keeping Information – Database Systems

To be able to retrieve the recorded digital version of the data effectively, we need some coherent structure determining how data are stored. A common tool used for providing this structure is databases, defined in Nordbotten(2006) as:

> **Definition 14 –** ***A database*** *is a logically coherent collection of related data, representing some aspect of the real world, designed, built, and populated for some purpose* (Nordbotten, 2006).

A database management system (DBMS) has built in functionality for storing, accessing, updating and deleting data:

> **Definition 15 –** ***A Database Management System, DBMS,*** *is a system providing 1) a schema defining the structure used for the data that represents the information in a database, 2) a database engine that supports storage, access to and modification of the database, 3) a language for definition and manipulation of the database* (Adapted from Hove, 2004).

DBMS are commonly vendor specific software systems developed to support a particular type of database system (DBS), e.g. business, administrative, bibliographic, image or research:

> **Definition 16 –** ***A Database System, DBS,*** *is an information processing system containing: a DB and DBMS, a number of DB applications, and well as ad hoc user interactions* (Nordbotten, 2006).

2.2.1 Processing and Indexing Information Items

The content recorded to populate the databases constitutes the data items available for retrieval, defined here as:

> **Definition 17 –** ***Data items*** *are the elements forming the total collection of catalogued data possible to locate and display upon request.*

In this thesis, a data item consists of an image with its corresponding text description. In order to be able to store, manipulate and use/retrieve documents effectively for retrieval purposes in database systems, the DBMS is required to perform some pre-processing operations on the data items.

The results of pre-processing operations preformed by the DBMS are commonly contained in the form of one or more indexes:

> **Definition 18 –** ***An index*** *is a data structure constructed from the data items to speed up searching and retrieval. The structure consists of terms used to refer to the content of a data item.*

As indices derived from images are different in structure and content compared to indices derived from text, a brief presentation of some common pre-processing operations follows below.

Pre-processing text
As the goal of a retrieval system is to retrieve relevant documents from the stored collection, the documents must be stored within a coherent structure. Several different structures exist (Lu, 1999), but a commonly used file structure is an inverted file in which the index consists of the terms used in the document collection and references to where, or in which documents the term occurs (Baeza-Yates & Ribeiro-Neto, 1999; Lu, 1999). The purpose of the index is to improve retrieval efficiency or performance.

Another solution used to improve performance is term processing, e.g. identifying term roots, or the removal of stop-words. The first operation reduces distinct words to their common grammatical root, while the second operation remove words which are very frequent and do not carry independent meaning (Baeza-Yates & Ribeiro-Neto, 1999). Improvements in performance may also be achieved by adding a thesaurus or dictionary.

Pre-processing images
In order to use image content for content-based retrieval purposes, the DBMS must extract and record the syntactical image features. Thus, the first pre-processing step is commonly called feature extraction:

> **Definition 19 – *Feature extraction*** *is the process of extracting structural data from a digital image that is then used by the DBMS to classifying the syntactical image content.*

By processing the raw data, the DBMS generates an overview and description of the features present in an image using descriptors:

> **Definition 20 – *Feature descriptors*** *are descriptors generated by the DBMS, capturing the specific visual characteristics in an image.*

As one image feature alone rarely is enough to describe the content of an image adequately, neither will a collection of isolated feature descriptors satisfactorily achieve this goal. Consequently, most often the DBMS combines several feature descriptors in a vector, constituting a *feature vector*:

> **Definition 21 – *A feature vector*** *is a set of descriptors describing one or more syntactical image features, represented as a binary string* (Hove, 2004).

The feature vector is comprised of a specific set of feature descriptors, resulting in a distinctive *image signature.*

Lu (1999), describes four levels of features and attributes: metadata, text annotations, low-level content features and high-level content features. The most common low-level features

used in content-based image retrieval techniques are colour, shape, texture, and spatial relationships.

2.3 Retrieving Image and Text Documents

The principles for the approach to document retrieval taken in this thesis, come from the field of information retrieval. According to Baeza-Yates and Ribeiro-Neto (1999), an information retrieval model is comprised of four components:

1. A set composed logical views, i.e. representations or signatures of the documents in the collection.
2. A set composed of logical views (or representations) for the user information needs, most often represented in the form of queries.
3. A framework for modelling document representations, queries, and their relationships.
4. A similarity function, which associates a real number with a query and a document representation. This number can be used to define an ordering among the documents with regard to the query.

2.3.1 Text-Document Retrieval

Queries executed against a document collection represent the information need as specified by the user. Before being returned to the user, the retrieved set of documents are ranked according to a likelihood of relevance (Baeza-Yates & Ribeiro-Neto, 1999). This approach to result ranking arranges the result set according to the documents *similarity score*, defined in this thesis as:

> **Definition 22 – *A document similarity score*** *is a numerical score assigned by the DBMS to each query-document pair in the collection, indicating how well the documents meets an information need specified through a query according to evaluation criteria implemented in the DBMS.*

Traditionally, text similarity scores are the result of examining how many times a query term appears in a document using algorithms for calculating term frequency. A common algorithm used for generating document similarity scores, combines the term frequency approach with an examination on how discriminative that query term is across the collection, resulting in a measurement referred to as *term frequency-inverse document frequency* (tf-idf). The query result usually consists of the documents with the best query-document score, sorted by the similarity score.

2.3.2 Image Retrieval

Two main approaches to image retrieval is low-level image retrieval and high-level image retrieval. Low-level retrieval is based on the use of an integrated feature-extraction/object-recognition subsystem that automates the process of feature-extraction and object-recognition. Low-level retrieval may also be based on the use of low-level image features to index images, and later retrieve images based on similarity.

By contrast, high-level retrieval is based on image content modelled as a set of manually assigned attributes managed within the framework of a conventional DBMS, or by annotating images using free text, and then employ information retrieval techniques to carry out image retrieval (Lu, 1999).

These two approaches to image retrieval are discussed further in the following sections.

2.3.3 Low Level Image Retrieval

Low-level image retrieval using image features for indexing the images, and later retrieve images based on these image features, is commonly referred to as content-based image retrieval:

> **Definition 23 –** ***Content Based Image Retrieval (CBIR)*** *is the process of retrieving images based on low-level features automatically extracted from images for retrieval purposes.*

CBIR came from a need for support for image retrieval in areas where textual descriptions of images are not feasible, e.g. satellite images, finger prints or x-rays (Karlsen & Nordbotten, 2005).

Low-level content-based image retrieval techniques ordinarily use *similarity* as criterion when searching for relevant images based on their low-level features, i.e. colour, texture, shape or spatial properties (Zhao & Grosky, 2001). By comparing image signatures assigned to all images in the collection, the DBMS can evaluate how similar the content of one image is to the content of another.

As discussed in section 2.2.1, an image signature is the product of extracting the low-level image features into a feature vector. Users can query low-level features by specifying which colours to search for, or by using an example, or seed, image for comparison. This query method is ofthe referred to as query by example (Xiangyu & James, 2003).

Typically, the DBMS retrieves images based on a calculated similarity score assigned by the DBMS to the images in the query collection. This score is based on how close their structural similarity is to the query criterion (Kretser *et al.*, 1998). An image similarity score is in this thesis defined as:

> **Definition 24 –** ***An image similarity score*** *is the output from calculating the distance between the image signature of images in the collection and the image signature of a seed image.*

2.3.4 Semantic Gap

Although CBIR systems support automatic registration of low-level image features, they lack the support for image retrieval based on high-level semantic concepts (Karlsen & Nordbotten, 2005). This lacking ability to extract and interpret semantic information in images is one of the major drawbacks associated with CBIR, and using low-level features to correspond to high-level abstractions is one aspect of the *semantic gap* between content-based organisation and the concept-based user (Zhao & Grosky, 2001).

An important part of the semantic gap problem, is the fact that visual similarity does not necessarily correspond to a semantic relationship. Therefore, the images retrieved on the basis of an example image are not necessarily related to it on a semantic level (Westerveld, 2000). The semantic gap thus refers to the discrepancy that exists between the information currently

possible to extract from visual data and the interpretation the same data has for a user in a given situation (Smeulders et. al in Dorai & Venkatesh, 2003).

Consequently, the CBIR approach relying on low-level features alone may or may not prove valuable. This depends on if the search for images has *similarity* or *relevance* as criterion for satisfying an information need, e.g. a similar image may prove irrelevant, or a relevant image may look very dissimilar.

2.3.5 High Level Image Retrieval – Utilizing Semantic information

The main strength of high-level image retrieval is the possibility to support semantic retrieval, by offering annotations describing the semantic image content in an effort to help ensure the retrieval of data items with a high degree of *relevance*:

> **Definition 25 – *Relevance*** *refers to what extent a data item contains the semantic properties needed to satisfy the information need of a user for a given query.*

By using annotations, an image description can cover the different levels of meaning discussed in section 2.1.3, thus describe both the visual content captured in the image and the non-visual meaning behind the image content. The annotations are later used in a text-based search (Baeza-Yates & Ribeiro-Neto, 1999), commonly based on evaluating similarity between a string of text given by the user and recorded annotations describing the non-visual image content of images in the collection.

Even though annotations may prove very valuable in describing semantics and the non-visual image content, there are two important problems. Firstly, as annotations most often represent the annotator's view of the image, described in the annotator's vocabulary, they may be biased. Secondly, annotating images is traditionally a manual and time consuming task, increasing the likelihood for an image only to be annotated with a small subset of the possible semantic interpretations (Karlsen & Nordbotten, 2005). As a consequence, retrieval based on keywords may be of fairly low quality due to the insufficient use of keywords or failure to capture the semantic content properly (Baeza-Yates & Ribeiro-Neto, 1999).

An alternative to using manually generated keyword annotations alone when describing image content is the use of full text descriptions, or documents (Lu, 1999). Here, the non-visual content acts as a complete logical unit. Using this approach provides a possibility for extracting keywords directly from the text, thus providing a *logical view* of the document, representing it through a set of index terms (Baeza-Yates & Ribeiro-Neto, 1999). These operations reduce the complexity of the document, enabling faster searching by structuring the index terms into an index.

A user study preformed by Markkula and Sormunen (2000), indicated that most of the users participating were interested in semantic entities rather than in visual appearance. From this, an argument questioning the usefulness of image retrieval could be put forward, but Westerveld (2000) claims that images are not only used to show a certain concrete object, but are also supposed to express a certain feeling. Simple textual queries cannot easily accommodate all these information needs, thus supporting the collaboration of both text-based retrieval and content-based retrieval as a viable solution when searching for images.

2.4 Combining Low- and High Level Image Retrieval

As discussed in the previous sections, there are both strengths and weaknesses associated with the low-level approach and the high-level approach to image retrieval.

The more severe deficiencies in the former approach lie in that current low-level image retrieval cannot capture the high-level abstractions contained in images. Limitations associated with the latter approach is that the current high-level image retrieval methods may be partial or incomplete and subjective (Lu, 1999).

In addition, specifying the syntactic content using written words in the form of annotations is often quite challenging.

A sensible approach when trying to avoid some of the weaknesses associated with low-level and high-level image retrieval respectively, while still capitalizing on their strengths, could be to integrate the two approaches (Lu, 1999; Müller *et al.*, 2005; Westerveld, 2000; Zhou & Huang, 2002).

A possible structure for a database management system utilizing both image content and text in image retrieval, is the image retrieval system architecture presented in Rui et al. (1999), see figure 7.

Here, the search engine retrieves data items utilizing three separate databases. The first database, the image collection database, typically contains raw images for the purpose of visual display. The visual features extracted in order to support content-based image retrieval are stored in a second repository, while the text annotations or descriptions are stored in the third repository (Li & Kuo, 2002; Rui et al., 1999). This architecture also accentuate the importance of text-based image retrieval, residing on the notion that only the integration of content-based and text-based can result in satisfactory retrieval performance (Rui et al., 1999).

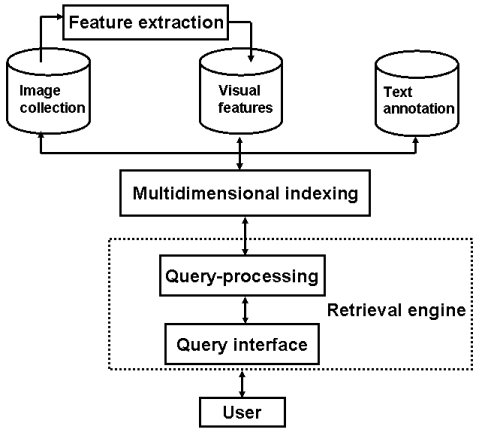


Figure 7 - An image retrieval system architecture, from Rui et al. (1999)

The image retrieval engine presented in figure 7 includes a retrieval engine consisting of some form of query interface and a query-processing unit. The query interface typically includes some form of presentation and manipulation functionality, whereas the query-processing unit serves as a means to translate user queries into an internal form, which is then submitted to

the DBS'. Multidimensional indexing is used to achieve fast retrieval and to ensure scalability (Li & Kuo, 2002).

2.4.1 Combining Image Queries with Text-based Queries

There exist several approaches to developing methods combining content-based queries with text-based queries in different domains. Within the medical domain, methods for content-based image searching have been used for ten years, and during this period suggestions of methods combining the two search approaches has also been proposed (Müller et al., 2005).

An early proposal for combining image content with associated text for retrieval purposes, was the *Image Indexing by Content network* (I^2Cnet), where additional data describing the selected images supported users in content-based queries (Orphanoudakis *et al.*, 1996). Queries were submitted as either *query by example* (QBE), or *query by sketch* (QBS). When forming queries served to the system, users participated directly in specifying the image description the system should use in the search.

Two more recent approaches, the first utilizing a standard cross-language information retrieval system combined with an image retrieval system (Jones *et al.*, 2004) and the second using the *medgift easyIR* system (Müller et al., 2005) originated from the CLEF 2004 conference[8]. Both approaches attempted to enrich content-based image retrieval with text, multi-lingual search terms. The former approach combined two result sets provided by two autonomous search systems, while the latter approach generated a text-based query expansion from the annotations accompanying the top three results provided by an initial image query.

Neither of the two newer approaches presented above provides possibilities for users to explicitly specify the text used in the query process.

In the non-medical domain, several approaches connecting visual and textual characteristics for retrieval purposes exist.

One such approach for retrieving images from the World Wide Web, started out with an initial text-based query specified by the user as the starting point for the image retrieval process, thus alleviating the zero page problem. Relying on the notion that images placed near text in an HTML document are related to the image, the retrieval system retrieve all images in near proximity of words specified in the initial query. Then the user performs several user-feedback cycles in order to refine the result set (Sclaroff *et al.*, 1999).

A slightly different approach is found in Zhou and Huang (2002). Here, an elaborate use of keywords in a semantic network, supported by online learning through a feedback algorithm and the use of a term similarity matrix, supports the content-based image retrieval process in order to draw upon the strength of both approaches. Users specify some keywords that accompany a seed image. The keywords given by the user supports the retrieval of images annotated with these words, or variations of the words provided by a thesaurus. By comparing images on their degree of syntactic similarity, the system can return results with a high score on both syntactic similarity and keyword similarity.

The former approach relies on automatic extraction of text found in close proximity of images. This is relatively straightforward in structured documents where <img /> tags most

[8] http://clef.iei.pi.cnr.it/2004/working_notes/WorkingNotes2004/CLEF2004WN%20-%20intro.pdf

often enclose the code specifying the image. The latter approach, though also having the possibilities of using automatic or semi automatic text extraction methods, may rely more heavily on manual annotation of image content. As discussed in section 2.3.5, annotating images manually is both time consuming and a potentially error prone task.

Westerveld (2000), presents an approach somewhat similar to that of Sclaroff et al (1999). The main difference in this new approach is that the author presents a method that combine text and images into the same semantic space using Latent Semantic Indexing. This is achieved by listing terms from both modalities in one term document matrix, before reducing the dimension of the matrix by a form of factor analysis, called Singular Value Decomposition (Westerveld, 2000). The text used in this approach, is image captions from a newspaper. The query process starts with an example document consisting of an image and its associated text, and the result set consists of the most similar documents with regard to both text and low-level features.

2.4.2 Retrieving Information Items Utilizing Context

User-evaluation of the relevance of an image is often dependent on context. This implies that the performance of a raking function may be very context dependent (Wang et al., 2003). A key component of a retrieval system drawing on this, is context awareness in the query process (P. J. Brown & Jones, 2001).

The term context-awareness, coined by Schilit et al. (1994), was introduced to help answer three important aspects of context: where you are, who you are with, and what desirable or useful recourses are nearby. The authors describe how context-aware software can adapt in order to aid in recording the data needed to answer the where, who, and what (Schilit et al., 1994).

An early approach to develop methods utilizing context in information retrieval was context-aware retrieval applications. These applications may be interactive, where the user directly issues a request to retrieve relevant data items, or proactive, where documents are presented to the user automatically (P. J. Brown & Jones, 2001).

Because of the exploding expansion in mobile computing in the early 1990's, an increasing interest in context-aware applications followed. The behaviour of these applications was, to some degree, governed by the current context surrounding the user (P. Brown *et al.*, 2000). The authors reviews six types of context-aware applications, typically including a mobile user whose context is changing, all relying on the same basic infrastructure for maintaining and manipulating contexts: capturing the current context, and containing a context memory.

To help define the field of context-aware applications, this thesis draws on a categorization for features of this type of applications, presented by Dey et a. (1999). This taxonomy shows a list of context-aware features that context-aware applications may support. There are three categories:

1. *presentation* of information and services to a user
2. automatic *execution* of a service
3. *tagging* of context to information for later retrieval

Even though much of the literature written on context-aware retrieval often sees it as a separate field of approaches to the process of retrieving information, Brown et al. (2001)

discuss how it also relates to the field of traditional information retrieval (IR), discussed in chapter 2.3.

Context-aware computing is at the heart of well functioning context aware retrieval applications, and since the end of the last century, there have been many attempts at defining context-aware computing. In this thesis we use the definition by Dey et al. (1999):

> **Definition 26 – *Context-aware computing*** *[is when a] system uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task* (Dey et al., 1999).

The *Fast Search & Transfer* (FAST) search engine (Øhrn, 2005) is one example of a system conducting context-aware computing, called contextual insight, for supporting text-based information retrieval.

In the FAST search engine, pre-processing operations offload data from participating databases in order to create one uniform text format through normalisation. Then the system forms standardized words through a process called "tokenizing", before extracting entities into vectors and categorize them using different thesauri. After the pre-processing stage, the system has aggregated data available for use in the retrieval process.

The FAST system relies on an XML-like structure, and context is defined as the structural aspects of a document. The context of a document as used in this system consist of the sentences and sections of a document, and users may then use context as criterion for where to search when searching for specific terms.

2.5 Retrieving Information Items from Single Database Systems

Retrieving data items from a single database system is relatively straightforward because of total compatibility between the internal structures of the database. The stored contents in the database and the vendor-specific functionality developed to aid users when operating on the database system.

A single database system may consist of one or more databases run by the DBMS, but as the database administrators and database programmers know the schema structure of the database (or databases) in advance, it is not problematic to implement and use multiple databases simultaneously within a single DBS.

As illustrated in figure 8, a database (or several databases), defined with specifications as to what contents to contain and what operations possible to perform on the contents, is the foundation for the information retrieval process.

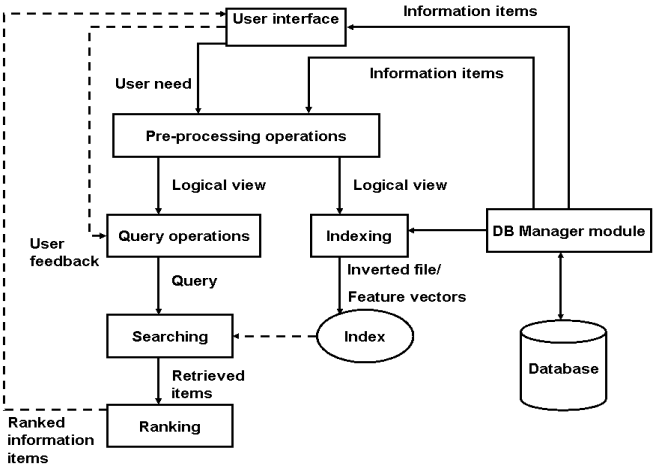


Figure 8 - IR from a single database, from Baeza-Yates and Riberiro-Neto (1999)

In systems built on the architecture presented in figure 8, the pre-processing operations analyze the original images and documents, and generate an image signature or document signature.

The ranking process in a database system acts on query results provided by the retrieval algorithms implemented in the system. Perhaps needless to say, these algorithms are an essential part of any information retrieval system, and several proposals for improving retrieval algorithms have been put forth. The process of ranking query results is discussed further in section 2.8.

2.6 Distributed Information Retrieval

Searching multiple database systems simultaneously and integrating the sub-results into one ranked result list according to a likelihood of relevance, is often referred to as distributed information retrieval:

> **Definition 27 – *Distributed information retrieval*** *is the use of multiple database systems, residing on one or more computers connected by a network to process a single information request.*

In increasing the number of database systems working together, the complexity of the information retrieval process increases accordingly. Two factors contributing to this increase in complexity are the fundamental issues of *heterogeneity* and *autonomy* of different systems (Bouguettaya *et al.*, 1999), and in order for the components of a distributed information system to function as a collective whole, the problems of heterogeneity and autonomy between the participating DBS's have to be resolved (Missier *et al.*, 1999).

2.6.1 Approaches to Distributed Information Retrieval

Information retrieval in distributed environments, where databases and database systems are heterogeneous, possibly autonomous, and often placed in different physical locations is a relatively well founded research field, dating at least back to 1980 (Breibart, 1990).

An early attempt at alleviating the problem of heterogeneity without using a global schema was presented in the *Multics Relational Data Store Multidatabase* (MRDSM) system, a prototype for multidatabase interoperability of relational database systems, proposed by Litwin and Abdellatif (1986). Here, various relational database systems could be integrated through describing the database definition of the relations being integrated in the MRDSM data definition language and the multidatabase data manipulation language available to the users of the system (Litwin & Abdellatif, 1986).

Others have described the tasks preformed by a distributed image retrieval system as being analogous to that of a meta-search engine (Losee & Church, 2004), where the meta-search engine uses other search engines to execute queries and provide query results.

According to Wu et al. (2003), this type of distributed information system typically involves the following major steps:

1. A broker, responsible for managing the query process, receives a user query from a client and selects a subset of resources that can best satisfy the user's information need according to the resource descriptions used by the broker (resource selection).
2. The broker sends the query to all the selected resources and collects the results from them.
3. The broker merges these results into a single list (results merging or data fusion).
4. The broker sends the merged result back to the client, which displays them to the user.

This is a commonly used solution known as a broker-based approach to distributed information retrieval (Missier *et al.*, 1999), illustrated in figure 9:

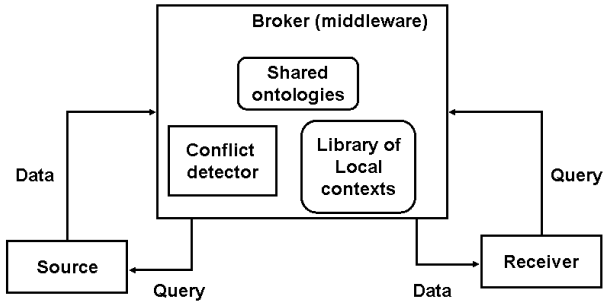


Figure 9 - Broker-based approach to distributed IR, from Missier et al. (1999)

As discussed above, using brokers in information retrieval is a common approach in methods used for developing search engines. These powerful resources are commonly used to help users find information on the World Wide Web in a transparent and efficient manner (Beigi *et al.*, 1998).

As search engines most often are developed to perform optimally within the environment they operate, they tend to perform very well in that environment. Meta-search engines draws from this, and consequently can make use of several local search engines to provide multiple sub-results to a given query, as is done by external meta-search engines (Montague & Aslam, 2001).

Meta-search engines may serve as common gateways, automatically linking users to multiple search engines (Beigi *et al.*, 1998). Figure 10 show the basic components of a meta-search engine.

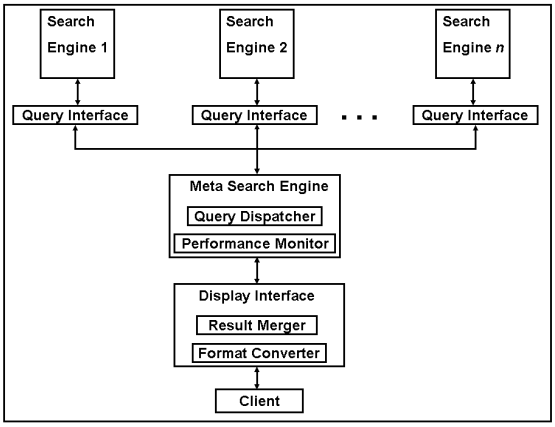


Figure 10 - Basic components of a meta-search engine, from Beigi et al. (1998)

This meta-search engine includes three basic components. The dispatching component selects target search engines for each query. The query interface component translates the user-specified query to compatible scripts to each target search engine. The display interface component merges the query results from each search engine, removes duplicates and displays them to the user in a uniform format (Beigi et al., 1998). Using the framework presented in Wu et al. (2003) on the model in figure 10, the meta-search engine and the display interface collectively forms the broker.

2.7 Challenges in Distributed Information Retrieval

A common task, and the main challenge of a distributed information retrieval system, is to search disparate sources or collections and present the data items that best satisfies the query submitted to the system (Losee & Church, 2004). These distributed environments change the way in which the system can evaluate the results from queries submitted to the different sources. This occurs because of the generation of multiple response sets that must be compiled into one ranked result list (Kretser et al., 1998).

In particular, the main challenges in distributed information retrieval are associated with merging and ranking of multiple query results according to their relative likelihood of relevance. The various sources may have differences in both the structure and implementation of their ranking algorithms, but as proprietary rights typically protect these algorithms, the vendors owning them usually keep them secret.

According to Steidinger (2000), the main challenges with merging and ranking query results in distributed information retrieval are:

- The sources use different ranking algorithms
- The ranking algorithms used by the sources are unknown
- The parameters used with these algorithms cannot be obtained from the sources

Since each different collection may use its own way to represent text and image content and use different methods for computing the similarity between images and documents, result lists returned by different sources may not be directly comparable.

Furthermore, even if having access to matching similarity scores, they are not normalized across collections. This prevents the possibility to evaluate the relative relevance of data items returned by different sources from the analysis of their matching scores. This comparison is necessary in order to present results to the user according to their relative relevance (Berretti *et al.*, 2004b). This issue is discussed further in the next sections.

2.8 Ranking Query Results

The function of ranking is to order data items according to the documents estimated match with a user query. This process relies on the ability to transform user queries into a form that can be effectively processes by computers. One of the most successful models is the so-called Vector Space Model (Fan *et al.*, 2004b). In fact, most of the existing search engines are built on the Vector Space Model, thus using a fixed ranking function for all contexts. This may pose serious problems when the same ranking functions are used to satisfy a wide range of user information needs (Fan *et al.*, 2004a).

According to Wang (2003), a ranking function consists of three parts: variables, constants and operations (which connect the first two parts). There are two types of variables, scalar and vector.

Traditional fusion techniques in information retrieval can be broadly divided into rank-based methods and score-based methods. Rank-based methods combine separate search results based on summing the rank position of documents from different result lists, while score-based methods typically: i) sums the multiple retrieval scores or ii) sums the scores from truncated result lists and multiplies the average by the number of retrieval models that returned it (McDonald & Smeaton, 2005). In addition to the rank-based and score-based approaches to merging and ranking distributed query results, the ranking can be probability-based.

Thus, nearly all existing ranking functions are static ranking functions, manually designed, based on experience, heuristic or probability theory (Wang et al., 2003). These methods use different mathematical strategies to rank occurrences into the final list, and have all different strengths and weaknesses depending on search criteria and the content queried (McDonald & Smeaton, 2005).

Recall from section 2.7, that algorithms for comparing similarity in images are not public knowledge, but as discussed in section 2.3.3, the common solution for content-based queries is also rank-based or score-based, utilizing vectors and similarity scores based on distance measures.

Two common distance measures involved in algorithms for measuring similarity between images, are the Euclidean distance (green line) and Manhattan distance (red, blue, and yellow lines), illustrated in figure 11. Euclidean distance is the distance between objects or values computed as a straight line. The Manhattan distance, also known as the L_1-distance, is the distance between two points in a Euclidean space with fixed Cartesian coordinate system as the sum of the lengths of the projections of the line segment between the points onto the coordinate axes.

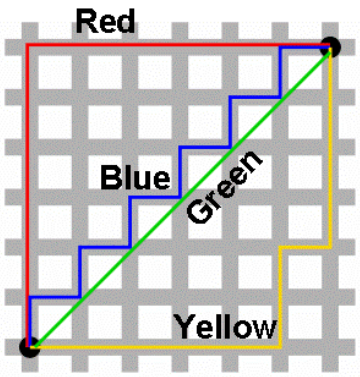


Figure 11 - Manhattan versus Euclidean distance

In figure 11, the red, blue, and yellow lines all have the same length (12), whereas the green line has the length $\sqrt{72} \approx 8.4853$. The consequence of this is that algorithms using the two approaches would generate different similarity scores for the same images.

Thus, a DBMS use distance measures to assign a similarity score to each image in the collection telling how much the image resemblances a seed image, evaluated by criteria determined by the system, the user, or a combination of the two. By retrieving images using this approach, the system can present hits only partially in accordance with the search criteria given, useful when relevance cannot be reduced to the dichotomy relevant/non-relevant (Losee & Church, 2004).

Various information retrieval studies have shown that the performance of a ranking function is very context-dependent. The context may depend on text collections or even properties of queries, and using a static ranking function cannot guarantee good performance under all situations (Wang et al., 2003). It is both time consuming and expensive to develop ranking methods that function optimally in all situations and contexts (Fan *et al.*, 2004b). In an effort to alleviate this problem, Wang et al. (2003) and Fan et. al. (Fan *et al.*, 2004a) propose a ranking approach that chooses the most effective ranking method depending on the query context.

2.8.1 Approaches to Merging and Ranking Multiple Query Results

As briefly discussed in the previous sections, the paramount objective when operating with multiple independent query results in a distributed system, is to compile these into one coherent list. This should also be done in a way ensuring that the effectiveness of the participating DBS is as high as if all documents came from one single collection (Voorhees *et al.*, 1994).

Most of the approaches proposed for merging results apply primarily to text libraries, and the extension to deal with multimedia libraries is not necessarily straightforward. First and foremost, this difference is due to the relative increase in computational complexity when comparing images for similarity (Berretti et al., 2004b).

Three commonly used approaches to the process of merging multiple query result sets are:

- Interleaving
- Normalization
- Weighted merging

Interleaving
The simplest approach to the process of merging multiple query results is simply to interleave them:

> **Definition 28 – *Interleaving*** *is the placement of returned query results in a presentation list in a notionally sequential manner, always selecting the next top item from each returned list of query results.*

If document rankings are available, the results from each collection can be interleaved in a Round Robin fashion (Callan *et al.*, 1995; Voorhees et al., 1994). Thus, result merging is accomplished by always picking up data items from the top of the returned lists of query results until all results are merged. In this approach it is assumed that each collection contains approximately the same number of relevant data items that are equally distributed on the top of the result lists provided by each collection (Berretti et al., 2004b).

Models using this approach, often referred to as Round Robin models, uniform models or interleaving models, removes the first element in the result lists in a Round Robin manner and puts them into a new list (Steidinger, 2000). Round Robin Random and Round Robin Block, two extensions of the simple Round Robin model, are slightly more sophisticated, but they both depend on knowing the length of the different result lists in order to work.

Normalization
The second common approach to merging query results is to utilize various normalization techniques in order to obtain precisely the same results that would be obtained if the individual document collections were merged into a single unified collection:

Definition 29 – *Normalization* *is a mathematical process that adjusts for differences among data from varying sources in order to create a common basis for comparison.*

Solutions developed on the idea of score normalization assume that each library returns a list of documents with matching scores. In this case, some technique is used to normalize matching scores provided by different libraries (Berretti *et al.*, 2004a).

Callan et al. (1995), present a proposal for normalizing results across document collections, where the participating sources through a pre-processing step provides the system with statistics about how many documents each query term or proximity operator matches (Callan et al., 1995).

Beretti et al. (2004b) propose a solution that develops on the solution presented in Callan et al. (1995), in developing a method for merging results from distributed content-based image retrieval. This approach is also based on statistics for performing the normalization, but decompose the fusion process into two separate steps: *model learning* and *normalization* (Berretti et al., 2004b).

Weighted merging
The third approach to the process of merging multiple query results is by using weights to help determine the value of a given list of query results.

> **Definition 30 – *Weighted merging*** *is an uneven interleaving, biased by the expected relevance of the collection to the query.*

Here, the merging and ranking process utilizes weighted scores. The weighs used in the process can be based on the score of a document and/or the score given to the collection where the data items come from. This solution is computationally simple in that participating collections does not need to provide query statistics, yet allegedly just as effective as normalized statistics merge (Callan et al., 1995).

This approach can be seen as an extension of a traditional Merge Sort model, which simply takes the elements into the global result list according to their DBMS-assigned scores (Steidinger, 2000). A prerequisite is that similarity scores are available for each data item. In addition, in order for this approach to work, the scores have to be normalized so that they are comparable.

2.9 Context Aware Image Ranking – A Combined Score Approach

As the discussion throughout this chapter has shown, there is a scarcity in methods usable for retrieving images from multiple DBS' simultaneously. In addition, several of the existing solutions for distributed image retrieval are using similarity criteria only. This may lead to query results containing images bearing close low-level syntactical resemblance to the search criterion. As users often may be interested in using contextual relevance as criteria, the similarity criteria alone may not always satisfy the actual information need.

A possible fruitful alternative is to utilize descriptions of image context actively when retrieving images. This could be achieved through using text-based information retrieval in combination with content-based image retrieval. This may possibly lead to query results bearing close syntactical resemblance as well as being relevant with regard to context.

The approach suggested in this thesis, is thus to use both similarity and context as query criteria in order to hopefully achieve a more precise ranking. To achieve this, a combined score approach using similarity scores from both content-based and text-based queries in order to produce a new score is proposed. This approach is discussed in the next chapter.

3 The CAIRANK Prototype

Resulting from some of the issues and challenges discussed in the previous chapter, an alternative approach to ranking of query results, represented by the Context Aware Image Ranking (CAIRANK) prototype, is proposed. This approach draws from and builds on some of the approaches discussed in chapter 2. The CAIRANK approach focuses on image retrieval from distributed databases using data items consisting of images accompanied by full-text descriptions. These text documents describe the depicted image context.

The CAIRANK approach as suggested in this thesis, made use of each participating DBS' implemented solutions for content-based image retrieval from the image collection and text-based queries against context information in order to create improved grounds for relevance-based ranking in a combined score approach.

The CAIRANK framework thus built on meta-search engine technology, and left the actual retrieval operations to the participating DBS' internal search engines. All participating DBS' utilized both content-based image retrieval as well as full-text queries on the context descriptions when generating the local query results. This solution should return images that were similar to the syntactical low-level features of the seed image in addition to being relevant from a context standpoint. The novel approach taken in this work, was utilizing context information in merging and ranking images already deemed relevant by the different DBS' queried. Hence, there was no conflict in applying CAIRANK in combination with some of the other methods discussed in chapter 2, which also combine text-based image retrieval with content-based image retrieval. These methods could be applied as an initial filtering mechanism before utilizing context information as a final ranking criterion on the results obtained by these other methods. In this thesis however, only the CAIRANK prototype was evaluated.

Users submitted queries to the CAIRANK system using both an example image and text input as search criteria. This solution enabled users to specify which kinds of contexts within an area are relevant in satisfying a given information need.

To help ensure that the final ranking of the distributed query results are correct regarding their relative relevance, the similarity scores in sub-results from each participating DBS were normalized locally with the help of implemented user defined functions. Then the results were processed further and merged using a weighted merging approach. The weights assigned to the each participating DBS' were determined using results from a test session where a standard query set for all participating databases was used.

A Key aspect of the proposed CAIRANK approach was to be compatible with most existing DBS' without being too intrusive. Hence, the approach would not be requiring query statistics or information on database structures.

The merging and ranking approach taken here, was to calculate a new score by combining the similarity scores returned from content-based and text-based queries in each of the participating DBS' and multiply them with the weight assigned to each DBS, before merging, ranking and presenting the results, as illustrated in Figure 12.

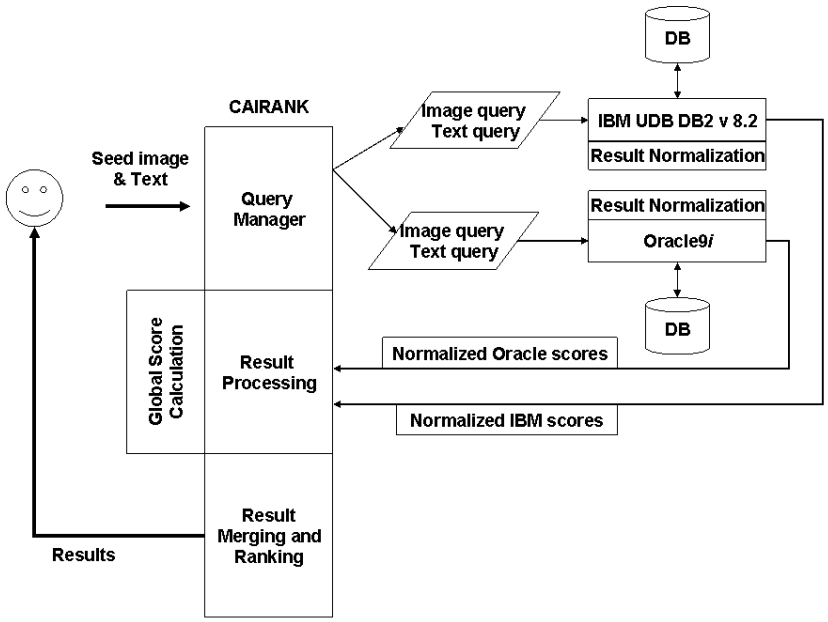


Figure 12 - The CAIRANK components

In order to evaluate this alternative framework, a system prototype capable of communicating with multiple DBS', submitting queries, and retrieving and ranking results based on using the combined score approach was developed. This version of the CAIRANK prototype, only served as a tool for evaluating any effect of merging and ranking of multiple results using the proposed framework. Thus, the implemented version of CAIRANK used in this project only included the minimum functionality needed to evaluate the hypotheses and research question.

3.1 Planning and Designing the CAIRANK Environment

The intended use of the CAIRANK framework when fully functional, was to provide users with a tool for obtaining ranked results from multiple existing image retrieval systems, all using a combination of text and image retrieval algorithms. To evaluate if this retrieval approach could aid in better ranking multiple result sets according to actual relevance with regards to context, a prototype (CAIRANK) to be used in an experimental setting, was developed from scratch. This helped ensure better control over the test environment.

The CAIRANK system components to be used in this project included three major components:

1. A basic meta-search engine able to communicate with several DBS' and having functionality to retrieve and process results, and calculate a new score implemented in the meta-search engine.
2. Two different DBS' with image and text collections.
3. Application for merging and ranking the results

In order to calculate a new score for items in the different result sets, the meta-search engine retrieved normalized similarity scores from the participating DBS' for further processing. The meta-search engine calculated the new score based partly on the similarity scores provided by

each participating DBS, and partly on a pre-determined weight assigned to each database in a prior session. The process of determining database weights is described in section 3.1.5.

3.1.1 Requirement Specification for the CAIRANK Prototype

The initial task when designing the CAIRANK prototype to be used in this project was to identify the required functionality of the main components constituting the system components. Table 1 displays the requirement specifications for the CAIRANK prototype. These are the minimum system requirements for a system generating data used to investigate the research question and hypothesis presented in this project. The specific system component used to solve the specific task is given under each requirement number.

Table 1 - Requirement specification for the CAIRANK components

Requirement no.	System requirements	Required functionality
1. CAIRANK Search Engine	Ability to receive queries from a user and submit them to several DBS'	• Communicate with participating DBS' using an appropriate database access interface.
2. DBS'	Ability to handle images and text Ability to handle relations between tables	• Have support for the CLOB and BLOB data types and being able to process and store text and image data types. • Have functionality to connect context descriptions to all corresponding images
3. DBS'	Management of Image collections with corresponding text objects	• Functionality to add images and text • Functionality to generate image signatures and text indices
4. DBS'	Basic CBIR algorithm	• Compare image signature of an example images to image signatures stored in the image collection • Have functionality to assign scores to query results
5. DBS'	Text query retrieval algorithm	• Determine similarity in documents, e.g. via tf/idf, based on query terms provided by the user • Have functionality to perform fuzzy or thematic full-text queries • Have functionality to assign similarity scores to query results
6. DBS'	Support for extending DBS' with User Defined Functions (UDF) and User Defined Types (UDT)	• Ability to create user defined procedures and functions in order perform image and text queries and store the results in a temporary □able • Normalize similarity scores locally in the DBS'
7. CAIRANK Search Engine	Ability to retrieve results returned to the temporary table by procedures in each participating DBS	• Use an appropriate database access interface and internal functionality to retrieve returned query results for further processing
8. CAIRANK Search Engine	Implemented functionality to process retrieved results	• Traverse the result sets and combine two similarity scores into one • Create a final score by multiplying each combined score with a weight assigned to the database returning the score
9. CAIRANK Search Engine	Deliver processed results	• Implemented functionality to write results to a list or table for the actual merging, ranking and further processing
10. Merging and Ranking Application	Merge and rank multiple result sets	• Implemented functionality to merge, rank and process multiple result sets using mathematical and statistical procedures

DBS' having functionality to meet the conditions listed in table 1, could execute queries submitted by a user to the CAIRANK prototype without having to make modifications to the internal structure of the systems database.

One possible implementation meeting these minimum requirements has been prototyped in the CAIRANK prototype. Here, functionality exists to communicate with multiple DBS', submit queries to several DBS', retrieve query results from participating DBS', generate a new score, and write the processed scores from participating DBS' to an external file for merging, ranking and further processing. The functionality of the CAIRANK system components is discussed further in the following sections.

3.1.2 Development Platform and Software

The functionality available in both the IBM and Oracle DBS satisfied the conditions specified in table 1, making them suitable tools for storing and querying the test collection. IBM UDB DB2 v 8.2.2 and Oracle9*i* were thus chosen as the primary development platforms for communication and participation with the search engine application in the CAIRANK prototype.

Both IBM DB2 and Oracle9*i* are object-relational database management systems with support for both SQL/3 and object-oriented software development through PL/SQL in Oracle9*i* and SQL/PL in IBM DB2. Furthermore, with the IBM imageExtender and NetSearch Extender (NSE), the Oracle interMedia and oracleText toolkits, both IBM and Oracle offer basic functionality for managing Large Binary Objects (LOB's) like images and text. This functionality includes syntactic feature descriptors and similarity functions through the db2image function in IBM, and the OrdDoc, OrdImage and OrdImageSignature classes in Oracle9*i*. Indexing of text-LOB's is supported through the implemented CONTEXT index type in Oracle and the text index in IBM DB2.

Table 2 displays the available DBMS functionality possible to use in this project to solve specific tasks. The requirement numbers in the table below correspond to the requirement numbers specified in table 1.

Table 2 - Overview of utilized DBMS functionality

Requirement no.	DBMS functionality	
	IBM DB2	**Oracle9*i***
2.	BLOB CLOB	BLOB CLOB
2.	Junction tables	Junction tables
3.	DB2Image	OrdImage, OrdSignature
3.	NSE - Text index	CONTEXT – Text index
4.	NetSearch Extender	Oracle Text - OrdDoc
4.	SCORE/RANK	SCORE
5.	SQL/PL	PL/SQL

The reasons for choosing these particular methods and functions for processing, storing text and images, and executing the queries, depended partly on availability and partly on their adequate abilities to solve each of the DBS-specific requirements specified in table 1[9].

There are five main reasons for choosing DBS' from IBM and Oracle for this project. Firstly, these two companies are dominating vendors in the market, and their products are quite

[9] For a thorough comparison of the implemented functionality for handling images and text in IBM DB2 and Oracle9*i*, please see (Bergli, 2006).

versatile when working with both images and text. Secondly, both Oracle and IBM provide database management systems available for download for personal use. Thirdly, both DB2 and Oracle 9i also have support for user-defined types (UDT) and user-defined functions (UDF). Fourthly, both DBS' support content-based image retrieval, IBM's DB2 through use of an Audio Image and Video (AIV) extender, and Oracle through the InterMedia extender. Finally, both DBS' support text-based information retrieval. IBM, previously supported text-based queries through several different extenders, but these are now consolidated into the NetSearch Extender (NSE), the one chosen as a tool in this experiment. Oracle provides support for text-based queries through OracleText.

IBM UDB DB2 and Oracle9*i* have DBMS' with specific capabilities that do not include normalizing of the similarity scores from media retrieval. This functionality was needed in this project, and as both systems support UDF, each DBS was extended with procedures for normalizing similarity scores generated by the content-based and text-based queries.

Enabling the CAIRANK search engine to submit queries to and retrieve data from the DBS' required a programming language that was easy to integrate with existing database systems. The Java programming language was chosen as development language for creating the required functionality to communicate with the DBS'. An application program written in Java could easily be set up to communicate with different DBS' through use of a Java Database Connectivity (JDBC) driver, a database access interface, and both IBM DB2 and Oracle9*i* support use of the JDBC driver, thus allowing for communication between the databases and the external Java application.

3.1.3 Obtaining comparable similarity scores across collections

As discussed in the first chapter, some substantial obstacles when retrieving data items from different DBS' are related to dissimilar standards, different database structures, different text and image retrieval algorithms, and different methods for ranking results.

According to Lu (1999), an ideal method of merging multiple results, should calculate similarities between the seed image and all returned images using the same feature and distance measurements before ranking the results. However, this was not a feasible solution for this project as different DBS' participated, all using different retrieval algorithms and different feature and distance measures.

As discussed in chapter 2, different approaches to the process for obtaining comparable results from multiple collections exist. One method is to normalize all results across collections according to statistic information regarding the effectiveness of each participating DBS. A potential drawback to using this approach is that normalizing scores obtained from multiple DBS', may involve considerable communicational and computational cost if collections are distributed (Callan *et al.*, 1995). Normalizing multiple query results across collections also demands that the ranking method has information about how the local ranking algorithms work, or that the participating systems provide query statistics for use in a normalization process. These approaches assume freely available information, or close cooperation between the different DBS'.

The methods chosen for this project were to interfere as little as possible with the internal structures of the source DBS', and this limited the possibilities to use the abovementioned normalization method in the experimental setting. This project did not meet the conditions

needed for making statistical normalization across collections a viable solution for the result sets used in the experiment.

On the other hand, by not having any method for further processing of results returned from the DBS', ranking methods would have to make use of the original DBMS using a Round Robin ranking approach, or use the original similarity scores for calculating a new score with the CAIRANK prototype. This could perhaps favour one DBS over another because of different usage of distance or similarity measures, but since both the Round Robin and CAIRANK ranking methods make use of the same subsets, the effect would be constant and therefore affect both methods in the same manner. Recording and measuring the effect of combining text and image queries would therefore still be possible, even with interleaved results in the combined result set.

However, using a method that simply interleaves items from the different result sets into the final result list in a round robin fashion, has proven to be very ineffective, with severe losses in precision (Callan *et al.*, 1995). This negative effect would probably increase if more than two DBS' were participating. Thus, the pure interleaving approach was not a viable solution from a usability point of view if for instance 50 % of the results from one DBS were to appear before the best results from another DBS simply because they use different ways of expressing similarity scores. In addition, comparing a rank based method (Round Robin) with a score based method (CAIRANK) would make it difficult to assess any improvements by using the CAIRANK approach.

Consequently, the results returned by the participating DBS' needed some form of processing before the final merging and ranking. Based on the aforementioned need for a method with abilities to rank query results without support from DBMS-provided meta-information, but still avoiding some of the disadvantages associated with interleaving, the method chosen for this experiment was therefore an implementation of the *weighted merging* method (Callan et al., 1995; Gauch *et al.*, 1996) discussed in chapter 2.

This merging method offers the computational simplicity of interleaving while avoiding some of its drawbacks (Callan et al., 1995). This approach use a combination of an initial training period with each DBS and registration of user activity when viewing the ranked result list, thus monitoring the distance between calculated similarity versus user-determined relevance (Gauch *et al.*, 1996). This initial training session is described in section 3.1.5.

A key aspect in the CAIRANK system environment was to have the participating DBS' provide as much help as possible in supplying data items of good quality for the two ranking methods considered in this thesis. However, it was also a goal to achieve this without having to provide extensive statistics of query performance or additional information about the internal structures of each database. In order to achieve this, both the IBM and the Oracle DBS was extended with user defined functions that processed the scores generated by the retrieval algorithms at query time by normalizing the query results from both the content-based and text-based queries.

The similarity scores from both the content-based and text-based queries were normalized using a *Min-Max Normalization* approach. In order to normalize the results, a temporary table in each DBS would be used to hold all results generated by both retrieval algorithms for each query. The highest and lowest similarity score calculated in each query by the image and text

retrieval algorithms could then be extracted to be used in the normalizing process as illustrated by the formulas in figure 13 and figure 14.

$$S'(i) = \frac{(S(i) - \min S(i))}{(\max S(i) - \min S(i))}$$

Figure 13 - Formula for normalizing text scores

In figure 13, S'(i) is the normalized score for each text-document in a given query (i), minS(i) and maxS(i) is the initial similarity score (S) range for each query (i). Here, for each query, the minimum score was subtracted from each score in the result, and this sum was divided by the product of subtracting the minimum score from the maximum score for the given query. This produced a new normalized score with a range going from "0" to "1".

In contrast to text-based queries, where the similarity scores most often run from "0" if evaluated to be irrelevant, to 1 or 100 if evaluated to be very relevant, many DBS' calculate similarity in content-based queries by measuring the distance between the signature of a seed image and the image signatures stored in the collection queried. The practical implications of this is that the image calculated to be most relevant will have a similarity score close to 0, while an image who's signature is evaluated as being less similar, is assigned a higher score.

Thus, combining the similarity scores for the retrieved images with the text-based similarity scores without modifying them, would in fact favour images calculated to be less similar to the seed image used as criterion. The similarity scores given to retrieved images were thus transformed and normalized using the altered formula illustrated in figure 14. This produced a normalized score of "0" for the image with the lowest similarity score, and increases towards "1" according to calculated similarity.

$$S'(i) = 1 - \frac{(S(i) - \min S(i))}{(\max S(i) - \min S(i))}$$

Figure 14 - Formula for normalizing image scores

3.1.4 Ranking Functionality

The CAIRANK approach needed functionality to rank the returned results by sorting data items according to similarity scores. In order to merge and rank the different result sets, the CAIRANK prototype was to be equipped with a modified variant of a traditional Raw Score merging method. As such, it operated in similar fashion by sorting elements according to similarity scores returned from the DBS'. However, the CAIRANK prototype would use two different similarity scores and further process these before merging them, as opposed to a traditional Raw Score method that ranks results using the image similarity score only.

As the purpose of the CAIRANK prototype was to help investigate effects of utilizing both the content-based similarity score and the text-based similarity score associated with each image in the result sets returned from each DBS, a formula for combining the scores was set up. This formula is illustrated in figure 15. In addition to processing all elements before merging and ranking them in the global result list, CAIRANK utilized additional data

provided by the DBS'. This information was in the form of a similarity score for the context information associated with each returned image. By using two scores instead of one, CAIRANK calculated a combined score for each data item returned using the formula presented in figure 15.

$$Score_{total} = \alpha * Score_{img} + \beta * Score_{txt}$$

α and β are the relative weights used for determining how much each score shall count in the global score.

Figure 15 - Formula for combining two normalized scores

By using relative weights for the returned similarity scores, different users could determine if the similarity score assigned to either image or text should count more than the other when calculating the total score. For instance, expert users, or users with extensive domain-specific knowledge, could perhaps make better use of context information, while novice users, or browsers, perhaps would place an equal weight, or even favour using images over context information.

In addition, if one of the DBS' systematically was to score its results higher than the other systems participating without actual differences in actual relevance, the weight put on each of the results could be reduced accordingly, thus creating more equal terms when producing the ranked list.

In this experiment, the similarity scores from both image and text queries were initially given equal weights of 0.5. These weights was only adjusted if one of the DBS' systematically scored its results higher than the other without actually supplying more relevant data items. This potential deviation was determined at the same time as the pre-defined database weights mentioned above was calculated and the results can be found on the enclosed media CD. These database weights was a measure on the expected quality of the results returned form each DBS. The process of determining database weights is discussed in section 3.1.5.

The merging of the elements in the result lists into the global result list with the CAIRANK prototype was done using the results produced by the formula presented in figure 16.

$$GlobalScore = Score_{total} * DatabaseWeight$$

Figure 16 - Calculating global score

As implemented UDF's in all participating DBS' transformed and normalized the scores given to the data items in the local result lists (using the formulas presented in figure 13 and figure 14), and as the CAIRANK search engine further processed the returned results, they became uniform across collections and could more easily be merged.

In order to merge and rank the processed results from the CAIRANK search engine, an application able to compile multiple results into a simple list or table would support this task. As the experiment set up in this project was to generate data to be used in assessing raking performance, abilities to process results using both mathematical and statistical methods was needed.

Microsoft Excel inhabits qualities to solve both these tasks with ready to use functionality suitable for this project. Microsoft Excel was thus chosen as the application to be used for both the merging and ranking of the results retrieved from the participating DBS' as well as the statistical processing of the experimental results regarding the performance of the ranking methods considered in this project.

3.1.5 Determining Database Weights

As briefly mentioned earlier, in order to differentiate between results provided by the participating DBS's based on the expected quality of the results, database weights was calculated in a separate query session initiated prior to submitting queries to the DBS'. Results obtained in this preliminary phase would determine the weight given to each of the participating DBS and used for calculating the global score as shown in figure 16.

In order to determine the relative database weights, results from a series of test queries would aid in assessing the effectiveness of each participating DBS by using the formula depicted in figure 17. This formula is presented in Yu and Meng (2003) where the authors describe the ProFusion formula originally introduced by (Gauch *et al.*, 1996).

$$c * \frac{\sum_{i=1}^{10} N_i}{10} * \frac{R}{10}$$

c is a constant applied to all databases. N_i is $1/i$ if the i-th ranked document is useful and 0 otherwise. R is the number of relevant documents in the first 10 retrieved documents for a given query.

Figure 17 - Formula for calibrating the effectiveness of a database

The weight reflecting the effectiveness of a DBS given a query is computed by capturing both the precision and rank order of retrieved documents in the test queries and multiply them with a constant (Yu & Meng, 2003) as shown in table 3. Using this approach opened for distinguishing between the influence of results retrieved from different DBS' thus possibly alleviate some of the potential problems with using normalized sub-scores from DBS' that used different retrieval and ranking algorithms. This was to be achieved by factoring the relative weight for the participating DBS into the global score.

Table 3 - Example of the process of determining database weights

Ibm Query 1			
608	SevernBridge4	1	1,0000
384	MillauBridgeFog3	0	0,0000
442	OldLisbonBridge25AprilBridge4	1	0,3333
214	GoldenGateConstruction6	1	0,2500
460	PascoKennewickBridge1	1	0,2000
430	NormandieBridge2	1	0,1667
686	TasmanBridgeConstruction2	0	0,0000
340	ManhattanBridge4	1	0,1250
46	AstoriaBridge1	0	0,0000
712	TingKauBridgeConstruction6	0	0,0000
	Sum:		2,0750
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**4,2330**

Oracle Query 1			
785	YangtzeRiverBridgeConstruction2	1	1,0000
515	Rama7BridgeConstruction9	1	0,5000
191	GeorgeWashingtonBridgeConstruction10	1	0,3333
257	HakuchoBridge3	1	0,2500
213	GoldenGateConstruction5	1	0,2000
271	HawthorneBridgeFog5	0	0,0000
109	BrooklynBridgeConstruction2	1	0,1429
715	TingKauBridgeFog3	1	0,1250
297	KapShuiMunBridgeConstruction3	0	0,0000
705	TingKauBridge3	0	0,0000
	Sum:		2,5512
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**6,0718**

Appendix F contains the results from the process of determining weights for the DBS' participating in this project.

3.2 Implementing the CAIRANK System

The CAIRANK system prototype was implemented as a Java application that used JDBC to communicate with two separate DBS', set up with an object/relational or relational database structure. The structure for the image collection and the associated context information were created as PL/SQL objects and tables in the Oracle9i database and as relational tables in the IBM DB2 database, as depicted in figure 20 (Section 3.2.2). SQL/PL objects were omitted in DB2, as relational tables proved to be sufficient for this project.

3.2.1 Java Application

As the system environment created in this project included a ranking method acting on data from two participating DBS', an application able to communicate with both systems simultaneously was required. Thus, to collect data for comparison and presentation, a simple interface and a basic search engine was developed. This interface and search engine only acted as tools for data collection, and as such, was not evaluated with regards to effectiveness or usability.

The Java application developed in this project consisted of four major classes. Figure 18 contains a listing of core variables and important methods. The structure and functionality of these classes builds partly on a prototype presented in Rønnevik (2005), and is summarized below. Please see Appendix C for a complete overview of the source code.

StartPage	**Interface**	**DbConnection**	**Queryproc**
JLabel Heading JButton CAIRANKquery JButton Quit JPanel page Font font Interface show_interface	JFrame JPanel JPanel pageCenter JPanel JPanel …	Connection connection	int SQL_Stmt0 String SQL_Stmt1 DbConnection dbConnection Interface myInterface
actionPerformed() StartPage() init_Menu() main ()	actionPerformed () Interface() init() addInformationItem()	connect() IBMdisConnect() ORACLEdisConnect() getIbmCon() getOracleCon() IbmisConnected() OracleisConnected()	DbConnection.connect() getResults() addInformationItem() close()

Figure 18 - Overview of classes and methods in the CAIRANK application

The choice of using this class structure for the Java application rested partly on the framework used as a starting point, and partly on simplicity. Here, the *StartPage* class contains methods for displaying the opening page where the user may enter the Java program. Upon entering the program, the user is presented with the query screen by methods in the *Interface* class, simultaneously, methods in the *DbConnection* class establish contact with the participating DBS'. Methods in the *Queryproc* class may then connect to the DBS', execute stored procedures, and retrieve the results. This query execution is illustrated in figure 19. Having this structure makes the development process relatively straightforward if implementing new functionality, adding more databases or updating different parts of the application.

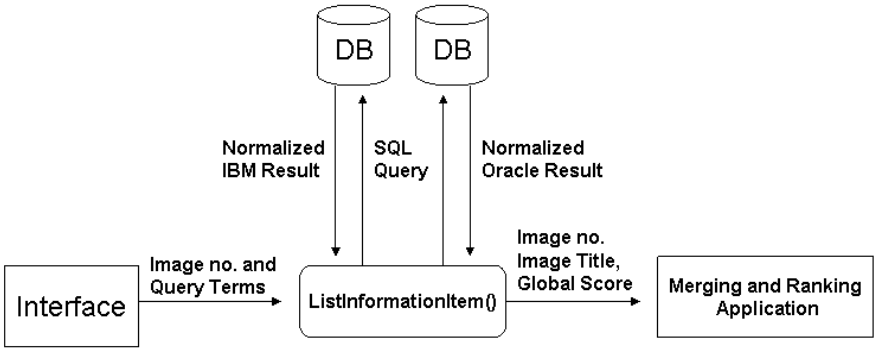


Figure 19 - Executing a query using the Java program

As illustrated in figure 19, the Java application had functionality to receive input to be used for queries from a user, connect to the participating DBS', and use the input from the user in procedure calls submitted to the DBS'. Then the *ListInformationItem()* method in the search engine could retrieve the results produced from the procedure calls, process results and write results to a list for further processing according to their relative relevance after being processed.

3.2.2 Database collections

The databases in the participating DBS's were kept as simple as possible while still maintaining the functionality required to work together with the CAIRANK application. Below, some code examples are used to illustrate central system components located in the DBS'. Please see Appendix B for a complete overview of the database code used in this project.

For convenience, both databases were modelled almost identically using the Structure Semantic Model (SSM) described in Nordbotten (2006), suitable for this project. The database structures were quite basic and contained only what was needed to perform the operations necessary for investigating the research question presented in this thesis. The database models are depicted in figure 20.

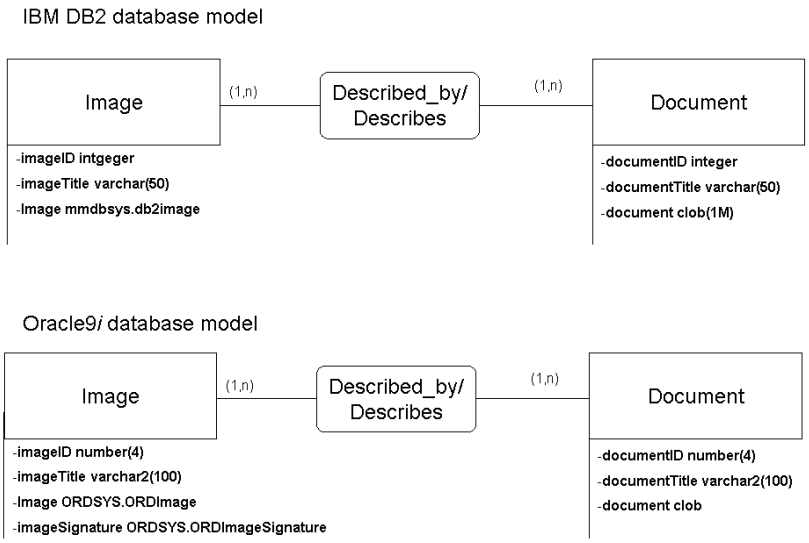


Figure 20 - Models of the databases used in this project

Both databases consisted of a simple database structure. The Oracle database consisted of an object-relational structure, while the IBM database consisted of a traditional relational structure, shown as the Image and Document entities in figure 20. The feature descriptors and the actual images were generated using the OrdImage and ORDImageSignature functionality in Oracle9*i* and db2image functionality in IBM DB2.

3.2.3 Building the Image Collection and the Text Descriptions

Populating the image and text tables in the Oracle database was done manually using the procedures shown in figure 21 and figure 22.

```
CREATE OR REPLACE PROCEDURE insert_image(Imgid number,Imgtitle varchar2,Imgfilename varchar2)
IS
Image ORDSYS.ORDImage;
Image_sig ORDSYS.ORDImageSignature;
ctx RAW(4000) := NULL;
BEGIN
INSERT INTO image_tab values(Imgid, Imgtitle, ORDSYS.ORDImage.init(), ORDSYS.ORDImageSignature.init());
    SELECT s.image, s.imageSignature INTO Image, Image_sig FROM image_tab s
        WHERE s.imageID = Imgid for UPDATE;
  Image.setSource('file','C_BILDEDIR',Imgfilename);
  Image.import(ctx);
  Image_sig.GenerateSignature(Image);
  Image.setProperties;
 UPDATE image_tab s SET s.image = Image, s.imageSignature = Image_sig WHERE s.imageID = ImgID;
 COMMIT;
END;
```

Figure 21 - Inserting images into Oracle

The procedure above imports an image from a pre-determined image directory, generates an image signature from the image using the OrdImage and ORDImageSignature, and stores this unique image signature together with the image identifier and the image name in the image table.

```
CREATE OR REPLACE PROCEDURE insert_document(Imgid number,Docid number,Doctitle varchar2, Docfilename varchar2)
IS
f_lob   bfile;
b_lob   clob;
BEGIN
INSERT INTO document_tab VALUES (Docid,Doctitle,empty_clob())
RETURN document into b_lob;
   f_lob := bfilename( 'C_DOCDIR', Docfilename );
   dbms_lob.fileopen(f_lob, dbms_lob.file_readonly);
   dbms_lob.loadfromfile( b_lob, f_lob, dbms_lob.getlength(f_lob) );
   dbms_lob.fileclose(f_lob);
INSERT INTO described_by_describes VALUES(Imgid,Docid);
   COMMIT;
END;
```

Figure 22 - Inserting text into Oracle

The procedure above imports a document from a pre-determined document directory and stores this document together with the document identifier and the document name in the document table.

Populating the image and text tables in the IBM DB2 database, also done manually, required a somewhat different approach than was the case with the Oracle database. Figure 23 and figure 24 show the code for image and text insertions into the IBM database. Although the code is quite dissimilar to the procedures used for populating tables in Oracle, the underlying principles are very much the same.

```
insert into image_tab values(<ID>,<TITLE>,<mmdbsys.db2image(current
server,'c:\DATABASECONTENT\Images\<FILENAME>','<EXTENTION>',1, '<NAME>'));
```

Figure 23 - Inserting images into IBM DB2

The procedure above also imports an image from an image directory, but in contrast to Oracle, this may be any folder on the system. However, in order to generate signatures for images stored in the collection, a QBIC folder must be created. The image table is monitored, and every time a new image is inserted, a signature is generated.

```
import from document_tab.del of del lobs from idocs\ modified by lobsinfi
le insert into DB2EXT.document_tab (documentID, documentTitle, document);
```

Figure 24 - Inserting text into IBM DB2

The procedure above also imports documents from a pre-determined document directory by using a list of documents to import in the form of a .del file. The import procedure extracts the documents that corresponds to names in the import list and stores them in the document table along with the document identifier and the document name.

The document collections containing the context descriptions were indexed using the default index functionality in DB2 and Oracle.

Links between images and the associated text descriptions were also created manually using a junction table. The actual image and text collections are discussed in chapter 4. The images

and text documents used in this project can be found on the enclosed CD-ROM. A compiled list of bridge names can be found in appendix D.

3.3 Image Retrieval Using the CAIRANK Prototype

Figure 25 displays the image retrieval process using the CAIRANK prototype. The process is described in detail below.

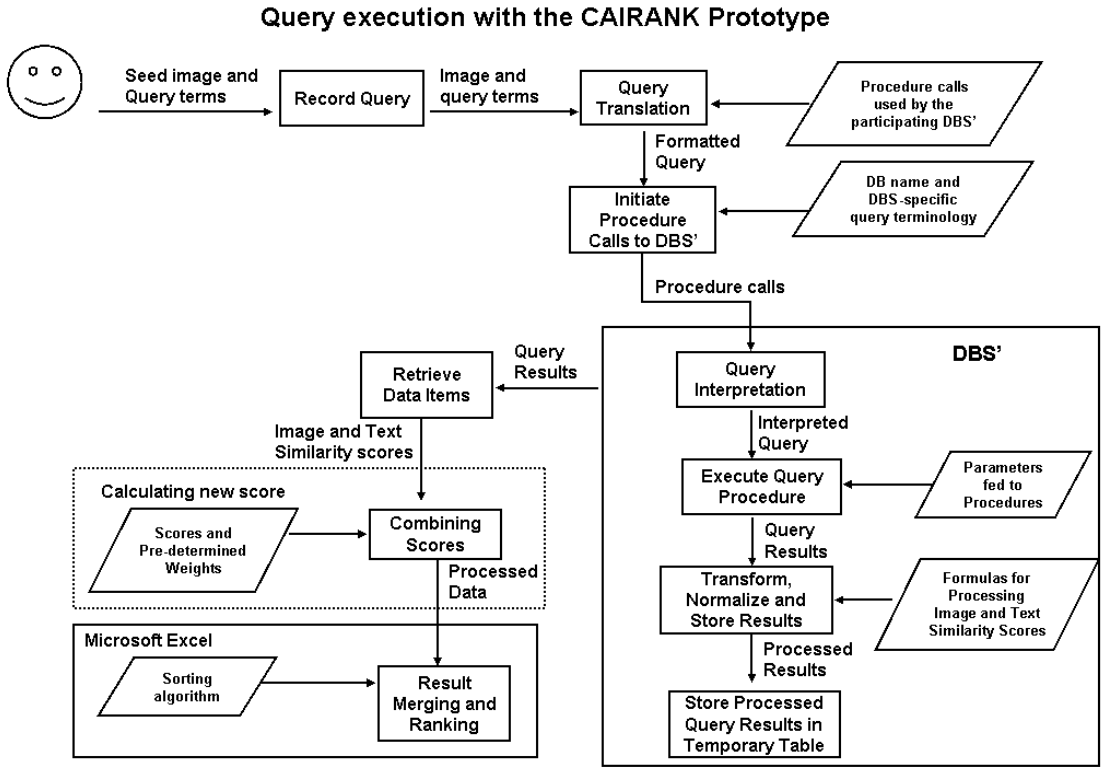


Figure 25 - Executing a query using the CAIRANK prototype

User interface

The *StartPage* class in the Java application contains the opening page of the CAIRANK application, a simple interface displaying "enter" and "exit" buttons.

Recording and Translating Queries

If the user chooses to enter the CAIRANK prototype, the *Interface* class is called on to present the implemented search options (See figure 18 page 41). Functionality implemented in the *Interface* class receives the image number and the keywords or phrases to be submitted as queries or procedure calls to the participating DBS' as illustrated in figure 26. Figure 27 illustrates the code for fulfilling these tasks.

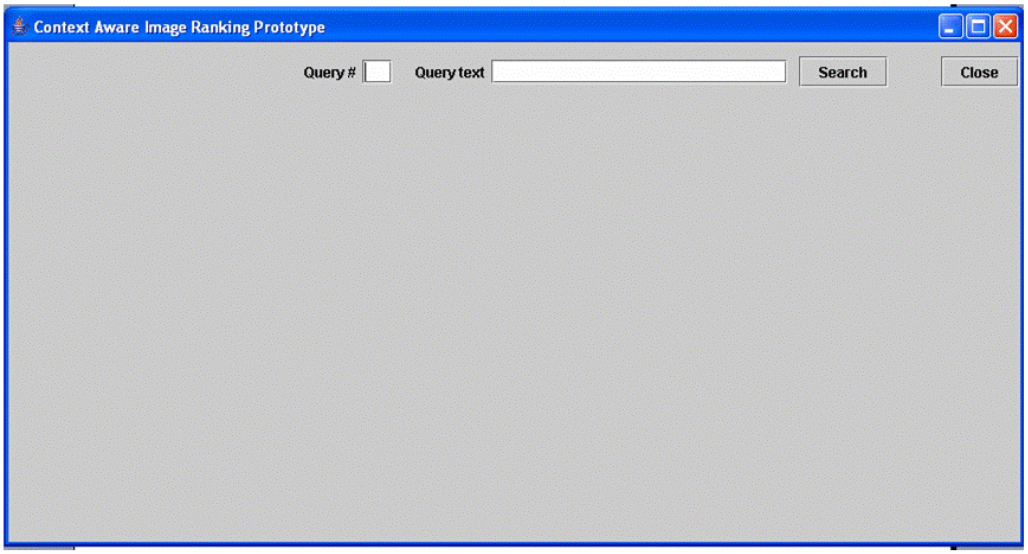


Figure 26 - CAIRANK Interface for submitting queries

In the first version of the CAIRANK prototype, queries were specified in the Java application using an image identifier and the query term to be used for the text-based query.

```
1. public void actionPerformed (ActionEvent event){
2. Object object = event.getSource();
3. if (object == btnCAIRANKSearch){
4. try{
5. imageNo = Integer.parseInt(textField0.getText());
6. }//end try
7. catch (NumberFormatException e) {
8.  System.out.println("You have not entered a number " +" "+ e);
9.              queryprocedure.close();
10. }//end catch
11. if (textField1.getText().length() == 0){
12.             System.out.println("You have not entered query text");
13.             queryprocedure.close();
14. }//end if
15. else if (textField0.getText().length() == 0 || textField1.getText().length() == 0){
16.             System.out.println("Not all field are filled in");
17.             queryprocedure.close();
18. }//end else if
19. else {
20. queryprocedure.getResults(imageNo, textField1.getText());
21. }//end else
22. } // end if
```

Figure 27 - Receiving query terms to be sent to getResults() method

Communicating with the DBS' and Initiating Queries

The *Queryproc* class calls on the *DBConnect* to connect to the participating DBS' and submit the queries provided by the user. The getResults() method in the *Queryproc* class performs the actual query execution by calling stored procedures located in the respective DBS'. These stored procedures produce and write the preliminary results into a result table (result_tab), and after executing all procedures, methods in the *Queryproc* class retrieves query results from both the content-based and text-based query from the result table as illustrated in figure 28. Here, the communication between the Java application and the IBM DBS is shown.

```
1. public void getResults(int textField0, String textField1) {
2.          SQL_Stmt0 = textField0;
3.          SQL_Stmt1 = textField1;
4.          try {
            […Code is truncated…]
5.          try{
6.          DbConnection.getIbmCon();
7.          String Iimgquery=Integer.toString(SQL_Stmt0);
8.          String idocquery = SQL_Stmt1;
9.          String imagecall= new String("");
10.         String doccall= new String("{ call docQuery(?) }");
11.         CallableStatement IMGstmt = DbConnection.getIbmCon().prepareCall(imagecall);
12.         IMGstmt.setString(1, (String) Iimgquery);
13.         CallableStatement DOCstmt = DbConnection.getIbmCon().prepareCall(doccall);
14.         DOCstmt.setString(1, (String) idocquery);
15.         IMGstmt.execute();
16.         IMGstmt.close();
17.         DOCstmt.execute();
18.         DOCstmt.close();
19.         }//end try
20.         catch(SQLException ex){
21.         System.err.println("SQLException: " +ex.getMessage());
22.         }//end catch
23.         try{
24.         DbConnection.getIbmCon();
25.         String iquery= "SELECT distinct r.image_no, r.sim_score, r.norm_image, r.img_title,
26.         r.doc_score, r.norm_text FROM result_tab r order by r.sim_score asc";
27.         PreparedStatement pstmt = DbConnection.getIbmCon().prepareStatement(iquery);
28.         ResultSet rs = (ResultSet)pstmt.executeQuery();

(…Continue below…)
```

Figure 28 - Calling procedures and submitting query to a DBS

3.3.1 Query Procedures in the DBS'

To generate the final query results to be retrieved by the Java application, each of the test queries submitted to the DBS' consisted of calling two separate query procedures where the first used the image number and the second used the query terms specified by the user in the CAIRANK interface. The first query procedure called from the Java application, consisted of a content-based query using a seed image corresponding to the image number given by the user as criteria. This query was submitted to each of the participating DBS', which in turn queried the stored image collection. Each image was given a similarity score that was inserted into a temporary result table.

The second query procedure called from the Java application used the keywords or phrases given as input to the CAIRANK interface along with the seed image. The stored procedure in each DBS queried the whole context collection and each context description was assigned a similarity score. For each context description with a corresponding image in the temporary result table, the similarity score from the text-search was inserted into the result table.

Figure 29 and figure 30 display the code for the CBIRQuery function as used in the Oracle9*i* and IBM DB2 systems respectively. Here, an example image is compared to other images in the collection through use of methods implemented in the DBMS. A similarity score is then calculated based on comparing the signature of the example image to the signatures of other images as described in section 2.3.3 (page 18). The actual similarity function used is the IMGSimilar from the OrdSys class.

```
1. CREATE OR REPLACE procedure CBIRQuery(queryseedimage number, threshold    number)is
[…Code is truncated…]

2. Cursor getphoto IS
3. select b.imageID,ORDSYS.IMGScore(123) score, b.imageTitle
4. from image_tab b
5. where ORDSYS.IMGSimilar(b.imageSignature,compare_sig,weighting,threshold,123) = 1
6. ORDER BY SCORE DESC;

7. CURSOR norm_imagescores IS
8. Select image_no, sim_score from result_tab
9. ORDER BY sim_score ASC;
10. BEGIN
11. select systimestamp into search_time from dual;
12. select p.seedimageSignature into compare_sig  from seedimage_tab p
13. where p.seedimageID = queryseedimage;
14. open getphoto;
15. loop
16. FETCH getphoto into imgnr,sim_score,title;
17. exit when getphoto%NOTFOUND;
18. select r.documentID, r.documentTitle into rdoc_no,docname from document_tab r,
19. described_by_describes
20. where document_describes = imgnr and image_described = documentID;
21. INSERT INTO result_tab
22. VALUES(search_time,imgnr,sim_score,title,rdoc_no,null,docname,null,null);
23. end loop;
24. close getphoto;
25. select max(sim_score) into max_score from result_tab;
26. select min(sim_score) into min_score from result_tab;
27. OPEN norm_imagescores;
28. LOOP
29. FETCH norm_imagescores INTO imgno,imgscore;
30. EXIT WHEN norm_imagescores%NOTFOUND;
31. norm_score := 1 - (imgscore-min_score)/(max_score-min_score);
32. UPDATE result_tab
33. SET norm_image = norm_score WHERE image_no = imgno;
34. end loop;
35. close norm_imagescores;
36. end CBIRQuery;
```

Figure 29 - The CBIRQuery procedure, illustrating an Oracle9*i* CBIR Search.

The CBIRQuery procedure shown in figure 29 has two input parameters. The first is the seed image identifier to the image to be used in the content-based query. The second parameter is a threshold determining the cut-off value for how similar images have to be in order to be returned. Choosing the ideal threshold value is a challenge, especially as the similarity comparison varies with the actual images involved, e.g. images taken during daytime versus images taken at night. Hence, using the threshold value to limit search results is not always a satisfactory solution and was therefore set to 100, i.e. the result set included all images in the collection.

The CBIRQuery procedure as used in Oracle9*i* combined a timestamp with the image identifier to identify the contents of result lists from different queries uniquely. A cursor was used to store the results of the image search. In figure 29, the *getphoto* cursor contained the image number, the similarity score and the image title. The results from the content-based query were inserted into the result table (result_tab). In order to normalize the results, the minimum and maximum values were retrieved from the result set.

A second cursor was used to retrieve parts of the results from the content-based query. In figure 29, the *norm_imagescores* cursor contained the image number and the image similarity

score. These results from the content-based query were normalized (line 31) according to the formula presented in figure 14 (page 38) and inserted back into the result table (result_tab).

Oracle9*i* also allows for using a set of weights determining the relative importance of the different low-level features in the image. The weights must add up to 100%. Unfortunately, after some testing of the procedure using different weight settings, some problems materialized when using the shape and location weights. The most acute problem was linked to the use of shape weights. The problem was that in the results from some of the test queries images very dissimilar to the seed images were given the image similarity score "0", i.e. was evaluated by the Oracle DBMS to be the same image. This anomaly did not occur if leaving out the shape weight. The algorithms used by the Oracle9*i* DBMS for similarity comparison are considered to be business secrets and are not public knowledge (Guros, 2004). Solving this problem was therefore not possible within the boundaries of this project. As IBM DB2 does not support shape and location weights, the shape weight was also omitted in the Oracle9*i* DBS.

Figure 30 illustrates the procedure code in IBM DB2. The differences between the IBM DB2 and Oracle9*i* procedures are related mainly to variations in the syntax used to create the procedures. In addition, the use of a timestamp was omitted from the IBM DB2 procedure as it was of little importance in this project.

```
1.CREATE procedure CBIRQuery(IN imagename VARCHAR(255))
[…Code is truncated…]

2. DECLARE norm_imagescores CURSOR FOR
3. Select image_no, sim_score from result_tab
4. ORDER BY doc_score ASC;
5. SET filename = '<server,C:\DATABASECONTENT\Images\'||imagename||'.bmp>';
6. SET stmt = 'SELECT a.imageID, decimal(mmdbsys.qbscorefromstr(' ||
7. 'bilde,''texture file=' || filename ||
8. ' and histogram file=' || filename || ' and averagecolor file=' || filename ||
9. ' and draw file=' || filename || '''), 10, 5)' ||
10. 'as score, a.imageTitle FROM image_tab a ORDER BY score DESC';
11. PREPARE selectStmt FROM stmt;
12. BEGIN
13. DECLARE getPhoto CURSOR FOR selectStmt;
14. OPEN getPhoto;
14. L1: LOOP
15. FETCH getphoto INTO imgnr,sim_score,title;
16. IF SQLSTATE = '02000' THEN LEAVE L1; END IF;
17. select r.documentID, r.documentTitle into rdoc_no,docname
18.  from db2ext.document_tab r, described_by_describes
19.   where document_describes = imgnr and image_described = documentID;
20.  INSERT INTO result_tab
21. VALUES(imgnr,sim_score,title,rdoc_no,null,docname,null,null);
22. END LOOP L1;
23. CLOSE getphoto;
24. END;
25. select max(sim_score) into max_score from result_tab;
26. select min(sim_score) into min_score from result_tab;
27. OPEN norm_imagescores;
28. L2: LOOP
29. FETCH norm_imagescores INTO imgno,imgscore;
30. IF SQLSTATE = '02000' THEN LEAVE L2; END IF;
31. SET norm_score = 1 - (imgscore-min_score)/(max_score-min_score);
32. UPDATE result_tab
33. SET norm_image = norm_score WHERE image_no = imgno;
34. end loop L2;
35. close norm_imagescores;
36. END
```

Figure 30 - The CBIRQuery procedure, illustrating an IBM DB2 CBIR Search

The structure of the CBIRQuery code in the IBM DB2 system is very similar to the code used for creating the same procedure in Oracle. The major differences lie in that DB2 does not support a threshold for cutting of the result and does not support the weighting of syntactical features. In addition, some differences exist in functionality for exiting a loop.

When testing the IBM DB2 procedure, a problem associated with the score interval became apparent. According to Cox (2001):

> *A score [in DB2] is a double-precision, floating point value between 0 and 1 that indicates how closely an image's features match those specified in the QBIC query. The lower the score, the closer is the match. A score of 0.0 indicates a perfect match.*

However, when testing the CBIRQuery procedure, the results spanned from “0.0” to “8.84627”. Stolze (2006), helped shed some light on the problem, and it appears as the actual program code of the IBM Image Extender has been altered, resulting in the removal the normalization function that sets the distance interval between “0.0” and “1”. No documentation on the reasons for this alteration exists, and consequently there is no certainty as to what the distance interval in DB2 actually is. This is not believed to have any practical implications on results in this project, as these were normalized using Min-Max normalization.

Text queries:
The query in the docQuery procedure, shown in figure 31 and figure 32 for Oracle and DB2 respectively, is directed at the document collection containing the context descriptions and makes use of the keywords or phrases given by the user. All documents are searched, and the scores of the documents associated with images inserted into the temporary result table (result_tab) by the CBIRQuery procedure, is inserted into the doc_score column. These scores are normalized (line 26) according to the formula presented in Figure 13 (page 38), before being inserted into the temporary result table.

```
1. CREATE OR REPLACE procedure docQuery(keyword1 in varchar2) is
[…Code is truncated…]
2. Cursor getdoc IS
3. SELECT SCORE(1),a.documentID
4. FROM document_tab a
5. WHERE CONTAINS(document, 'about({'||keyword1||'})', 1) >= 0
6. ORDER BY SCORE(1) DESC;

7. CURSOR norm_docscores IS
8. Select doc_no, doc_score from result_tab
9. ORDER BY doc_score ASC;
10. begin
11. open getdoc;
12. loop
13. FETCH getdoc into rel_score, docnr;
14. exit when getdoc%NOTFOUND;
15. UPDATE result_tab
16. SET doc_score = rel_score
17. WHERE doc_no=docnr;
18. end loop;
19. close getdoc;
20. select max(doc_score) into max_score from result_tab;
21. select min(doc_score) into min_score from result_tab;
22. OPEN norm_docscores;
23. LOOP
24. FETCH norm_docscores INTO docno,docscore;
25. EXIT WHEN norm_docscores%NOTFOUND;
26. norm_score :=(docscore-min_score)/(max_score-min_score);
27. UPDATE result_tab
28. SET norm_text = norm_score WHERE doc_no = docno;
29. end loop;
30. close norm_docscores;
31. end DOCQuery;
```

Figure 31 - The docQuery procedure, illustrating an Oracle9*i* text Search

```
1. CREATE procedure docQuery(IN keyword1 VARCHAR(255))
[…Code is truncated…]
2. DECLARE norm_docscores CURSOR FOR
3. Select doc_no, doc_score from result_tab
4. ORDER BY doc_score ASC;

5. SET stmt = 'SELECT a.documentID, SCORE(document,''' || 'IS ABOUT EN_US '''||keyword1||''''' || ''')
6. FROM DB2EXT.document_tab a';
7. PREPARE selectStmt FROM stmt;
8. BEGIN
9. DECLARE getDoc CURSOR FOR selectStmt;
10. OPEN getDoc;
11. L1: LOOP
12. FETCH getDoc INTO docnr,rel_score;
13. IF SQLSTATE = '02000' THEN LEAVE L1; END IF;
14. UPDATE result_tab
15. SET doc_score = rel_score
16. WHERE doc_no=docnr;
17. END LOOP L1;
18. CLOSE getDoc;
19. END;
20. select max(doc_score) into max_score from result_tab;
21. select min(doc_score) into min_score from result_tab;
22. OPEN norm_docscores;
23. L2: LOOP
24. FETCH norm_docscores INTO docno,docscore;
25. IF SQLSTATE = '02000' THEN LEAVE L2; END IF;
26. SET norm_score = (docscore-min_score)/(max_score-min_score);
27. UPDATE result_tab
28. SET norm_text = norm_score WHERE doc_no = docno;
29. end loop L2;
30. close norm_docscores;
31. END
```

Figure 32 - The docQuery procedure, illustrating an IBM DB2 text Search

3.3.2 Merging and ranking results

The getResults() method in the *Queryproc* class retrieves and combines the similarity scores from each DBS, adds the database weight assigned to each participating database, and writes the results to an Excel sheet for further processing.

The lines 38 and 39 in figure 33 illustrate how the CAIRANK prototype processed the results retrieved from the participating DBS' according to the formulas presented in figure 15 and figure 16 (page 39). In line 38, it is shown how the relative weight to be multiplied with the image similarity score is reduced because IBM DB2 systematically scored its results higher than the Oracle9*i* without actually supplying more relevant data items. See the enclosed media CD on how this relative weight was determined. In line 39, the weight assigned to the database is added to the combined score. Results from the actual process of calculating database weights for each participating DBS are shown in appendix F.

```
(…Continue from figure 28 page 47…)
29. while (rs.next()){
30.         int Iimagenumber =(rs.getInt(1));
31.         String Iimagescore =rs.getString(2);
32.         double Inorm_image = rs.getDouble(3)*100;
33.         int new_Inorm_image= (int)Inorm_image;
34.         String Iimagetitle =rs.getString(4).trim();
35.         String Idocscore =(rs.getString(5));
36.         double Inorm_text =rs.getDouble(6)*100;
37.         int new_Inorm_text= (int)Inorm_text;
38.         double IscoresToComb= ((0.468223394* new_Inorm_image)+(0.5 * new_Inorm_text)/2);
39.         double Iglobalscore= 5,7506* IscoresToComb;
40.         String ItotalCombscore = Double.toString(Iglobalscore );
41.         String Istring_score = ItotalCombscore.replace ( '.', ',');

(…Continue below…)
```

Figure 33 - Processing results from a DBS

As discussed in section 3.1.4, in the experiment, an Excel worksheet was used for the actual merging of the results from the two DBS' by sorting them according to the new score calculated by CAIRANK as shown in table 4. The table illustrates how results, here represented by the top-twenty CAIRANK results from query 1, were presented in an Excel sheet. Figure 34 display the actual Java code used.

Table 4 - Example of a result set written to Excel

Query No:1		
Image No:	Image Title:	CAIRANK Score:
353	Manhattan Bridge	4,70213803
323	MacDonald Bridge	4,696848842
587	San Francisco Bay Bridge	4,675068365
673	Tacoma Bridge	4,60708234
89	Bosphorus Bridge	4,570581953
675	Talmadge Bridge	4,362432446
585	San Francisco Bay Bridge	4,337583242
589	San Francisco Bay Bridge	4,333616351
674	Tacoma Bridge	4,298027854
235	Golden Gate Bridge	4,123021842
586	San Francisco Bay Bridge	3,982681948
335	Mackinac Bridge	3,97262927
488	Queen Isabella Causeway	3,903573151
490	Queen Isabella Causeway	3,877509118
236	Golden Gate Bridge	3,877095819
588	San Francisco Bay Bridge	3,8264593
234	Golden Gate Bridge	3,53770346
237	Golden Gate Bridge	3,45992732
232	Golden Gate Bridge	3,384146452
387	Mississippi Bridge	3,328296396

Excel worksheets were also used to determine the degree of precision, as well as for comparing of the results to an ideal result set in order to measure the distance between the ideal placement of the results and the actual placement of the returned results.

```
(…Continue from figure 33…)
42. try {
43.         BufferedWriter out = new BufferedWriter(new FileWriter("C:\\Documents and
44.         Settings\\Christian\\Mine dokumenter\\Mastergradsprosjekt\\Datainnsamling\\CAIRANKResults.xls",
45.         true));
46.         out.write(Iimagenumber+"\t"+Iimagetitle+ "\t"+Istring_score+"\n");
47.         out.close();
48.         BufferedWriter out2 = new BufferedWriter(new  FileWriter("C:\\Documents and
49.         Settings\\Christian\\Mine
50.         dokumenter\\Mastergradsprosjekt\\Datainnsamling\\RoundRobinResults.xls", true));
51.         out2.write(Iimagenumber+"\t"+Iimagetitle+ "\t"+new_Inorm_image+"\n");
52.         out2.close();
53.         }//end try
54.         catch (IOException e) {
55.                     System.out.println(e);
56.                     }//end catch
57.         } // end while
```

Figure 34 - Displaying results on screen and writing the results to an Excel sheet

4 Research Framework and Strategy

The principal motivation underlying this thesis was to investigate and measure any effect of merging the query result sets from multiple DBS' by combining the similarity scores obtained through queries on both image content and their associated text-based descriptions of contexts present in the image. The question initially raised was if such an approach could lead to significant improvements in the process of merging and ranking result sets from separate and autonomous sources as opposed to ranking results based on similarity scores alone.

Implemented functionality to calculate a combined score based on both similarity scores could help to make the CAIRANK prototype a useful tool in evaluating the effect of combining results produced by content-based and text-based retrieval algorithms in the ranking process. A basic Raw Score merging model could be used for comparison since the CAIRANK approach is an extension of this model. The Raw Score approach ranked query results based on their normalized similarity score alone.

In order to evaluate the ranking functionality implemented in the CAIRANK prototype, it has been tested in an experimental setting, and the performance has been compared to the performance of a traditional Raw Score merging method.

4.1 Experimental Design

4.1.1 Experiment Classification

This project used an experimental approach as a means to investigate a cause-effect relationship. The experiment conducted was set up in a laboratory setting, which helped to create an environment that ensured the necessary control over the different variables in the experiment.

A common purpose of laboratory experiments is to examine the effect of an experimental unit by investigating what effect a certain factor has on the researcher's attribute of interest. To be a scientific method, a control unit is required in order to be able to record results from absence of the factor. Both units use the same data generator, providing comparable results for further statistical analysis. Table 5 presents the different components used in this project and the components of an experimental framework to which they correspond.

Table 5 - Classification of the experiment

Experiment component	Component in this project
Experiment unit	Result ranking with the CAIRANK prototype
Control unit	Raw Score merging ranking using only similarity scores from CBIR
Factor	A combined score from both image content and context information for each image
Attribute of interest	Ranking results from image retrieval from distributed image databases
Data generator	Query set

The experiment reported this study served as a means for comparing two methods for ranking query results, applied on distributed queries.

The first of the two ranking methods considered in this experiment, was the implemented ranking functionality of the Context Aware Image Ranking (CAIRANK) prototype presented in the previous chapter, using similarity scores computed from both content-based queries and text-based queries when ranking results.

The second ranking method considered was an implementation of a traditional Raw Score merging approach, served as base for comparison. The Raw Score merging method carried out the ranking process based on content-based similarity scores alone, while the ranking method implemented in CAIRANK processed all similarity scores before they were made ready for presentation.

The CAIRANK prototype thus served as the experimental unit, using the implemented ranking method, whilst the traditional Raw Score merging model was the control unit. The effect of a combined score, combined from both content- and text-based similarity scores, was the factor of analysis. The attribute of interest was the calculated degree of relevance in the merged and ranked results from the distributed queries.

The function of the query set was to generate data aiding the assessment on how well the two ranking methods considered performed. These data then went through mathematical and statistical processing in order to evaluate the results obtained from the experiment.

4.2 Determining Relevance

The methods used to evaluate how well the two ranking approaches perform were based on determining the degree of relevance in the results. Determining relevance judgement sets is a very challenging task when creating a test collection.

The author chose both the image and context description collections, developed the text-based queries used to accompany the seed images in the experiment, as well as selecting the seed images used in the queries and determining the ideal result sets. Relevance in the query results obtained in this experiment was thus based on the subjective interpretation of the researcher.

The obvious alternative to producing the test queries alone would be to include other people in the process of developing the queries. Using people external to the project could help prevent bias in the process, thus reduce the danger of favouring the approach proposed here over the traditional approach. However, the process of recruiting participants to help construct the query set and submitting the actual queries can be both time and resource consuming, and therefore not used in this project.

Some of the possible consequences the choice of not including other people in these parts of the project may have had on the results recorded in the experiment are discussed in chapter 6.

4.3 Image and text collections

4.3.1 Image Collection

To perform the experiment, a set of about 800 images was collected from sites on the internet[10]. The collection contained images of 84 different bridges, mainly suspension bridges in various forms, but also some truss and cantilever bridges. Images in the collection

[10] The images used in the test collection came from various internet sources. Two of the major contributing sites were http://portlandbridges.com/ and http://en.structurae.de/structures/stype/index.cfm?ID=1001. Please see the enclosed CD-Rom for a list of all images.

consisted of both black and white and colour photos. Some images were old and of relative poor quality, while other images were of very good quality clearly taken using professional equipment.

The reasons for constructing an image collection exclusively for this particular project were twofold. Firstly, there was a lack of suitable image collections available when starting up this project. Secondly, by developing a specialized collection, maintaining control and having a clear overview over the images and ideal result sets was possible. Some possible implications of this concerning bias and limitations in generality, is discussed in chapter 6.

All 800 images were numbered and distributed equally between the DB2 and the Oracle database. Images with even numbers were stored in the DB2 database, while images with odd numbers were stored in the Oracle9*i* database. Each of the image collections thus consisted of 400 images of the same 84 bridges. By assigning images to the databases in this manner, the total collection was relatively evenly distributed with regards to the number of images of different bridges in the two databases.

Although containing images of the same bridges, the images stored in each collection were not identical, but containing different images of the same bridges. Each bridge was thus depicted in various circumstances or contexts, e.g. during construction, at night or at day. An example of this is illustrated in figure 35. Here, the Golden Gate Bridge is depicted in various contexts. Other bridges could be depicted in different circumstances.

Figure 35 - Example of variation on a bridge in the image collection

The first and third images from the left were stored in the DB2 image collection while the second and forth image were stored in the Oracle9*i* database.

The total collection was thus relatively small with regards to the number of different bridges, but at the same time relatively diverse by having several different images of most bridges, hence creating a good deal of variation in the image content as a whole.

Central prerequisites concerning syntactical features in all the images chosen for this project were that they had a fair amount of syntactical similarity to some of the other images in the collection, or that they had a semantic relationship to other images in the stored in the databases, e.g. was the same object depicted in different circumstances.

Bridges represent one example of structures displaying features like those mentioned above; often they have a high degree of syntactical resemblance but may not necessarily have any semantic relevance to a given query. Other times, they can display less syntactical resemblance but have greater relevance to a given query. Therefore, the reason for choosing bridges is not entirely coincidental. As bridges are relatively similar in appearance, this should increase the DBMS chance of finding relevant images, based on their syntactical properties.

In addition to a syntactical similarity, most of the images in the collection have a fair amount of semantic overlap in that they depicted variations of the same bridge, or that they portrayed examples of similar contextual contents like night or day, fog, construction, lightning etc.

4.3.2 Context Information Documents

As a supplement to the images kept in each of the databases, text-documents containing general full-text descriptions of different aspects of the image contents were also stored.

The site supplying the main portion of text-documents used as context descriptions in this project was the free encyclopaedia wikipedia[11]. This is a popular online encyclopaedia with a wide range of articles covering many topics. Most articles posted in wikipedia are the result of a collaborative effort from several authors. In addition, members with the responsibility of acting as moderators, regularly review the articles posted in order to help ensure articles of reasonable quality.

Wikipedia allows for corrections, additions and alterations of stored articles. If an article does not cover all aspects of a topic or lacks nuance, other authors can address the problem by simply extending or editing the article. The result is articles reflecting the views and words of several writers with knowledge of the subject.

Information taken from wikipedia thus provided context information formed by consensus, and the hope was that the information therefore was well suited to offer good contextual descriptions[12]. In addition, using just a few different external sources for describing context, created a uniform collection of context information across different DBS'.

There has been a debate about the quality of articles published by wikipedia, and thus questioning the quality wikipedia as an encyclopaedia[13, 14]. The quality of the information presented in wikipedia has not been evaluated in this thesis. As the content used for context information is both quite general and only serves as test data, no problems were anticipated in using source material from wikipedia. Developers of the *Fast Search & Transfer* (FAST) search engine, discussed in chapter 2, used contents gathered from wikipedia when testing their latest version of the search engine, and did not report any problems associated to the use of wikipedia for test purposes. For a full-scale system, other information sources with quality assurance of the contents should be considered.

Context information, as defined in this thesis, is not necessarily of a uniform format where all text-documents have exactly the same structure. Following from this is that different text-stubs, sections or segments, describes various aspects of an image context. In this sense, the collective sum of all the document sections containing context information together forms the description of context for each image.

However, as the definition of context is quite open, and in order to help ensure better control over the experiment some fundamental categories for describing context was adopted in this experiment. The context information associated with images used here, should therefore

[11] http://en.wikipedia.org/wiki/Main_Page
[12] See all context descriptions on the enclosed CD-ROM.
[13] http://www.freedom-to-tinker.com/?p=674
[14] http://many.corante.com/archives/2004/08/29/wikipedia_reputation_and_the_wemedia_project.php

describe the image content in terms of at least one of the primary categories for characterizing context for a particular entity presented by Dey et al. (1999) in chapter 2, repeated here:

- Activity
- Identity
- Location
- Time

Concepts chosen to represent the context categories in this thesis where selected manually by the author, and an effort were made to keep these quite clear and concise. The reason for choosing just a few concepts that would represent the different context categories was done in an effort to reduce ambiguity when evaluating the actual value of using context for relevance purposes.

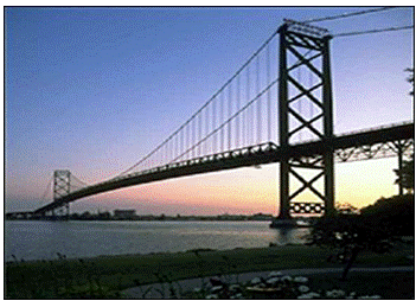
Activity
Activities used in the context descriptions for images in the collections was limited to either contexts clearly visible in the image, or well known facts associated with an entity. Examples of clearly visible information are construction work, collapse, or lightning. On the other hand, an example of a well-known fact that is not necessarily depicted is suicides. Bridges in general, and Golden Gate Bridge in particular, are used by people when committing suicide.

Identity
The actual bridge names were omitted for identification purposes, as this would favour the CAIRANK approach over the Raw Score method. Instead, general identifying concepts were used, like hollow box girder, or that the type of bridge desired is a railroad bridge.

Location
General specifications were used instead of for instance GPS positions. Here, location criteria were specified using for instance name of a continent of interest, the name of a country, or the name of different cities.

Time
Context criteria associated with the time category were also set up to be as little ambiguous as possible. The concepts of day, night, and low and high tide were used to represent this category in the queries.

One or more of these free-text descriptions thus formed the compiled context information for each image as illustrated in figure 36.



Ambassador Bridge
From Wikipedia, the free encyclopedia.
Jump to: navigation, search

Ambassador Bridge from the Canadian side of the Detroit River.The Ambassador Bridge is a privately-owned suspension bridge which connects Detroit, Michigan, in the United States, with Windsor, Ontario, in Canada. The Detroit-Windsor Tunnel also connects these two cities.

The bridge, over the Detroit River, had the largest central span in the world when it was completed in 1929 — 1,850 foot (564 m). The total bridge length is 7,500 feet (2,286 m). The roadway rises 152 feet (46 m) above the Detroit River. It is the busiest international border crossing in North America in terms of trade volume: more than 25% of all merchandise trade between the United States and Canada crosses the bridge.

It is a four-lane bridge and carries more than 10,000 commercial vehicles on a typical weekday. Jointly owned by the Detroit International Bridge Company and the Canadian Transit Company the bridge will have direct access to and from interstate highways (I-75 and I-96) on the U.S. side when a major redesign of its U.S. Plaza is completed in 2008.

Due to the extremely high traffic volume, the American and Canadian governments are jointly examining proposals for the construction of a second bridge downriver.

It was featured in the Eminem movie 8 Mile, and Crossing the Bridge, starring Stephen Baldwin.

Night
From Wikipedia, the free encyclopedia.
Jump to: navigation, search
This article describes the time of day. For the work by Elie Wiesel, see Night (book). For the Norse goddess see Nótt.

[Truncated...]

Figure 36 - Example of image and context information

In figure 36, Ambassador Bridge is depicted at night. The first information stub, which describes the characteristics of the bridge in general terms, represents the *Identity* context category and is used for all images depicting Ambassador Bridge. In this image, the only additional context information recorded for this particular image is information representing the *Time* category, represented by an information stub describing night. For other images, several information stubs containing context information representing the different context categories in describing the image content was appended in a similar manner.

An important aspect when utilizing context, is so ensure that the retrieval systems access to context information is as good as possible. By doing this, the interaction between machine and man may potentially improve the relevance of query results with regards to context (Dey et al., 1999). In this project, full-text descriptions were used to record context information.

Three main factors motivated the choice of using full-text context descriptions. Firstly, describing content in full-text using natural language using an external source could perhaps help to avoid some of the problems associated with annotations, discussed in chapter 2, as the same context descriptions could be associated with all images displaying the same object in similar circumstances. This could potentially remove some of the negative drawback of subjectivity in the descriptions.

Secondly, by using full-text descriptions from external sources, and then relating the descriptions to all relevant images, time could be saved compared to time used to annotate each and every image manually. For example, one description of the objective statistics of a

bridge could be used to describe all images of this bridge instead of having to annotate all images with exactly the same keywords.

Thirdly, having the text stored as documents written in natural language, allowed for usage of a wider range of query functionality in the DBMS, e.g. fuzzy queries, queries about words in near proximity, or queries about certain themes.

However, these assumptions have not been scientifically tested, and the experienced effects of using full-text descriptions in natural language are discussed in chapter 6.

4.4 Test Queries

For this project, the assumption was made that the DBS' was participating in a consortium and would probably configure their CBIR query procedures in an optimal manner. Hence, each DBS used in this project was set up to provide as good results as possible.

Concerning the image queries, these were configured to retrieve images based on combining features extracted from colour and texture. The reason underlying the choice of using a combination of these low-level features in the experiment was to attempt to create as equal conditions as possible when measuring the effect of the context information in merging and ranking multiple results.

In addition, at least in the case of Oracle's DBMS, it has been indicated that queries utilizing a combined set of features perform reasonably well, just outperformed by queries using only colour features as criteria (Vinsvold, 2005). However, the option of using colour features only was not well suited for this project as this could discriminate many images. Thus, all content-based queries was set up using a combination of colour and texture features for calculating similarity.

4.4.1 Selecting the Query Set

The information used to portray the specific contexts present in the images described them in general terms, focusing on relatively unambiguous types of contexts, this information also contributed in forming the base for the ideal response set. These ideal response sets contained the images that most likely would satisfy search criteria judged by human standard. The images in the ideal response set were the ones assumed to be chosen if a person instead of the DBMS' was to traverse the collection looking for images similar to a seed image accompanied by an information need expressed using text.

The query set was developed after the image collection was created, and the images used as seed images in the experiment were chosen according to how well they corresponded to images stored in the collections when it came to syntactical resemblance. By having seed images with similar contents to the stored images, both DBS' would have good chances to return similar images, and any effect of utilizing context information when ranking the results would thus become more visible.

Most of the bridges stored in the collection are quite famous either from an engineering standpoint, or with regard to historical significance. The assumption made was that because of this, images of many of the bridges stored could probably also often be used as seed images. As a result, the experiment used images of bridges already stored in the collection in half of the queries while the other half of the queries used seed images of bridges not previously

stored in the collection. These images still had a high degree of syntactical resemblance to the stored images.

4.4.2 Query Set

Queries used in this experiment were set up as content-based queries based on the image contents of seed images supplemented with text-based queries. Thus, each query in the query set consisted of two input parameters sent by the user to the participating DBS', namely i) an example image, accompanied by the ii) keywords or phrases that the user wanted the systems to use for querying the stored context information.

The queries used in the experiment are shown in table 6. The Raw Score merging used results from the queries using the seed images only. The seed images used in the experiment are compiled in Appendix E

Table 6 - Queries used in the experiment

Query no.	Query
1	Find images resembling seed image 1. context of interest: Bridges being struck by or being near lightning (activity)
2	Find images resembling seed image2. context of interest: Bridges in Boston, Portland, San Francisco, St. Louis, Cincinnati, or Rockport (location)
3	Find images resembling seed image 3. context of interest: bridges carrying railway (identity)
4	Find images resembling seed image 4. context of interest: depicting daytime (time)
5	Find images resembling seed image 5. context of interest: depicting construction work (activity)
6	Find images resembling seed image 6. context of interest: hollow box girder and steel girder (identity)
7	Find images resembling seed image 7. context of interest: Bridges in Norway, China or Japan (location)
8	Find images resembling seed image 8. context of interest: depicting illuminated bridges at night (time)
9	Find images resembling seed image 9. context of interest: commit suicide (activity)
10	Find images resembling seed image 10. context of interest: A bridge collapsing galloping in the wind. (identity)
11	Find images resembling seed image 11. context of interest: connecting Europe and Asia (location)
12	Find images resembling seed image 12. context of interest: low or high tide (time)

The context category represented in each query is specified in parentheses. Because the context descriptions used were quite general, it was necessary to have some knowledge of the contents of the stored text-documents as this determined the kinds of contexts that were available for queries.

Each context category is represented three times in the query set. This approach allowed for some variation to be made in the query terms used for each query category. An example of this is the query set 2, 7, and 11. Query numbers 2 and 11, is quite location specific, while query 7, is less specific in pinpointing location.

Due to specification of image context in relatively unambiguous terms, and presented in a structured framework, all queries should provide results.

4.5 Ranking Query Results

As use of a traditional Raw Score model, described in chapter 2, is an available approach to merging and ranking of multiple query results in environments similar to that presented in this thesis, the Raw Score model thus served as a benchmark tool for results produced by the extended ranking algorithm implemented in the CAIRANK prototype.

The Raw Score merging method utilize the normalized content-based similarity scores alone when ranking result sets returned from the DBS'. These similarity scores come from image queries using a seed image only, as illustrated in figure 37.

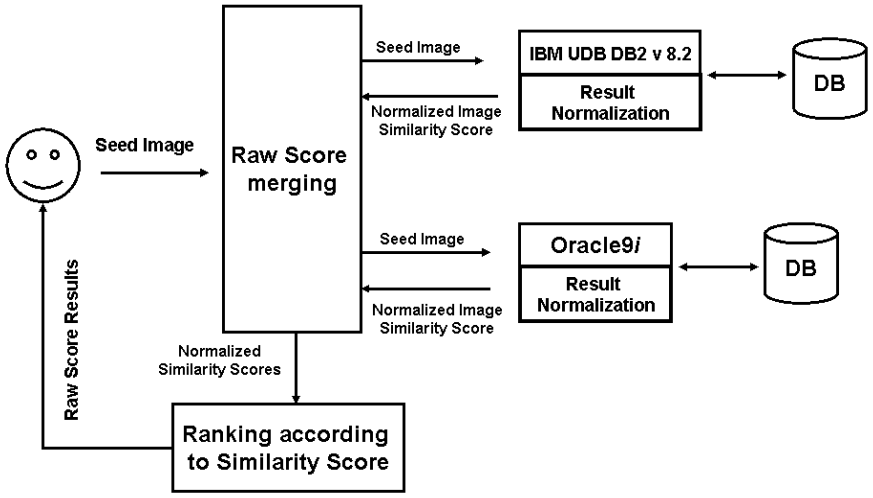


Figure 37 - Raw Score Ranking Process

The CAIRANK prototype used a similarity score from the content-based query, a similarity score from the text query, as well as weights calculated for each DBS when computing a new score. This is illustrated in figure 38. Thus, the major difference between the between the CAIRANK prototype and a traditional Raw Score merging model is in the CAIRANK prototype's ability to utilize both the content-based similarity scores and the text-based similarity scores in combination with the database weight in calculating a new score to be used for ranking purposes.

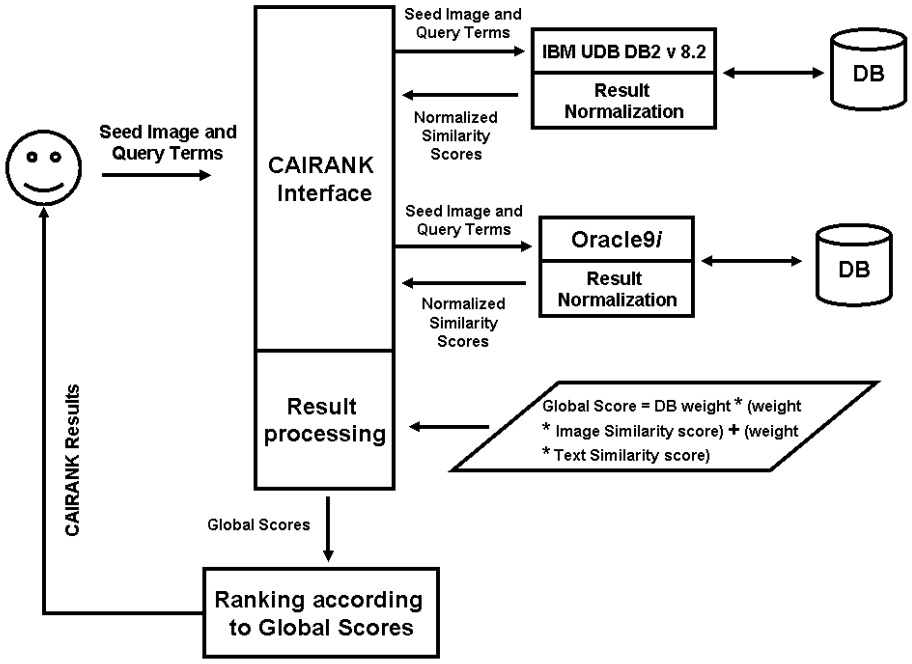


Figure 38 - CAIRANK Ranking Process

By modifying and extending a Raw Score ranking model to be used in the CAIRANK prototype, while using the original ranking model as foundation for comparison, two similar models rank results from identical queries against the same collections in an experimental setting. In doing so, nothing separates the two models except for the modified ranking method's ability to take in and calculate a score using weights and the similarity score from two incoming parameters in the ranking process.

4.6 Data analysis – Measuring Precision and Distance

Both the CAIRANK prototype and the standard Raw Score merging model generated one merged and ranked result list gathered from each distributed visual query against collections stored in the participating DBS's. Assessing these results with statistical methods provided the possibility of measuring to what extent the results obtained with the CAIRANK prototype deviated from results obtained with the regular Raw Score merging method.

A commonly used tool for evaluating the effectiveness and quality of different methods for information retrieval is evaluating the recall and precision measures for each method (Baeza-Yates & Ribeiro-Neto, 1999; Lu, 1999). Utilizing these measures forms a good base for comparing how well different techniques for information retrieval perform. Here, the recall measure indicates how many of the relevant items in the collection as a whole are retrieved, while precision is a measure on how many of the retrieved items actually are relevant.

4.6.1 Measuring Precision While Omitting Recall

When executing queries against the entire collection stored in each DBS, the retrieved data items form a sub set of the total database content. Measuring recall and precision is relatively straightforward by looking at the relevant items retrieved in the sub set versus the relevant items in the collection as a whole to measure recall, or the number of relevant images

retrieved versus the total number of retrieved images to measure precision (Baeza-Yates & Ribeiro-Neto, 1999).

There were some problems associated with the usage of recall measures in this project. The first problem was that the recall measures for CAIRANK initially was evaluated against the collection as a whole, but later evaluated against the retrieved data items from the first query, now acting as the whole collection. Since CAIRANK was executing queries against two different collections in each DBS, while the Raw Score model used only one reference collection, the two recall measures were incomparable.

A second problem associated to using recall measures for evaluating effectiveness in this project relates to the result sets forming the basis for the merging and ranking process. Since each participating DBS provided all the results, without involvement from either CAIRANK or the Raw Score method, the recall measures were merely usable for measure the effectiveness of retrieval algorithms in the participating DBS. The aim of this project was not to assess the algorithms of the participating DBS; the calibration formula presented in section 3.4.4 served as an aid to factor in the expected recall level in each participating DBS, so the measures of their individual effectiveness were not particularly interesting in this phase of the experiment.

Following from this, the final ranking from both the Raw Score model and CAIRANK would ultimately present the same images in the final merged result, and both methods would therefore have identical recall measures. The main difference is the approach taken to rank images from multiple result sets, not in the ability to retrieve relevant data items.

Precision on the other hand, may aid in assessing how well one ranking method is performing compared to another ranking method. Precision measures the number of relevant images in the whole result set, as illustrated in figure 39. Since CAIRANK processed the returned result lists further and ranked results using both a text-based similarity score and a content-based similarity score, the global result should differ from that of the Raw Score merging method.

$$\text{Precision} = \frac{r}{N}$$

where
r is the number of relevant documents retrieved for a given query.
N is the number of documents retrieved for a given query.

Figure 39 - Formula for calculating precision

In this experiment, interval precision was chosen with an interval of 5 documents as the inspection point for calculating the precision levels, resulting in 20% intervals by calculating precision for:

- first 5 documents retrieved
- first 10 documents retrieved
- first 15 documents retrieved
- first 20 documents retrieved

This gave four precision measures per query result, reasonable for comparing the performance of the two ranking approaches. These measures were comparable, and illustrated the performance level for each of the ranking method considered in this project.

The reason for calculating Precision values for the top-twenty retrieved images resided on the notion that most users are interested in relevant images being placed early in the result set. In addition, this experiment also evaluated performance by measuring distance, and here the full result set for each query in each of the participating DBS' was used.

4.6.2 Measuring Result set quality using Distance

In order support the precision measures with a more specific measure than the 20 % Precision measure, Distance was used. The goal was to calculate the distance between the ranked results of each participating DBS' and an ideal result set in order to assess if the approach proposed in this thesis increased the relevance of the data items in the result sets returned from the DBS'. This operation required that all test queries to be used with the CAIRANK prototype also were submitted manually to each of the participating DBS'. This was achieved by executing the same procedures used by the CAIRANK procedure. Please see appendix B for the actual code.

When using distance as precision measure in order to determine the quality of each result set, an ideal result set for every query was set up for each query in both databases. Each ideal result set had the most relevant data items of each DBS positioned according to their evaluated relevance. The positioning of an data item in a result set ranked using a content-based similarity score only and a result set ranked using both a content-based similarity score as well as a text-based similarity score was compared to the positioning of data items in an ideal result set. This comparison provided the distance measures for each retrieved data item.

The average distance measure for each result set item for each query was then calculated, and the ranking approach with the lowest average distance measure signalled a ranked result set of higher precision according to the given ideal result set.

5 Experimental Results

After designing and implementing all components described in the previous chapter, the experiment was conducted. The purpose of the experiment was to generate and collect data for further analysis.

5.1 Collecting data

As discussed in chapter 3, queries using the CAIRANK prototype were initiated by typing the number of the image to be used as the seed image in the content-based query and the keywords or phrases to be used for the text-based query as shown in figure 26 (page 46). The query text could consist of single words or phrases.

As its primary function in this project was to produce data for further analysis, the implemented version of the CAIRANK prototype wrote both the result sets produced by CAIRANK as well as the results to be used for the Raw Score Merge approach directly to an Excel file for further processing. Both sets generated by the prototype consisted of the image numbers used as primary keys in the database, the image titles and the score for each image as illustrated in table 7. The table is depicting the results from query number one submitted to the CAIRANK prototype.

Table 7 - Example of output using the CAIRANK prototype

Query No:1

Image No:	Image Title:	Raw Merge Score:		Image No:	Image Title:	CAIRANK Score:
608	Severn Bridge	1		353	Manhattan Bridge	4,70213803
785	Yangtze River Bridge	1		323	MacDonald Bridge	4,69684884
384	Millau Bridge	0,99504		587	San Francisco Bay Bridge	4,67506837
515	Rama VII Bridge	0,99129		673	Tacoma Bridge	4,60708234
442	Old Lisbon Bridge	0,9891		89	Bosphorus Bridge	4,57058195
191	George Washington Bridge	0,98868		675	Talmadge Bridge	4,36243245
214	Golden Gate Bridge	0,98799		585	San Francisco Bay Bridge	4,33758324
460	Pasco Kennewick Bridge	0,98776		589	San Francisco Bay Bridge	4,33361635
430	Normandie Bridge	0,98444		674	Tacoma Bridge	4,29802785
257	Hakucho Bridge	0,98442		235	Golden Gate Bridge	4,12302184
686	Tasman Bridge	0,98388		586	San Francisco Bay Bridge	3,98268195
340	Manhattan Bridge	0,98382		335	Mackinac Bridge	3,97262927
46	Astoria Bridge	0,98102		488	Queen Isabella Causeway	3,90357315
712	Ting Kau Bridge	0,98007		490	Queen Isabella Causeway	3,87750912
412	Natcher Bridge	0,97901		236	Golden Gate Bridge	3,87709582
274	Hoga Kusten Bridge	0,97824		588	San Francisco Bay Bridge	3,8264593
20	Ambassador Bridge	0,97718		234	Golden Gate Bridge	3,53770346
220	Golden Gate Bridge	0,97678		237	Golden Gate Bridge	3,45992732
750	Verrazano Bridge	0,97584		232	Golden Gate Bridge	3,38414645
213	Golden Gate Bridge	0,9743		387	Mississippi Bridge	3,3282964

In table 7, the *Raw Merge Score* is the normalized image scores as calculated in each participating DBS. The *CAIRANK Score,* is the global score as calculated in the CAIRANK prototype using the normalized content-based and text-based similarity scores combined with the calculated database weights.

In addition to running the queries using the CAIRANK prototype, where Precision was used to measure the quality of the result sets, each query was also submitted manually to each of the participating DBS' by using the DBMS interface. This was done to be able to compare the single score approach to the combined score approach while avoiding potential conflicts in query and ranking algorithms implemented in the participating DBS'. The results from the

manually submitted queries were also collected in an Excel file for further processing. Here, the result sets were ranked either using the content-based similarity score alone, or by using the combination of content-based and text-based similarity scores as illustrated in table 8. The full result set from the queries submitted manually to the participating DBS' can be found on the enclosed CD.

Table 8 - Example of results from manually submitted queries

Ibm Query 1:

	Single score ranking				Combined score ranking				
#	Image no	Sim Score	Image title		Image no	Image title	Norm image	Norm text	COMBO
1	608	0,06949	Severn Bridge		674	Tacoma Bridge	0,95721	0,99656	1,95377
2	384	0,07802	Millau Bridge		586	San Francisco Bay Bridge	0,85119	0,98949	1,84068
3	442	0,08824	Old Lisbon Bridge		488	Queen Isabella Causeway	0,80904	1	1,80904
4	214	0,09016	Golden Gate Bridge		236	Golden Gate Bridge	0,81122	0,98934	1,80056
5	460	0,09055	Pasco Kennewick Bridge		490	Queen Isabella Causeway	0,79936	1	1,79936
6	430	0,09626	Normandie Bridge		588	San Francisco Bay Bridge	0,79317	0,98949	1,78266
7	686	0,09723	Tasman Bridge		430	Normandie Bridge	0,98444	0,72226	1,7067
8	340	0,09734	Manhattan Bridge		46	Astoria Bridge	0,98102	0,72132	1,70234
9	46	0,10216	Astoria Bridge		428	Nordhordlands Bridge	0,9311	0,77029	1,70139
10	712	0,10379	Ting Kau Bridge		300	Lantau Link Bridge	0,96936	0,72882	1,69818
11	412	0,10561	Natcher Bridge		712	Ting Kau Bridge	0,98007	0,7101	1,69017
12	274	0,10694	Hoga Kusten Bridge		412	Natcher Bridge	0,97901	0,70995	1,68896
13	20	0,10877	Ambassador Bridge		654	Second Bosphorus Bridge	0,97405	0,71413	1,68818
14	220	0,10945	Golden Gate Bridge		460	Pasco Kennewick Bridge	0,98776	0,69652	1,68428
15	750	0,11107	Verrazano Bridge		608	Severn Bridge	1	0,68389	1,68389
16	654	0,11416	Second Bosphorus Bridge		342	Manhattan Bridge	0,97301	0,70394	1,67695
17	342	0,11595	Manhattan Bridge		234	Golden Gate Bridge	0,67872	0,99466	1,67338
18	54	0,11678	Bear Mountain Bridge		20	Ambassador Bridge	0,97718	0,69531	1,67249
19	250	0,11683	Great Belt Bridge		150	Dames Point Bridge	0,94687	0,72516	1,67203
					Result is truncated				
398	358	1,56004	Marquam Bridge		798	Zakim Bridge	0,19742	0	0,19742
399	766	1,74232	Walt Whitman Bridge		358	Marquam Bridge	0,13417	0	0,13417
400	768	1,79102	Walt Whitman Bridge		768	Walt Whitman Bridge	0	0	0

In table 8, displaying a shortened version of the results from query number one submitted to the IBM DBS, the *Single score ranking* approach ranked the query results according to the content-based similarity score alone. In contrast, the *Combined score ranking* approach ranked the query results according to the combination, called COMBO, calculated using the normalized content-based and text-based similarity scores.

5.2 Organizing and Processing Data Collected in the Experiment

As discussed in chapter 4, the data collected in the experiment were analyzed using Precision and Distance as measuring tools.

Result sets from both the Raw Score merging approach and the CAIRANK were arranged in tables for calculating precision for results obtained by the two approaches on each query. An average Precision performance over all queries was also calculated for each of the two approaches. Please see appendix H for Precision results for all queries.

Result sets from the manually submitted queries were arranged in Distance tables in order to record differences in ranking performance when using one versus two similarity scores. Distance diagrams were used to visualize these differences. Please see appendix G for Distance results for all queries.

5.2.1 Calculating Precision

Figure 40 is an example of how the data from each query was organized and processed when calculating Precision. Here, information listing the number of the query image used and the query terms used is presented at the top of the table. Below, the results produced of the two approaches are presented side-by-side. Relevant images are highlighted in grey and Precision values at each inspection point are presented for both approaches. These Precision values are also illustrated graphically below.

Images deemed relevant when calculating Precision had to have image content displaying a fair amount of syntactical resemblance to that of the seed image. In addition, context information associated with the image had to satisfy the text-query. As this project did not include other people in the experiment, the author did the relevance evaluation when calculating precision. Possible ramifications of this regarding the achieved results and some possible consequences concerning bias are discussed in chapter 6.

Query No:2 **Query image 2**

QT: U.S. Boston, Portland, San Francisco, St. Louis, Cincinnati, Rockport

Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
797	4,9898		614	1	
407	4,597402128		797	1	
795	4,506063839	80	774	0,9958	40
409	4,403872735		588	0,9853	
523	4,386682874		292	0,98494	
631	4,281946972		540	0,98346	
799	4,265854867		208	0,98337	
793	4,241928776	80	498	0,98322	40
129	4,221520494		524	0,98278	
640	4,219580895		640	0,98064	
95	4,218152379		126	0,97795	
208	4,212836879		128	0,97605	
133	4,210093852	80	82	0,97102	33,33333333
128	4,207808581		706	0,97075	
588	4,170625583		468	0,96285	
317	4,166882184		258	0,96216	
648	4,159805942		648	0,95844	
574	4,14195211	70	254	0,95736	30
572	4,138694106		574	0,95714	
591	4,134523331		572	0,95593	

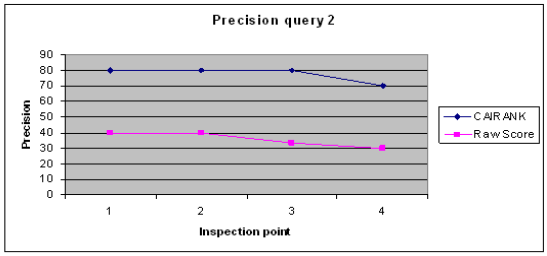


Figure 40 - Example on how Precision is calculated and presented graphically

In the example in figure 40, the Precision graph, with a blue curve for Precision values in the CAIRANK results and a pink curve for Precision values in the Raw Score results, displays the Precision of each approach at each of the four inspection points. The horizontal axis represents the four inspection points while the vertical axis represents the degree of precision.

In the Precision graph presented in figure 40, the Precision curve of CAIRANK is above the Precision curve of the Raw Score approach. Thus, CAIRANK has the best Precision values for this query.

5.2.2 Calculating Average Precision

The average Precision values were calculated by adding the Precision values taken at each inspection point for all queries and then divide the sum on the total number of queries. Hence, the Precision value taken at the first inspection point in the first query was added to the Precision value taken at the first inspection point in the second query and so on for all queries. The total sum for each inspection point for all queries was then divided by 12. Table 9 displays the average Precision values for both approaches.

Table 9 - Average precision for the two approaches

	Average precision CAIRANK			Average precision Raw Score
Inspection point	Precision		Inspection point	Precision
5	81,66666667		5	5
10	70		10	8,333333333
15	66,66666667		15	6,111111111
20	59,58333333		20	5,833333333

The results presented in table 9 were visualized as precision curves in figure 41. Again, the horizontal axis represents the four inspection points while the vertical axis represents the degree of Precision.

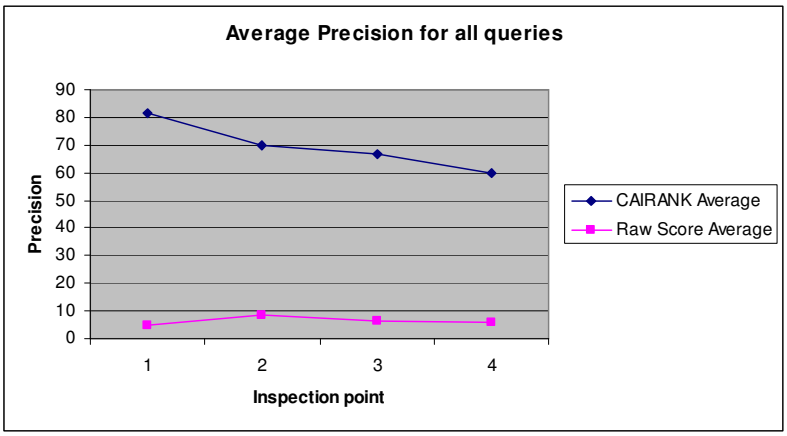
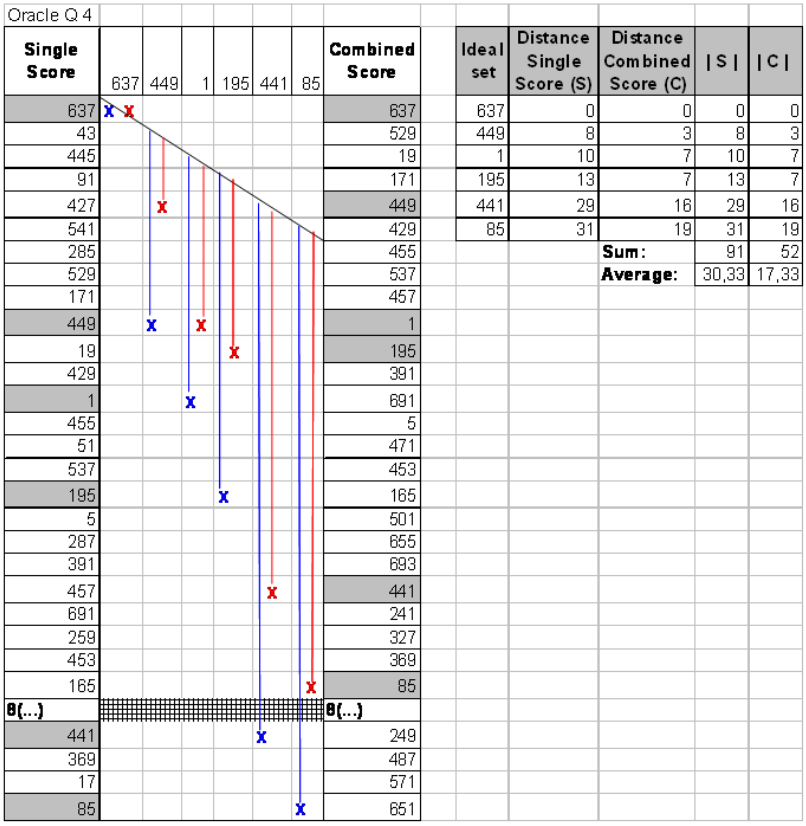


Figure 41 - Visual presentation of average Precision values

5.2.3 Calculating Distance

The Distance diagram in figure 42 is an example of how the data from each query was organized and processed when calculating Distance. The results produced using the two ranking approaches are presented side-by-side. Relevant images are highlighted in grey and the distance between the placement of images in the ideal set and the placement of the images

retrieved by the two approaches is illustrated visually in the middle columns, using red and blue lines for the combined and single scores respectively.



Oracle Q 4

Single Score	637	449	1	195	441	85	Combined Score
637	X	X					637
43							529
445							19
91							171
427		X					449
541							429
285							455
529							537
171							457
449		X		X			1
19				X			195
429							391
1			X				691
455							5
51							471
537							453
195				X			165
5							501
287							655
391							693
457					X		441
691							241
259							327
453							369
165						X	85
8(...)							8(...)
441					X		249
369							487
17							571
85						X	651

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
637	0	0	0	0
449	8	3	8	3
1	10	7	10	7
195	13	7	13	7
441	29	16	29	16
85	31	19	31	19
		Sum:	91	52
		Average:	30,33	17,33

Figure 42 - Example of a Distance diagram and a Distance table

The first vertical column in the Distance diagram in figure 42 displays the results, represented by the image identifiers, ranked using content-based similarity score only. The horizontal row, lists the ideal set from left to right ranged by relevance. The last vertical column displays the results ranked using a combination of content-based and text-based similarity scores. Each blue “X” represents the placement of a relevant image using content-based when ranking results using similarity score only, while a red “X” represents the placement of a relevant image when ranking results using both the content-based and text-based similarity scores. The diagonal line represents the ideal placement of the relevant images. If images ranked by the two approaches considered here deviated from the ideal placement, i.e. did not follow the diagonal line, this is visualized with a blue or red vertical line. The longer the line, the further away from the ideal set an image was placed.

As these result sets consisted of 400 images, results were truncated by collecting irrelevant images in the visual representation of Distance. This was marked with the number of irrelevant images collected in each set as well as a horizontal square pattern across the diagram. For instance, 8(…), represents eight irrelevant images.

The table to the right of the Distance diagram in figure 42 is an example of a Distance table as used in the processing of data collected in the experiment. The first column consists of the

ideal set for a particular query. The second and third column shows the distance from the ideal set for each of the ranked result sets. If an image is placed earlier in the result set than specified in the ideal set, a negative distance is recorded. Otherwise, a positive distance is recorded. When calculating distance, positive and negative distance is used in the same manner, i.e. both 4 and -4 is calculated as a displacement of 4.

The fourth and fifth column shows the absolute distance value between the placing of each relevant image by the two approaches respectively and the ideal set. At the bottom of these columns, distance values are summed up, and an average distance value is calculated for each ranking approach. The approach with the lowest sum or average has ranked the results better as it is closer to the ideal set. In the distance table displayed in figure 42, the Distance Single score (S) ≈ | S | has the highest sum with a total displacement of 91 places, an average of 30,33 for each image. The Distance Combined score (C) ≈ | C | has the lowest sum of the two ranking approaches with a total displacement in the result ranking numbering 52, an average of 17,33 for each image. This indicates that the Distance Combined score approach has ranked the results better than the Distance Single score on this query.

Images deemed relevant when calculating Distance were more closely scrutinized than was the case in determining which images were relevant when measuring Precision. The reason for this was that images chosen for the ideal set should represent the sample selected if a human was to pick relevant images from the collection.

Images, if to be used in the ideal set, had to display an equal or higher degree of syntactical resemblance to the seed image than was the case with images deemed relevant when calculating Precision. In addition, they also had to fulfil criteria given by the text-query for that given query. Again, as this project did not include other people in the experiment, the author also did the relevance evaluation when calculating Distance. As with the possible consequences concerning results from calculating Precision, some possible ramifications of this regarding the achieved Distance results is also discussed in chapter 6.

5.2.4 Calculating average Distance

A table showing the total Distance and average Distance for each query was created for results produced by each of the participating DBS'. In addition, the table displayed the average Distance combined for all queries in each system as shown in table 10.

Table 10 - Distance summary table

Ibm						Oracle					
Query no.	| S |	| C |	Relevant retrieved	Average | S |	Average | C |	Query no.	| S |	| C |	Relevant retrieved	Average | S |	Average | C |
1	578	56	3	192,67	18,67	1	719	29	5	239,67	9,67
2	29	6	4	9,67	2,00	2	106	23	4	35,33	7,67
3	529	54	5	176,33	18,00	3	543	27	4	181,00	9,00
4	1305	943	8	435,00	314,33	4	91	52	6	30,33	17,33
5	1943	1269	8	647,67	423,00	5	476	251	8	158,67	83,67
6	805	25	4	268,33	8,33	6	1306	47	6	435,33	15,67
7	674	93	7	224,67	31,00	7	261	268	5	87,00	89,33
8	1451	588	7	483,67	196,00	8	833	503	4	277,67	167,67
9	699	49	4	233,00	16,33	9	382	29	5	127,33	9,67
10	1162	14	5	387,33	4,67	10	1034	25	5	344,67	8,33
11	463	828	3	154,33	276,00	11	398	15	3	132,67	5,00
12	262	18	5	87,33	6,00	12	542	95	5	180,67	31,67
Total	9900	3943	63	157,14	62,59	Total	6691	1364	60	111,52	22,73

The first column in table 10 lists the twelve queries. The second and third column show the total distance value between the ideal set and result ranked by the two approaches considered in the project. The fourth column shows the number of relevant images retrieved, i.e. in the ideal set. The fifth and sixth column present the average Distance between the ranked results and the ideal set for each query submitted to the participating DBS'. At the bottom of the table, results are summarized and a total Distance value for the two approaches is calculated.

The average Distance values calculated for each query is visualized using histograms as shown in figure 43. Each histogram illustrates the average Distance for each query in each of the participating DBS'. The results for the ranking approach using single score are visualized using blue while the results for the ranking approach using a combined score approach are visualized using red bars. The approach with the lowest average Distance value has ranked the results better when compared to the ideal result set.

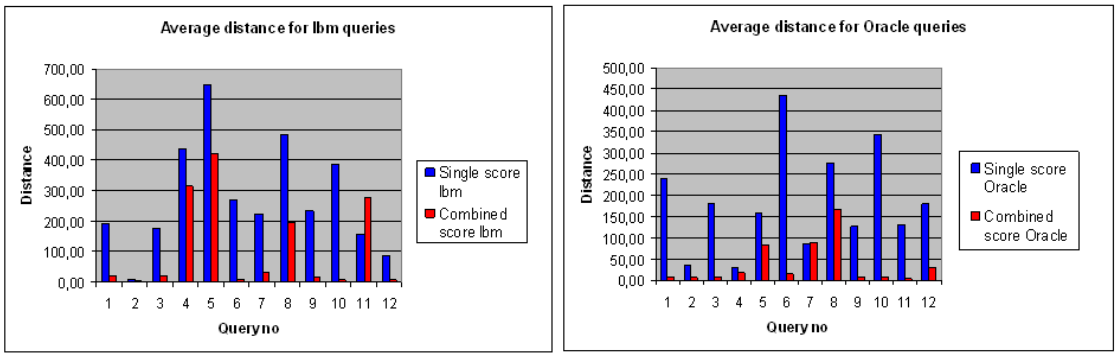


Figure 43 - Visual presentation of average Distance values

The average differences in the actual positioning from the five first ideal images in the result sets from each query when using the two approaches is illustrated in figure 44. Here, the Distance value measured between the first ideal image and the image ranked as number one in the first query was added to the Distance measured between the first ideal image and the image ranked as number one in the second query and so on for all queries. The total Distance

sum from each ideal image for all queries was then divided by 12. The horizontal axis represents the five first ideal positions in each query. The vertical axis displays the average Distance from each ideal position to the actual positioning by the two different ranking approaches. The method with the lowest curve has ranked the results better when compared to the ideal result set.

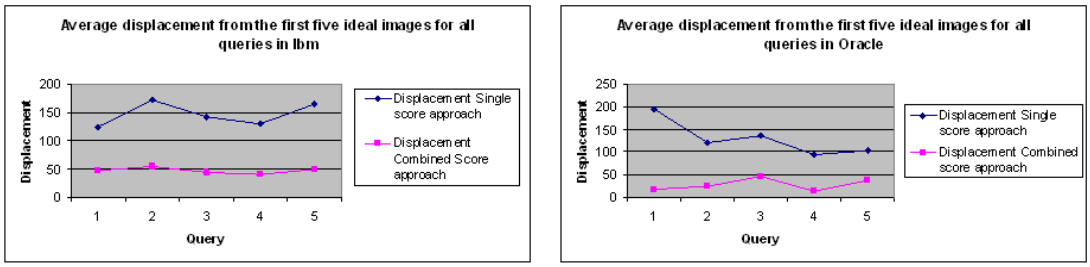


Figure 44 - Average displacement from first five ideal images for all queries

Figure 44 display that images that should have been placed at the top of the result sets from the IBM DB2 and Oracle9*i* databases on average was displaced by 47,83 and 17,25 positions respectively using the combined score approach. For the single score approach the average displacement was 125,33 and 196,16 positions respectively.

5.3 Significance testing

The tables and graphs presented in the prior section generally indicate that ranking results utilizing both a content-based and a text-based similarity score outperforms ranking done using a content-based similarity score alone. The diagram displaying average Precision for all queries, presented in figure 41, indicates that the degree of Precision is higher when ranking results using CAIRANK (utilizing both content-based and text-based similarity scores). The diagrams displaying average Distance for all queries, presented in Figure 43, indicates that the degree of displacement is lower in both participating DBS' when ranking results using both content-based and text-based similarity scores.

In order to determine if the differences indicated are real or if they are the result of a spurious effect, significance testing of the results was preformed. There are several tests available for testing if significant differences exist between two sets of data. Still, some problems in using these test for evaluating IR systems has been noted (van Rijsbergen, 1999). Statistical tests, like *Students-t* test, are based on certain assumptions about the nature of the underlying data:

- The samples are independently and randomly drawn from the source population
- That the scale of measurement for both samples has the properties of an equal interval scale
- That the source population(s) can be reasonably supposed to have a normal distribution

When evaluating IR systems, problems are most often associated with assuming a normal distribution of the results. When evaluating if results were significant in this thesis it is assumed that data had a normal distribution, and thus a *student-t* test is used to determine if results are significant. However, any conclusions drawn from these tests must take the assumption of a normal distribution in the results into account.

The following null hypotheses were formulated to test for significance:

H0-1:
Combining similarity scores from both text- and content-based queries will not significantly improve Precision when ranking results from an image database compared to ranking using scores from content-based queries only.

H0-2:
Merging and ranking query result sets from multiple database systems by combining similarity scores from both text- and content-based queries will not significantly improve precision compared to ranking using a Raw Score merging approach relying on content-based similarity scores only.

The null hypotheses above will be rejected if it can be established with a probability of 95 %, i.e. a 0,05 level of significance, that the observed results are not the result of a coincidence.

H0-1, refers to the average Distance value results produced by the Single score ranking approach and Combined score ranking approach from both DBS' presented in figure 43. H0-2 refers to the average Precision values produced by CAIRANK and Raw Score merging presented in figure 41 (page 69).

Table 11 and table 12, display results from significance testing for the average Distance values for *Single score* and *Combined score* ranking approaches (figure 43). The significance tests were conducted using statistical analyse tools implemented in Microsoft Excel. A basic understanding of significance testing with student-t test is assumed.

In the tables presenting the significance results, *t-Stat* is the t-value of the t-test. *T-Critical, one-tail* is the t- threshold for a significance level of 0,05 in a one-tailed test, while *T-Critical, two-tail* is the similar threshold for a two-tailed test. Since the H0-1 hypothesis indicates that the difference should be from ranking using a combination of content-based and text-based similarity scores to ranking using content-based similarity score only, it is a directional test, or a one-tailed test, and thus the first threshold value is used.

Table 11 - Paired two-sample t-test for average Distance in IBM

	Variable 1	*Variable 2*
Mean	109,5277778	275
Variance	22763,80724	32879,79798
Observations	12	12
Pearson Correlation	0,69375583	
Hypothesized Mean Difference	0	
Df	11	
t-Stat	-4,31049939	
P(T<=t) one-tail	**0,000617052**	
T-Critical, one-tail	1,795884814	
P(T<=t) two-tail	0,001234105	
T-Critical, two-tail	2,200985159	

Table 12 - Paired two-sample t-test for average Distance in Oracle

	Variable 1	*Variable 2*
Mean	37,88888889	185,8611111
Variance	2530,511785	14767,74663
Observations	12	12
Pearson Correlation	0,076555836	
Hypothesized Mean Difference	0	
Df	11	
t-Stat	-4,0072738	
P(T<=t) one-tail	**0,001030209**	
T-Critical, one-tail	1,795884814	
P(T<=t) two-tail	0,002060418	
T-Critical, two-tail	2,200985159	

Table 11 shows that P(T<=t) one-tail is 0,000617052, while table 12 shows that P(T<=t) one-tail is 0,001030209. This indicates that the differences between the datasets produced for both DBS' are significant at the 0,05 level.

Table 13 - Paired two-sample t-test for average Precision CAIRANK vs. Raw Score

	Variable 1	*Variable 2*
Mean	64,14552845	6,319444444
Variance	74,53359323	2,025462963
Observations	4	4
Pearson Correlation	-0,242480883	
Hypothesized Mean Difference	0	
Df	3	
t-Stat	12,73152081	
P(T<=t) one-tail	**0,000522682**	
T-Critical, one-tail	2,353363435	
P(T<=t) two-tail	0,001045364	
T-Critical, two-tail	3,182446305	

Table 13 shows that P(T<=t) one-tail is 0,000522682. This indicates that the differences between the datasets produced by ranking using the CAIRANK and the Raw score approach are significant at the 0,05 level.

Results from the significance tests indicates that there are significant differences in favour of the Combined score approach both for the datasets produced manually and when using the CAIRANK prototype. The results are discussed further in chapter 6.

6 Evaluation of Results and Conclusion

The focus of this research project, as stated in the research question, has been to investigate if the utilization of context through a combination of content-based and text-based queries could significantly improve the process of merging and ranking of results from distributed image retrieval. Ranking performance was assessed in two different ways. Firstly, by measuring number of relevant images early in a merged and ranked result set by determining Precision, and secondly, by measuring the distance between the rank of an ideal set and the rank produced by two different ranking methods. The principle approach used was to implement a prototype able to produce a score generated by combining similarity scores from content-based and text-based data retrieval. The prototype then wrote the results to a Microsoft Excel file to be merged and ranked. These results were then compared to results produced by merging and ranking result sets using the content-based similarity score only.

With the implementation of the combined score approach in the CAIRANK prototype, it has been shown that implementing a method that can merge and rank distributed results using both content-based and text-based scores is feasible, even if the current implementation has several limitations.

6.1 Hypothesis Evaluation

6.1.1 Verification/Falsification of the Hypotheses

Two hypotheses were put forth in this project:

> H1:
> *Combining similarity scores from both text- and content-based queries will significantly improve Precision when ranking results from an image database compared to ranking using scores from content-based queries only.*
>
> H2:
> *Merging and ranking query result sets from multiple database systems by combining similarity scores from both text- and content-based queries will significantly improve precision compared to ranking using a Raw Score merging approach relying on content-based similarity scores only.*

These hypotheses consist of three major components: i) use of combined similarity scores, ii) quality of search results in Distance or Precision, and iii) a significant improvement.

Use of combined similarity scores has been done by manually executing stored procedures in each participating DBS, and by using functionality implemented in the CAIRANK prototype. The experimental results are described in chapter 4 and can be found in appendices G and H.

Results described in sections 5.2.2 and 5.2.4, indicate that ranking result sets using a combined score approach clearly outperforms results from ranking using a single score approach with regards to both Precision and Distance (figure 41 and figure 43).

The significance tests described in section 5.3, indicate that both the average Distance and the average Precision were significantly better when using a combined score approach versus a single score approach.

The experimental results generated in this project seem to offer support to both hypotheses proposed in this thesis. However, as both the experiment and the foundation from which the results are generated is created specially for this project, and are based on several assumptions, these must be addressed before any conclusions may be drawn regarding answering the research question presented in section 1.5.1 (page 7).

6.2 Evaluation of the Experimental Approach

6.2.1 Reliability and validity

Reliability and validity are central in ensuring the quality of scientific experiments. Reliability refers to if repeated measurements using the same tools would provide the same results. Validity refers to if the tools used actually measure what is intended. High reliability is a prerequisite for high validity (Ringdal, 2001).

The measuring tool used in this project is Precision and Distance. The Precision measurements were based on comparing the images deemed relevant by the author, to results obtained by the two different ranking methods considered based on the seed images and context terms submitted to the CAIRANK interface. The Distance measurements were based on comparing relevant images in the ideal result set to results obtained by manually executing stored procedures locally in each of the participating DBS'.

Reliability in this experiment may have been put at risk by human errors occurring while executing the queries or local procedures, and/or while calculating the Precision and Distance values. Examples of such errors include, but are not restricted to, use of wrong example image, using wrong keywords or phrases, or errors in the calculation of Distance or Precision.

To avoid errors completely, is very difficult, and in this project, precautions were taken in trying to avoid serious errors. Amongst these precautions were multiple checks of both the data and the results from the Precision and Distance calculations in order to help ensure correctness in the results. Hence, it is assumed that there is a sufficiently high degree of reliability in the data material. However, as noted in section 5.3, this is an assumption based on the underlying data material having a normal distribution. Thus, any conclusions based on results from the significance tests must be considered with this in mind.

Other factors influence the validity of the experiment. Of primary concern regarding the outer validity of the experiment, is how the CAIRANK prototype reflects the combined score approach proposed in this thesis. If the prototype does not accurately represent the described framework, claiming that the results gathered in the experiment can be used to answer the research question and hypotheses is problematic at best. Another aspect is associated with seeing the implemented functionality in IBM DB2 and Oracle9*i* as representatives for standard content-based and text-based data retrieval systems.

Another concern, related to the inner validity in the experiment, is to what extent the observed results indeed is an effect of using the combined score approach implemented in the CAIRANK prototype and not a spurious effect caused by uncontrolled variables.

Concerning the prototype itself, the outer validity may be assessed by evaluating the structure and the different components of which it is composed. Of interest is eventual problems, conditions and restrictions associated with the prototype, and if found, determining if these

problems conditions and restrictions are limited to the CAIRANK prototype or are of a more general nature.

A final concern, relevant to the outer validity of the project as a whole, is associated with the Precision and Distance tools used for evaluation of the proposed framework. The question is to what extent these tools actually measure what is supposed to be measured according to the hypotheses.

6.2.2 CAIRANK as Representative for the Combined Score Approach

Central elements of this part of the discussion are related to what extent the functionality implemented in the CAIRANK prototype reflects the combined score approach as described in chapter 2, as well as eventual problems conditions and restrictions associated with the prototype. The experimental results depend on that the prototype sufficiently reflects the combined score approach in order to be of use in determining how useful the approach is.

Table 1 (page 34) in chapter 3, specifies the minimum functionality requirements for the CAIRANK environment to function as an experimental tool. Of these requirements, the numbers 1, 7, 8 and 9 describes specific requirements associated with the Java application:

- Ability to receive queries from a user and submit them to several DBS'
- Ability to retrieve results returned to the temporary table by procedures in each participating DBS
- Implemented functionality to retrieve and process results
- Deliver processed results

In addition, the environment is dependent on that the participating DBS' have the ability to execute both content-based and text-based queries, and being able to store the results for retrieval by the Java application. An application for merging and ranking results is also required.

As discussed in section 3.2, the implemented version of the CAIRANK prototype fulfils these requirements through the getResults() method in the *Queryproc* class that:

- calls on the *DBConnect* class to connect to the participating DBS' and submits the queries provided by the user
- retrieves query results from the temporary table in each participating DBS
- combines the scores retrieved from each DBS, and adds the database weight assigned to each participating database to produce the new score
- and also writes the combined results from the CAIRANK prototype to an Excel sheet for further processing

In addition, the CBIRQuery and docQuery procedures implemented in each of the participating DBS' provide the sub-results to be processed further in the Java application.

Regarding the use of user defined functions for processing results locally in the participating DBS', this may possibly reduce their autonomy. However, the way in which this system was modelled, these modifications did not interfere with local database structure as they acted on a result set stored in a temporary table. In effect, this means that the user defined functions can be adapted to the existing database structure, not the other way around.

From this it is proposed that the implemented version of the CAIRANK prototype represents the combined score approach sufficiently to act as an indication of the capabilities of the framework.

Implementing extensions or changes to the prototype is relatively straightforward, and including the functionality to perform the actual merging and ranking in the prototype for a fully functional version, can be achieved using some sort of vectors or array lists and using some form of sorting routine before results are presented to the user. Furthermore, support for other configurations of the procedures, e.g. using text-based queries as starting point followed by content-based queries, should be easy to implement as they are based on using the same principles as the solutions already implemented. However, this should be tested before drawing any conclusions.

As the implementation and usage of the similarity functions used by Oracle9*i* and IBM DB2 are not public knowledge, it was not possible to control these factors in the project. The problem associated with the shape weighting in Oracle and the problem associated with determining the distance interval in DB2 discussed in section 3.3.1 (page 47), illustrates some of the potential pitfalls of using proprietary software without having access to the source code. However, as both ranking approaches considered in this project used the same result sets as a starting point for the merging and ranking process, any consequences of not knowing the underlying structures of the software, or errors in the program code, should affect both approaches equally.

In addition, as this project used the implemented algorithms in both systems without modifications in assessing the two different ranking approaches, the use of text-based data retrieval to support the content-based data retrieval in the merging and ranking process is believed to be the main difference between the two approaches. The variance in results is thus attributed to the use of a combined score approach. Furthermore, as none of the DBS' considered in this project utilized object recognition processes in generating signatures, it is believed that they can be said to stand as representatives for standard content-based and text-based data retrieval systems.

Using text search in combination with content-based image retrieval seems promising with regards to increase the number of relevant images in the result set. However, there are several fundamental problems commonly associated with text search:

- Text search is language-specific and context-specific. When a user searches by text, he or she must choose a language in which to specify the search. Even within a given language, there are many ways to specify (or attempt to specify) a request for information objects.
- Text search is highly error-prone. Typographical errors result in erroneous results or an empty result set.
- Text may be cumbersome. Search by text inevitably means that a user must know about the keywords used by that site, or master a complex syntax for specifying non-trivial searches.

Countering these problems is a challenge, but using the full-text approach described in this thesis, provides developers with better chances of rectifying the situation by allowing for fuzzy searches, theme searches, elaborate indexing, and access to dictionaries or thesauri. In addition, benefits related to improved efficiency when using full-text documents for

describing image content was felt. However, this should also be tested in controlled circumstances before drawing any conclusions.

6.2.3 Concerning Database Weights

When determining database weights, a series of content-based image queries formed the basis for the calculation of weights. Even if the approach used in this project can be said to function adequately, a somewhat different approach should possibly be considered. In the actual weighting session preformed in this project, IBM DB2 was assigned a higher weight than Oracle9*i*. In retrospect, the weighting should perhaps be based on using the combined score approach as some of the results indicate that the text-searching algorithms in the two DBS' deviated from another in some of the text-based queries run in the experiment. This led to a poorer performance in the IBM system than was the case with the content-based query, indicating that the results produced by the text-searching algorithms actually hampered the results. However, as a weighting session using the combined score approach probably would favour the CAIRANK approach, only content-based image queries was used in this project. Further testing is required in order to draw any conclusions on which approach will be most suitable.

6.2.4 Concerning the Queries Used in the Experiment

The query set used for collecting data is a central part of the experiment. The query set was created by the author and is supposed to represent a selection of queries plausible to be submitted to an image retrieval system like the one created for this experiment. The query set contains relatively common and mundane terms related to the bridges or their surroundings in accordance with the context categories specified by Dey et al. (1999).

To what extent the query set is representative is a difficult question to answer as it was developed without involving other people. In order to try to mimic a query set that represents the information need of different people, several queries for each context category was created in order to create some variance in the query set. This variance is hoped to help avoid bias in the queries as well as contribute to a realistic query set. Based on this, it is assumed that the query set is representative. However, it is not complete as the query set was kept relatively simple in an effort to better being able to spot any effect of using the combined score approach.

6.2.5 Concerning the Experimental Results

The foundation of the experimental results was the number of relevant images in a result set. As no one else was involved in the project, the author created the image collection and also determined which images were relevant when calculating Precision and Distance for each query. A serious problem here may be bias and perhaps an unintentional favouring of the CAIRANK approach. In an effort to counter this, the images in the collection had a low degree of semantic complexity. The images are for the most part very basic, having one defined shape against a relatively stable, homogeneous background. In addition, the results from each query were closely examined, and in order to be deemed relevant, an image had to display a syntactical resemblance to the seed image as well as depict the desired context. An example of this is illustrated below. Given the query image in figure 46 (with lightning being the context of interest), images in both figure 45 and figure 47 was ranked high by the CAIRANK prototype. When manually determining relevance, the images in figure 45 were by this author deemed relevant in satisfying an information need. However, the images in figure 47, although depicting lightning (thus satisfying the context criterion), were deemed irrelevant for this query because of their lacking in syntactical resemblance.

Figure 45 - Example of images deemed relevant

Figure 46 - Query image 1

Figure 47 - Example of images deemed irrelevant

Another step taken in trying to avoid bias, both human and related to scoring algorithms implemented in the two DBS', was to separate the Precision and Distance measurements. As a result, Precision was measured in the top-twenty results produced by the CAIRANK application, while Distance was measured using the whole result sets (about 400 images pr. database) produced by manually executing all queries using the stored procedures locally in the two DBS' considered in this project.

The approach taken in this experiment regarding determining relevance was somewhat different for Distance and Precision. When calculating Precision the approach was to view the results produced by the CAIRANK prototype with "fresh" eyes as if having no knowledge of the images in the collection and then calculate Precision. With Distance on the other hand, the images chosen for the ideal set were selected after a thorough examination of all the images in the collection. To what extent the ideal result sets and images deemed relevant are representative is also difficult to answer as these were also chosen by the author. Even if the context categories used were relatively clear, as illustrated in the example above, it is still possible that they are biased. However, determining the truthfulness of an ideal set is always difficult as the relevance criterion is based on human interpretation which also is context dependent. The ideal sets and images deemed relevant are therefore assumed to be representative.

6.2.6 Concerning Precision and Distance as Measuring tools

As discussed in section 1.5.1, Precision is a widely used tool for evaluating a retrieval system. However, the question remains if Precision is the best tool in evaluating which of two approaches has the best performance when ranking results form distributed image retrieval. Precision may give an indication of the quality of a result set, but in this case, the Precision measures do not say anything about how well CAIRANK actually ranked results from distributed image retrieval.

Another concern is that a Precision measure is based on a binary evaluation. An image is either relevant or non-relevant. In reality however, relevance is usually not binary in that something may be "very" relevant, or "somewhat" relevant. This varying degree of relevance cannot be captured when measuring Precision, revealing a weakness associated with the tool itself, emphasizing the importance of also considering alternative measuring tools in evaluating performance.

By using Distance, the actual displacement of an image can be measured, giving an indication on how well a ranking method performs. Given two result sets, determining the best performance is relatively straightforward by identifying the result set with the lowest distance from an ideal set.

However, there are also weaknesses associated with using Distance values in evaluating ranking performance. First, use of Distance is not a common strategy in evaluating image retrieval systems. This may make comparison to other systems more difficult. Second, the weakness related to determining relevance in measuring Precision also applies to Distance.

In the end, the results produced in this experiment can not be generalized to other populations or application areas. To widen the scope, further experiments using different collections and other measurement tools should be run in order to evaluate the fruitfulness of the approach proposed in this thesis.

6.3 Conclusions from the Research Project

Concerning what conclusions to draw from this research project, the concepts of reliability, validity and generalization will be discussed.

Based on the discussion in the previous sections in this chapter, this research project is believed to have a high degree of reliability as a high degree of control has been enforced throughout. In addition, this experiment consists of relatively few variables, making controlling the environment less problematic.

In addition, it is believed that the internal validity of the experiment with regard to the research question and hypotheses is fairly high. The only difference between the ranking approaches is the use of two similarity scores in the combined score approach. The query set, the collections and the ideal result sets are identical for the two approaches. Thus, the observed differences in Precision and Distance are most likely a result of using the combined score approach. It is not likely that other elements have influenced the ranking of the result sets.

Some factors have indeed affected the external validity of this project. There are problematic issues associated with using proprietary systems without having access to the source code. In addition, the determining of database weights might not have been optimal. Furthermore, as laboratory experiments are run with a high degree of control, this creates a less realistic situation, reducing the possibilities of generalizing the results to other populations or areas.

The research question formulated in section 1.5.1 is repeated here:

> *Can combining similarity scores from both text- and content-based queries significantly improve the process of merging and ranking multiple result sets from distributed image retrieval?*

The experiment run in this project has shown, given certain limitations and conditions, that it is possible to improve the ranking of results from distributed image retrieval by using context information in a combined score approach. Both Precision and Distance results indicate that the combined score approach implemented in CAIRANK perform significantly better than Raw Score merging. However, it is not easy to determine how well the combined score approach performs.

The results from the experiment run in this project thus show that the combined score approach outperform Raw Score merging with higher Precision values, indicating that using a combination of image retrieval algorithms and text retrieval algorithms is fruitful when ranking results from distributed image retrieval. Although the results cannot be generalized, this project may be viewed as a pilot study providing useful results for further studies utilizing the proposed framework.

6.4 Limitations

A master's degree thesis has limited possibilities to address many aspects related to execution of distributed content-based queries, and retrieving and merging the result sets. In this project, the focus has solely been on ranking result sets from distributed image queries based on results provided by the participating DBS', hence excluding important problem areas like database heterogeneity, identification and selection of the best information sources, and removal of duplicates in the local result sets. In addition, search efficiency in terms of computational costs and execution time have not been evaluated.

The CAIRANK prototype has not been developed into a fully working image retrieval meta-search engine and does not yet contain all the suggested functionality presented in this thesis. Only the basic functionality needed to evaluate the hypothesis and research question has been implemented.

Much of the known problems associated with both image retrieval and distributed retrieval remain. The most noticeable of these are perhaps the use of different algorithms. The well known problem of calculating similarity scores persist. In this project the score interval in IBM DB2 was categorically much more compact than was the case of Oracle9*i*. In effect, this does affect the final ranked result. However, in what ways and to what extent was not tested, and there was not enough time to evaluate this properly within the time frame of this project.

6.5 Future Research

The discussion in chapter 3 and previously in this chapter, highlight some limitations in the CAIRANK prototype. As the current version only functions as a tool for collecting data in an experimental setting, the first step is to expand the current CAIRANK version into a fully functional prototype to further explore the usefulness of the proposed approach.

In order to conduct a proper evaluation of the framework, the prototype should be tested on a larger scale, having more DBS' participate and adding several new image collections from different domains. The results from the experiment conducted in this project show that the use of text does influence performance when used like in the CAIRANK prototype. However, a more thorough examination of the nature and influence of using of full-text documents should be conducted.

Perhaps of greatest importance to this author, is how further to extend the CAIRANK prototype to involve the users more in the process of determining relevance. As discussed above, there are some difficulties associated with using both Precision and Distance to determine relevance. If the *combined score* approach is as promising as the results obtained in this experiment indicates, providing the CAIRANK interface with a relevance feedback functionality would be the next logical step.

However, a traditional relevance feedback system where the user selects a few relevant images to form the criteria for the next query is not the proposed solution. Here, "relevance" is left to be determined by the DBMS and algorithms. A common approach in systems providing users with possibilities to give feedback on the relevance of results is to have users select some relevant images, which in turn is used in a new query. A potential drawback with this approach is that often a new result set is produced, depriving the user of the possibility of traversing the first result set. This is often not a satisfying solution.

Instead, a more interactive solution is what is desired. A possible alternative is a relevance feedback solution making use of both relevant and irrelevant images in the result set. The relevant images can be used to search through the current result set and "update" the result set with similar images in an interactive manner while irrelevant images may be used to remove similar image from the result set. For the next query in the feedback cycle, the relevant images serve as criteria for desirable images while the irrelevant may serve as a filtering mechanism.

Here, the screen can be split in half by a horizontal line with a given number of relevant images above the line and a "relevance worksheet" below the line. For each of the images above the line the user actually finds relevant and selects, more images resembling these will appear in the "relevance worksheet". If images above the line prove to be irrelevant, the user may remove them, and at the same time be given the opportunity to choose similar images from the "relevance worksheet". This solution does not call for skimming through several pages with images, but instead, the user only use the "first" result page in an interactive manner. If taking this approach, the retrieval system is viewed like a tool for perception, or a pair of goggles, supporting the user to the best of its capabilities.

7 References

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York: ACM Press.

Beigi, M., Benitez, A. B., & Chang, S.-F. (1998). *Metaseek: A content-based meta-search engine for images.* Paper presented at the SPIE 1998 Conference on Storage and Retrieval for Image and Video Databases VI (IST/SPIE-1998), San Jose, CA.

Bergli, N. (2006). *A comparison of oracle and db2 pertaining to text and image retrieval.* Unpublished Masters Thesis, University of Bergen, Bergen.

Bergman, L., Castelli, V., & Li, C.-S. (1997). Progressive content-based retrieval from satellite image archives. *D-Lib Magazine*.

Berretti, S., Del Bimbo, A., & Pala, P. (2004a). Collection fusion for distributed image retrieval. In J. P. Callan, F. Crestani & M. Sanderson (Eds.), *Distributed multimedia information retrieval* (Vol. 2924, pp. 70-83): Springer.

Berretti, S., Del Bimbo, A., & Pala, P. (2004b). Merging results for distributed content based image retrieval. *Multimedia Tools and Applications, 24*(3), 215 - 232.

Bouguettaya, A., Beatallah, B., & Elmagarmid, A. (1999). An overview of multidatabase systems: Past and present. In A. Elmagarmid, M. Rusinkiewicz & A. Sheth (Eds.), *Management of heterogeneous and autonomous database systems.* San Francisco, CA: Morgan Kaufmann.

Breibart, Y. (1990). Multidatabase interoperability. *ACM SIGMOD Record, 19*(3).

Brown, P., Burleston, W., Lamming, M., Rahlff, O., Romano, G., Scholtz, J., et al. (2000). *Context- awareness: Some compelling applications.* Paper presented at the the CH12000 Workshop on The What, Who, Where, When, Why and How of Context-Awareness.

Brown, P. J., & Jones, G. J. F. (2001). Context-aware retrieval: Exploring a new environment for information retrieval and information filtering. *In press*.

Callan, J. P., Lu, Z., & Croft, W. B. (1995). *Searching distributed collections with inference networks.* Paper presented at the ACM-SIGIR, Seattle (WA).

Colombo, C., & Del Bimbo, A. (2002). Visible image retrieval. In V. Castelli & L. Bergman (Eds.), *Image databases: Search and retrieval of digital imagery*. New York: Wiley.

Cox, R. (2001). Developing java applications using db2 image and audio extenders (part 1). Retrieved December 6th, 2006, from http://www-128.ibm.com/developerworks/db2/library/techarticle/cox/0201cox.html

Dey, A. K., Abowd, G. D., Brown, P. J., Davies, N., Smith, M., & Steggles, P. (1999). Towards a better understanding of context and context-awareness. *Handheld and Ubiquitous Computing, Proceedings, 1707*, 304-307.

Dorai, C., & Venkatesh, S. (2003). Bridging the semantic gap with computational media aesthetics. *Ieee Multimedia, 10*(2), 15-17.

Fan, W. G., Gordon, M. D., & Pathak, P. (2004a). Discovery of context-specific ranking functions for effective information retrieval using genetic programming. *Ieee Transactions on Knowledge and Data Engineering, 16*(4), 523-527.

Fan, W. G., Gordon, M. D., & Pathak, P. (2004b). A generic ranking function discovery framework by genetic programming for information retrieval. *Information Processing & Management, 40*(4), 587-602.

Gauch, S., Wang, G., & Gomez, M. (1996). Profusion: Intelligent fusion from multiple, distributed search engines. *The Journal of Universal Computer Science, 2*(9), 637-649.

Guros, L. (2004). Retrieving data from the "ordimagesignature field. Retrieved 06 December, 2006, from http://forums.oracle.com/forums/thread.jsp?forum=78&thread=222477&message=617161&q=7369676e6174757265#617161

Hausken, L., & Larsen, P. (1999). *Medievitenskap*. Bergen: Fagbokforl.

Hove, L. J. (2004). *Improving image retrieval with a thesaurus for shapes the vortex prototype.* University of Bergen, Bergen.

Jaimes, A., & Chang, S. F. (2002). Concepts and techniques for indexing visual semantics. In V. Castelli & L. D. Bergman (Eds.), *Image databases: Search and retrieval of digital imagery* (pp. 497-565). New York: John Wiley & Sons, Inc.

Jones, G. J. F., Groves, D., Khasin, A., Lam-Adesina, A., Mellebeek, B., & Way, A. (2004). *Dublin city university at clef 2004: Experiments with the imageclef st andrew's collection.* Paper presented at the Proceedings of the CLEF 2004: Workshop on Cross-Language Information Retrieval and Evaluation, Bath, UK.

Karlsen, R., & Nordbotten, J. (2005). *Project proposal caim - context aware image management*. Tromsø, Bergen: Dept. of Computer Science, University of Tromsø. Dept. of Information and Media Science, University of Bergen.

Kjørup, S. (1978). Pictorial speech acts. In *Erkenntnis 12* (pp. 55-71): Springer Netherlands.

Kretser, O., Moffat, A., Shimmin, T., & Zobel, J. (1998). Methodologies for distributed information retrieval, *Proceedings of the The 18th International Conference on Distributed Computing Systems*.

Li, Y., & Kuo, C. C. J. (2002). Introduction to content-based image retrieval - overview of key techniques. In V. Castelli & L. D. Bergman (Eds.), *Image databases: Search and retrieval of digital imagery* (pp. 261-284.): John Wiley & Sons, Inc.

Litwin, W., & Abdellatif, A. (1986). Multidatabase interoperability. *IEEE Computer, 19*(12), 10 - 18.

Losee, R. M., & Church, L. (2004). Information retrieval with distributed databases: Analytic models of performance. *Ieee Transactions on Parallel and Distributed Systems, 15*(1), 18-27.

Lu, G. (1999). *Multimedia database management systems*. Boston: Artech House.

Markkula, M., & Sormunen, E. (2000). End-user searching challenges indexing practices inthe digital newspaper photo archive. *Information Retrieval, Volume 1*(Issue 4), 259 - 285.

McDonald, K., & Smeaton, A. F. (2005). A comparison of score, rank and probability-based fusion methods for video shot retrieval., *CIVR 2005 - International Conference on Image and Video Retrieval*. Singapore, 20-22 July.

Missier, P., Rusinkiewicz, M., & Jin, W. (1999). Multidatabase languages. In A. Elmagarmid, M. Rusinkiewicz & A. Sheth (Eds.), *Management of heterogeneous and autonomous database systems*. San Francisco, CA: Morgan Kaufmann.

Montague, M., & Aslam, J. A. (2001). Relevance score normalization for metasearch, *Proceedings of the tenth international conference on Information and knowledge management*. Atlanta, Georgia, USA: ACM Press.

Müller, H., Ruch, P., & Geissbuhler, A. (2005). Enriching content-based image retrieval with multi-lingual search terms. *Swiss Medical Informatics, volume 54*, 6-11.

Nordbotten, J. C. (2006). ADM - Advanced Data Management - Now: Multimedia Information Retrieval Systems. Retrieved March 30th, 2006, from http://nordbotten.com/ADM/ADM_book/

Orphanoudakis, S. C., Chronaki, C. E., & Vamvaka, D. (1996). I cnet: Content--based similarity search in geographically distributed repositories of medical images. *Computerized Medical Imaging and Graphics, 20*(4), 193 - 207.

Ringdal, K. (2001). *Enhet og mangfold: Samfunnsvitenskapelig forskning og kvantitativ metode*. Bergen: Fagbokforlaget.

Rui, Y., Huang, T. S., & Chang, S. F. (1999). Image retrieval: Current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation, 10*(4), 39 - 62.

Rønnevik, H. (2005). *Rangering av bilder fremhentet fra en HTML dokumentsamling.* Universitetet i Bergen, Bergen.

Schilit, B. N., Adams, N. I., & Want, R. (1994). *Context-aware computing applications.* Paper presented at the Workshop on Mobile Computing Systems and Applications, Santa Cruz, CA.

Schwebs, T., & Østbye, H. (1999). *Media i samfunnet* (4. utg. ed.). Oslo: Samlaget.

Sclaroff, S., Cascia, M. L., & Sethi, S. (1999). Unifying textual and visual cues for content-based image retrieval on the world wide web. *CVIU, 75*(1-2), 86 - 98.

Steidinger, A. (2000). *Comparison of different collection fusion models in distributed information retrieval.* Paper presented at the Proceedings of the First DELOS Network of Excellence Workshop on

"Information Seeking, Searching and Querying in Digital Libraries".

Stolze, K. (2006). How to pass user input to a store procedure. Retrieved December 6th, 2006, from http://www-128.ibm.com/developerworks/forums/dw_thread.jsp?message=13882357&cat=19&thread=139935&treeDisplayType=threadmode1&forum=291#13882357

van Rijsbergen, C. J. (1999). Information retrieval. Retrieved January 12th, 2007

Vinsvold, R. (2005). *En evaluering av bildefremhenting i or-dbms/image.* Universitetet i Bergen, Bergen.

Voorhees, E. M., Gupta, N. K., & Johnson- Laird, B. (1994). *The collection fusion problem.* Paper presented at the TREC-3, NIST Special Publ. 500- 225F.

Wang, L., Fan, W., Yang, R., Xi, W., Luo, M., Zhou, Y., et al. (2003). Ranking function discovery by genetic programming for robust retrieval., *The Twelfth Text Retrieval Conference* (pp. 828-836). Gaithersburg: NIST.

Westerveld, T. (2000). *Image retrieval: Content versus context.* Paper presented at the RIAO 2000 Conference Proceedings, Paris.

Wu, S., Crestani, F., & F, G. (2003). New methods of results merging for distributed information retrieval. In J. P. Callan, F. Crestani & M. Sanderson (Eds.), *Distributed multimedia information retrieval: Sigir 2003 workshop on distributed information retrieval*. Berlin: Springer.

Xiangyu, J., & James, C. F. (2003). Improving image retrieval effectiveness via multiple queries, *Proceedings of the 1st ACM international workshop on Multimedia databases*. New Orleans, LA, USA: ACM Press.

Yu, C., & Meng, W. (2003). Web search technology. In H. Bidgoli (Ed.), *The internet encyclopedia* (pp. pp.738-753): Wiley Publishers.

Zhao, R., & Grosky, W. (2001). Bridging the semantic gap in image retrieval. In T. K. Shith (Ed.), *Distributed multimedia databases: Techniques and applications.* Pennsylvania: Idea Group Publising.

Zhou, X. S., & Huang, T. S. (2002). Unifying keywords and visual contents in image retrieval. *Ieee Multimedia, 9*(2), 23-32.

Øhrn, A. (2005). *Contextual insight in search: Enabling technologies and applications.* Paper presented at the Proceedings of the 31st international conference on Very large databases, Trondheim, Norway.

THE FACULTY OF SOCIAL SCIENCES,

DEPARTMENT OF INFORMATION SCIENCE AND MEDIA STUDIES

Utilizing context in ranking results from distributed image retrieval – the CAIRANK Prototype

APPENDICES

Appendix A – Glossary and List of Definitions

This appendix contains definitions, central terms, and acronyms used in the master's thesis are compiled here in alphabetical order, with the definition number in parentheses behind the main term.

Binary Large Object (BLOB) - is a term for a binary file stored as part of a database record.

Character Large Object (CLOB) - is a term for a large file of characters stored as part of a database record.

Consortium - is an agreement, combination, or group (as of companies) formed to undertake an enterprise beyond the resources of any one member (m-w.com/dictionary/consortium).

Context-aware computing (26) - [is when a] system uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task (Dey et al., 1999).

Context Aware Image Ranking (CAIRANK) prototype - is the prototype developed as a method to investigate the research question and hypothesis put forth in this project.

Content Based Image Retrieval (CBIR) (23) - is the process of retrieving images based on low-level features automatically extracted from images for retrieval purposes.

Context (13) - is any information that can be used to characterize the situation of an entity (Dey et al., 1999).

Data (1) - are symbols inscribed in formalized patterns, representing facts, observations and/or ideas, that are capable of being communicated, interpreted and manipulated by some human or mechanized process (Nordbotten 2006).

Data items (17) - are the elements forming the total collection of catalogued data possible to locate and display upon request.

Database (14) - A database is a logically coherent collection of related data, representing some aspect of the real world, designed, built, and populated for some purpose (Nordbotten 2006).

Data Management System, DMS - is a generic term for a software system for data management including: data definition, storage management, data retrieval, transaction (update) management, and security & integrity control (Nordbotten 2006).

Database Management System (15) - A Database Management System, DBMS, is a system providing 1) a schema defining the structure used for the data that represents the information in a database, 2) a database engine that supports storage, access to and modification of the database, 3) a language for definition and manipulation of the database (Adapted from Hove 2004).

Database system (16) - A Database System, DBS, is an information processing system containing: a DB and DBMS, a number of DB applications, and well as ad hoc user interactions (Nordbotten 2006).

Database Weight - is the output from a series of test queries preformed to determine the effectiveness of each participating DBS.

Digital image (7) - A digital image is a photograph or graphic, composed of discrete pixels of digitally quantized brightness and colour, created or rendered on a computer from an ultimate input source such as a digital camera or a scan of an image.

Distributed information retrieval (27) - is the use of multiple database systems, residing on one or more computers connected by a network to process a single information request.

Document (8) - A document is a representation of a unit of information, and may consist of plain or formatted text, images, inline graphics, sound, other multimedia data, and/or hyperlinks to other documents.

Document similarity score (22) - A document similarity score is a numerical score assigned by the DBMS to each query-document pair in the collection, indicating how well the documents meets an information need specified through a query according to evaluation criteria implemented in the DBMS.

Feature descriptors (20) - are descriptors generated by the DBMS, capturing the specific visual characteristics in an image.

Feature extraction (19) - is the process of extracting structural data from a digital image that is then used by the DBMS to classifying the syntactical image content.

Feature vector (21) - A feature vector is a set of descriptors describing one or more syntactical image features, represented as a binary string (Hove 2004).

Heterogeneous Database systems - are database systems where the software used to create and manipulate data in one site differs from that used at the other sites, and where the underlying data models differ, i.e. follow different structure and format.

Image (7) - An image is a visual representation of an entity or entities, produced on a medium.

Image feature (11) - An image feature is a distinguishing primitive characteristics or attribute of an image (W. Pratt in Berman, Castelli et al. 1997).

Image similarity score (24) - An image similarity score is the output from calculating the distance between the image signature of images in the collection and the image signature of a seed image.

Index (18) - An index is a data structure constructed from the data items to speed up searching and retrieval. The structure consists of terms used to refer to the content of a data item.

Information (2) - is the meaning that a human extracts from data by means of known conventions of the representation used (Gould, in Nordbotten 2006).

Information Retrieval System, IRS – [is] a data management system, DMS, for locating bibliographic and/or full text documents from one or more databases (Nordbotten 2006).

Interleaving (28) - is the placement of returned query results in a presentation list in a notionally sequential manner, always selecting the next top item from each returned list of query results.

Java Database Connectivity (JDBC) - is a Sun Microsystems standard for database-independent connectivity between the Java platform and a wide range of databases. The JDBC interface provides a call-level API for SQL-based database access.

Knowledge (3) - is the fact or condition of being aware of, or knowing, something with familiarity gained through experience or association, by acquaintance with or understanding of, a science, art, or technique, thus apprehending truth or fact through reasoning.

Media data (4) - is digital data used to record the information presented in a particular type of media object, f. ex. text, image, sound, or tables (of alphanumeric data) (Nordbotten 2006).

Min-Max Normalization - is a type of normalization that transforms the data of an initial range into a desired range, usually [0,1].

Non-visual image content (12) - corresponds to information that is closely related to the image, but that is not necessarily explicitly given by its appearance.

Normalization (29) - is a mathematical process that adjusts for differences among data from varying sources in order to create a common basis for comparison.

Query - An information request, formulated in the query language of a DMS (Nordbotten 2006).

Relevance (25) - refers to what extent a data item contains the semantic properties needed to satisfy the information need of a user for a given query.

Round Robin (RR) – is a ranking model that always removes the first elements of the result lists in a round robin fashion and puts them into the global result list, proceeding until all elements are merged.

Semantic image content (10) - is the meaning given to both the visual elements perceived in an image and the way in which they are arranged.

Structural Semantic Model (SSM) - is an extension and graphic simplification of the Extended Entity Relationship (EER) modelling tool.

Syntactic image content (9) - is the spatial arrangement of characteristics, like colour, shape and texture, associated with the visual elements perceived in an image.

Text (5) - is the vehicle of a communicative act when expressing something using written words in accordance with grammar.

User Defined Function (UDF) - are external or SQL functions which can be called from a SQL server.

User Defined Types (UDT) - are base data types explicitly defined with a new name through the pre-compiler's type definition directive. They are equivalent to the base types from which they were defined.

Weighted merging (30) - is an uneven interleaving, biased by the expected relevance of the collection to the query.

Appendix B – PL/SQL and SQL/PL Code

This appendix contains the complete code used to implement the DBS' used with the interface of the CAIRANK prototype. TheDBS' used with the prototype was written in Oracle 9i and IBM UDB DB2 v 8.2. The programming languages used, was the Procedural Language / SQL, or PL/SQL, and SQL / Procedural Language, or SQL/PL. The code is not commented in any other way than through comments inserted directly in the procedures.

Executing procedure for inserting seed images into Oracle:

```
execute insert_seedimage(1,'Stord Bridge', '1.bmp');
execute insert_seedimage(2,'Old Lisbon Bridge', '2.bmp');
execute insert_seedimage(3,'Yangluo Bridge', '3.bmp');
execute insert_seedimage(4,'Akashi Kaikyo Bridge', '4.bmp');
execute insert_seedimage(5,'Lions Gate Bridge', '5.bmp');
execute insert_seedimage(6,'Ambassador Bridge', '6.bmp');
execute insert_seedimage(7,'Semipalatinsk Bridge', '7.bmp');
execute insert_seedimage(8,'Great Belt Bridge', '8.bmp');
execute insert_seedimage(9,'Kvalsund Bridge', '9.bmp');
execute insert_seedimage(10,'San Francisco Bay Bridge', '10.bmp');
execute insert_seedimage(11,'Golden Gate Bridge', '11.bmp');
execute insert_seedimage(12,'Severn Bridge', '12.bmp');
```

Oracle database code:

```
***Access to images and documents***

CREATE OR REPLACE DIRECTORY C_BILDEDIR AS 'C:\DATABASECONTENT\Images';
GRANT READ ON DIRECTORY C_BILDEDIR TO PUBLIC WITH GRANT OPTION;

CREATE OR REPLACE DIRECTORY C_DOCDIR AS 'C:\DATABASECONTENT\ODocs';
GRANT READ ON DIRECTORY C_DOCDIR TO PUBLIC WITH GRANT OPTION;

***Constructing the Image Types***

CREATE OR REPLACE TYPE image AS OBJECT(
imageID number(4),
imageTitle varchar2(50),
image ORDSYS.ORDImage,
imageSignature ORDSYS.ORDImageSignature
)NOT FINAL;

CREATE OR REPLACE TYPE seedimage AS OBJECT(
seedimageID number(4),
seedimageTitle varchar2(50),
seedimage ORDSYS.ORDImage,
seedimageSignature ORDSYS.ORDImageSignature
)NOT FINAL;

***Constructing Image Tables***

CREATE TABLE image_tab OF image(
CONSTRAINT image_pk PRIMARY KEY(imageID))
;
```

```
CREATE TABLE seedimage_tab OF seedimage(
CONSTRAINT seedimage_pk PRIMARY KEY(seedimageID))
;

***Constructing the Document Type***

CREATE OR REPLACE TYPE document AS OBJECT(
documentID number(4),
documentTitle varchar2(100),
document clob
);

***Constructing the Document Table***

CREATE TABLE document_tab OF document(
CONSTRAINT document_pk PRIMARY KEY(documentID)
);

***Constructing the Described_by/Describes Relationship Table***

CREATE TABLE described_by_describes(
image_described number(4) NOT NULL,
document_describes number(4) NOT NULL,
CONSTRAINT PK_described_by_describes PRIMARY KEY(image_described, document_describes),
CONSTRAINT FK_described FOREIGN KEY(image_described) REFERENCES image_tab,
CONSTRAINT FK_describes FOREIGN KEY(document_describes) REFERENCES document_tab
);

***Procedures for inserts into image_tab***

SET SERVEROUTPUT ON
SET ECHO ON

CREATE OR REPLACE PROCEDURE insert_image(Imgid number,Imgtitle varchar2,Imgfilename varchar2)
IS
Image ORDSYS.ORDImage;
Image_sig ORDSYS.ORDImageSignature;
ctx RAW(4000) := NULL;
BEGIN
INSERT INTO image_tab values(Imgid, Imgtitle, ORDSYS.ORDImage.init(),
ORDSYS.ORDImageSignature.init());

     SELECT s.image, s.imageSignature INTO Image, Image_sig FROM image_tab s
        WHERE s.imageID = Imgid for UPDATE;

   Image.setSource('file','C_BILDEDIR',Imgfilename);
   Image.import(ctx);
   Image_sig.GenerateSignature(Image);
   Image.setProperties;

  UPDATE image_tab s SET s.image = Image, s.imageSignature = Image_sig WHERE s.imageID = ImgID;

  COMMIT;
END;

***Procedure for seedimage***

CREATE OR REPLACE PROCEDURE insert_seedimage(seedImgid number,seedImgtitle
varchar2,seedImgfilename varchar2)
```

```
IS
seedImage ORDSYS.ORDImage;
seedImage_sig ORDSYS.ORDImageSignature;
ctx RAW(4000) := NULL;
BEGIN
INSERT INTO seedimage_tab values(seedImgid, seedImgtitle, ORDSYS.ORDImage.init(),
ORDSYS.ORDImageSignature.init());

     SELECT s.seedimage, s.seedimageSignature INTO seedImage, seedImage_sig FROM seedimage_tab s
        WHERE s.seedimageID = seedImgid for UPDATE;

   seedImage.setSource('file','C_BILDEDIR',seedImgfilename);
   seedImage.import(ctx);
   seedImage_sig.GenerateSignature(seedImage);
   seedImage.setProperties;

  UPDATE seedimage_tab s SET s.seedimage = seedImage, s.seedimageSignature = seedImage_sig WHERE
s.seedimageID = seedImgID;

  COMMIT;
END;

***Inserts into image_tab***

execute insert_image(ID,Title, Image);

***Procedure for inserts into d_document_tab and the Described_by/Describes Relationship Table***

CREATE OR REPLACE PROCEDURE insert_document(Imgid number,Docid number,Doctitle varchar2,
Docfilename varchar2)
IS
f_lob   bfile;
b_lob   clob;
BEGIN
INSERT INTO document_tab VALUES (Docid,Doctitle,empty_clob())
RETURN document into b_lob;

   f_lob := bfilename( 'C_DOCDIR', Docfilename );
   dbms_lob.fileopen(f_lob, dbms_lob.file_readonly);
   dbms_lob.loadfromfile
  ( b_lob, f_lob, dbms_lob.getlength(f_lob) );
   dbms_lob.fileclose(f_lob);

INSERT INTO described_by_describes VALUES(Imgid,Docid);

   COMMIT;
END;

***Inserts into document_tab***

execute insert_document(imageID,docID,documentTitle,documentFilename);

***LIST OUT DOCUMENT ID***
select documentID from document_tab
order by documentID;

***Procedure for querying images based on similarity***
ALTER SESSION SET NLS_LANGUAGE=ENGLISH;
alter session set NLS_TIMESTAMP_FORMAT = 'YYYY-MM-DD HH:MI:SS.FF';
SET SERVEROUTPUT ON SIZE 1000000;
```

```
SET ECHO ON;
SET LINES 32767;

***Constructing the Temporary Result Table***

CREATE TYPE result_type AS OBJECT(
search_date timestamp(2),
image_no number(10),
sim_score decimal(10,5),
img_title varchar2(50),
doc_no varchar2(10),
doc_score decimal(10,5),
doc_title varchar2(50),
norm_image varchar2(10),
norm_text varchar2(10)
);

***MANIPULATING TESTTABLE***
CREATE TABLE result_tab OF result_type(
CONSTRAINT PK_result PRIMARY KEY(search_date,image_no)
);
drop TABLE result_tab Force;
select * from result_tab;
OR
select image_no from result_tab
order by sim_score desc;

***CREATING INDEX***

***CONTAINS***
--DATASTORE
--INSO FILTER
--LEXER
-- WORDLIST
--STOPLIST

***LEXER PREFERENCE***

begin
ctx_ddl.create_preference('C_doklex', 'BASIC_LEXER');
ctx_ddl.set_attribute('C_doklex', 'printjoins', '_-');
ctx_ddl.set_attribute('C_doklex', 'continuation', '-\');
ctx_ddl.set_attribute ( 'C_doklex', 'index_themes', 'YES');
ctx_ddl.set_attribute ( 'C_doklex', 'index_text', 'YES');
ctx_ddl.set_attribute ( 'C_doklex', 'prove_themes', 'YES');
end;

***WORDLIST PREFERENCE***

begin
ctx_ddl.create_preference('C_ordliste', 'BASIC_WORDLIST');
ctx_ddl.set_attribute('C_ordliste','SUBSTRING_INDEX', 'YES');
ctx_ddl.set_attribute('C_ordliste','FUZZY_MATCH','ENGLISH');
ctx_ddl.set_attribute('C_ordliste','FUZZY_SCORE','0');
ctx_ddl.set_attribute('C_ordliste','FUZZY_NUMRESULTS','5000');
ctx_ddl.set_attribute('C_ordliste','SUBSTRING_INDEX','TRUE');
ctx_ddl.set_attribute('C_ordliste','STEMMER','ENGLISH');
end;
```

```
****ALTER SYSTEM SETTINGS WHEN USING CBIR/TEXT-BASED QUERIES*****
ALTER SESSION SET NLS_LANGUAGE=ENGLISH;
alter session set NLS_TIMESTAMP_FORMAT = 'YYYY-MM-DD HH:MI:SS.FF';
SET SERVEROUTPUT ON SIZE 1000000;
SET ECHO ON;
SET LINES 32767;

--***PROCEDURES***

CREATE OR REPLACE procedure CBIRQuery(queryseedimage number, threshold number)is
-------------------------------------------------------------------------------------------
-- This procedure will take the number of an image in the table seedImg_tab, compare it to
-- the image collection, return a list of images with a certain degree of similarity,
-- and populate a temporary table with the results as well as the context information
-- associated with retrieved images
-------------------------------------------------------------------------------------------

weighting varchar2(256):='color="0,5"texture="0,5" shape="0,0" location="0,0"';
compare_sig ORDSYS.ORDImageSignature;
imgnr number ;
sim_score decimal(10,5);
docname varchar2(50);
title varchar2(50);
search_time timestamp;
min_score decimal(10,5);
max_score decimal(10,5);
norm_score decimal(10,5);
rdoc_no number;
imgno number;
imgscore decimal(10,5);


Cursor getphoto IS
select b.imageID,ORDSYS.IMGScore(123) score, b.imageTitle
from image_tab b
where ORDSYS.IMGSimilar(b.imageSignature,compare_sig,weighting,threshold,123) = 1
ORDER BY SCORE DESC;

CURSOR norm_imagescores IS
Select image_no, sim_score from result_tab
ORDER BY sim_score ASC;

begin

select systimestamp into search_time from dual;
select p.seedimageSignature into compare_sig
from seedimage_tab p
where p.seedimageID = queryseedimage;
open getphoto;
loop
FETCH getphoto into imgnr,sim_score,title;
exit when getphoto%NOTFOUND;

select r.documentID, r.documentTitle into rdoc_no,docname
from document_tab r, described_by_describes
where document_describes = imgnr and image_described = documentID;
INSERT INTO result_tab
VALUES(search_time,imgnr,sim_score,title,rdoc_no,null,docname,null,null);
end loop;
close getphoto;
```

```
select max(sim_score) into max_score from result_tab;
select min(sim_score) into min_score from result_tab;

OPEN norm_imagescores;
LOOP
FETCH norm_imagescores INTO imgno,imgscore;
EXIT WHEN norm_imagescores%NOTFOUND;

norm_score := 1 - (imgscore-min_score)/(max_score-min_score);

UPDATE result_tab
SET norm_image = norm_score WHERE image_no = imgno;
end loop;
close norm_imagescores;

end CBIRQuery;

--***INDEXING THE TEXT TABLE***

create index Oraindex on document_tab(document)
indextype is ctxsys.context parameters ('DATASTORE CTXSYS.DEFAULT_DATASTORE FILTER
CTXSYS.INSO_FILTER LEXER C_doklex WORDLIST C_ordliste STOPLIST
CTXSYS.DEFAULT_STOPLIST');

--***PROCEDURE FOR CLEARING THE TEMPORARY TABLE***

CREATE procedure clearTable is
-------------------------------------------------------------------------------------
-- This procedure will clear the result table (result_tab), prepearing it for the next query
-------------------------------------------------------------------------------------
begin
execute immediate 'DELETE FROM result_tab';
end clearTable;

--***PROCEDURES FOR QUERYING TEXT DOCUMENTS***
-------------------------------------------------------------------------------------
-- This procedure will take the keywords provided by the user, and query the document
-- collection (document_tab) using a thematic query. Then the temporary table is updated with
-- the relevance scores given by the DBMS.
-------------------------------------------------------------------------------------

CREATE OR REPLACE procedure docQuery(keyword1 in varchar2) is
docname varchar2(50);
docnr number ;
rel_score decimal(10,5);
--kword1 varchar2(100):= keyword1;
min_score decimal(10,5);
max_score decimal(10,5);
norm_score decimal(10,5);
docno number;
docscore decimal(10,5);

Cursor getdoc IS
SELECT SCORE(1),a.documentID
FROM document_tab a
WHERE CONTAINS(document, 'about({'||keyword1||'})', 1) >= 0
ORDER BY SCORE(1) DESC;

CURSOR norm_docscores IS
```

```
Select doc_no, doc_score from result_tab
ORDER BY doc_score ASC;

begin

open getdoc;
loop
FETCH getdoc into rel_score, docnr;
exit when getdoc%NOTFOUND;
UPDATE result_tab
SET doc_score = rel_score
WHERE doc_no=docnr;
end loop;
close getdoc;

select max(doc_score) into max_score from result_tab;
select min(doc_score) into min_score from result_tab;

OPEN norm_docscores;
LOOP
FETCH norm_docscores INTO docno,docscore;
EXIT WHEN norm_docscores%NOTFOUND;

norm_score :=(docscore-min_score)/(max_score-min_score);
UPDATE result_tab
SET norm_text = norm_score WHERE doc_no = docno;
end loop;
close norm_docscores;

end DOCQuery;
```

IBM database code:

```
***KOBLE TIL DATABASEN***
C:\Programfiler\IBM\SQLLIB\BIN>start dmbssd

C:\Programfiler\IBM\SQLLIB\BIN>db2ext
db2ext => connect to ibm user ch using ******

***DB2EXT Kommandolinjevindu***

db2ext => enable database for db2image

DMB0027I: The current database is enabled for extender "db2image".

***LAGE TABELL FOR BILDER***
***DB2 Kommandolinjevindu***

--Etter at db2image er enabled

create table image_tab(imageID integer NOT NULL PRIMARY KEY,imageTitle varchar(50), bilde
mmdbsys.db2image)


***SETTE INN I BILDETABELL***

insert into image_tab values(<ID>,<TITLE>,<mmdbsys.db2image(current
server,'c:\DATABASECONTENT\Images\<FILENAME>','<EXTENTION>',1, '<NAME>'))
```

```
***LAGE QBIC KATALOG***
***DB2EXT Kommandolinjevindu***

db2ext => connect to ibm user ch using [*******]
   Database Connection Information

Database server      = DB2/NT 8.2.3
Local database alias = IBM

db2ext => enable database for db2image
DMB0027I: The current database is enabled for extender "db2image".

db2ext => enable table image_tab for db2image
DMB0028I: Table "image_tab" is enabled for extender "db2image".

db2ext => enable column image_tab bilde for db2image
DMB0029I: Column "bilde" in table "image_tab" is enabled for extender "db2image".

db2ext => create qbic catalog image_tab bilde on
DMB0055I: The "CREATE QBIC CATALOG" command completed successfully.

db2ext => open qbic catalog image_tab bilde
DMB0055I: The "OPEN QBIC CATALOG" command completed successfully.

db2ext => set qbic autocatalog on
DMB0055I: The "SET QBIC AUTOCATALOG" command completed successfully.

db2ext => add qbic feature QbColorFeatureClass
DMB0055I: The "ADD QBIC FEATURE" command completed successfully.

db2ext => add qbic feature QbColorHistogramFeatureClass
DMB0055I: The "ADD QBIC FEATURE" command completed successfully.

db2ext => add qbic feature QbDrawFeatureClass
DMB0055I: The "ADD QBIC FEATURE" command completed successfully.

db2ext => add qbic feature QbTextureFeatureClass
DMB0055I: The "ADD QBIC FEATURE" command completed successfully.

db2ext => close qbic catalog
DMB0055I: The "CLOSE QBIC CATALOG" command completed successfully.

db2ext => get extender status
EXTENDER TABLESPACE                    TABLE
-------- ------------------------------ --------------------------
DB2IMAGE USERSPACE1,USERSPACE1,USERSPACE1 CH.TEST MMDBSYS.QBICTEMP
DMB0024I: The current database is enabled for "1" extenders.

db2ext => get qbic catalog info
In the QBIC catalog for column BILDE table CH.TEST, auto-cataloging has been set to ON.
There are 4 features: QbColorFeatureClass QbColorHistogramFeatureClass QbDrawFeatureClass
QbTextureFeatureClass
DMB0055I: The "GET QBIC CATALOG INFO" command completed successfully.

db2ext => catalog qbic column
DMB0055I: The "CATALOG QBIC COLUMN FOR NEW" command completed successfully.


****CREATING DOCUMENT TABLE*****
```

```
CREATE TABLE db2ext.document_tab(documentID integer NOT NULL PRIMARY KEY, documentTitle
varchar(50), document Clob(1M))

****INSERTING DOCUMENTS*****-->Teksteditor (JEdit)

Create a .del file and use import statement to populate db.

****Populate the text database*****

import from document_tab.del of del lobs from idocs\ modified by lobsinfile insert into db2ext.document_tab

***IMPORT COMMAND***-->Kommandolinjevindu --> SAMPLES/DB2EXT

db2 => import from document_tab.del of del lobs from idocs\ modified by lobsinfi
le insert into DB2EXT.document_tab (documentID, documentTitle, document)

***UPDATE TABLE*** -->Kommandosenteret

update document_tab m
set(m.documentTitle)='Quebec Bridge'
where m.documentID = 582

***CONSTRUCTING THE Described_by/Describes RELATIONSHIP TABLE***

CREATE TABLE described_by_describes(
image_described integer NOT NULL,
document_describes integer NOT NULL,
CONSTRAINT PK_des_by_descr PRIMARY KEY(image_described, document_describes),
CONSTRAINT FK_described FOREIGN KEY(image_described) REFERENCES image_tab,
CONSTRAINT FK_describes FOREIGN KEY(document_describes) REFERENCES db2ext.document_tab
)

***Constructing the Temporary Result Table***

CREATE TABLE result_tab(
image_no integer NOT NULL PRIMARY KEY,
sim_score decimal(10,5),
img_title varchar(50),
doc_no integer,
doc_score decimal(10,5),
doc_title varchar(50),
norm_image decimal(10,5),
norm_text decimal(10,5)
)

***CREATING THE CONTENT BASED IMAGE QUERY***

CREATE procedure CBIRQuery(IN imagename VARCHAR(255))

LANGUAGE SQL

BEGIN

DECLARE imgnr integer;
DECLARE sim_score decimal(10,5);
DECLARE docname varchar(50);
DECLARE new_sim_score decimal(10,5);
DECLARE title varchar(50);
DECLARE rdoc_no integer;
DECLARE SQLSTATE CHAR(5);
```

```
DECLARE filename VARCHAR(255);
DECLARE stmt VARCHAR(500);
DECLARE min_score decimal(10,5);
DECLARE max_score decimal(10,5);
DECLARE norm_score decimal(10,5);
DECLARE imgno integer;
DECLARE imgscore decimal(10,5);

DECLARE norm_imagescores CURSOR FOR
Select image_no, sim_score from result_tab
ORDER BY doc_score ASC;

SET filename = '<server,C:\DATABASECONTENT\Images\'||imagename||'.bmp>';

SET stmt = 'SELECT a.imageID, decimal(mmdbsys.qbscorefromstr(' ||
'bilde,''texture file=' || filename ||
' and histogram file=' || filename || ' and averagecolor file=' || filename ||
' and draw file=' || filename || '''), 10, 5)' ||
'as score, a.imageTitle FROM image_tab a ORDER BY score DESC';

PREPARE selectStmt FROM stmt;
BEGIN
DECLARE getPhoto CURSOR FOR selectStmt;
OPEN getPhoto;

L1: LOOP
FETCH getphoto INTO imgnr,sim_score,title;
IF SQLSTATE = '02000' THEN LEAVE L1; END IF;

select r.documentID, r.documentTitle into rdoc_no,docname
  from db2ext.document_tab r, described_by_describes
  where document_describes = imgnr and image_described = documentID;
  INSERT INTO result_tab
VALUES(imgnr,sim_score,title,rdoc_no,null,docname,null,null);
END LOOP L1;
CLOSE getphoto;
END;

select max(sim_score) into max_score from result_tab;
select min(sim_score) into min_score from result_tab;
OPEN norm_imagescores;

L2: LOOP
FETCH norm_imagescores INTO imgno,imgscore;
IF SQLSTATE = '02000' THEN LEAVE L2; END IF;

SET norm_score = 1 - (imgscore-min_score)/(max_score-min_score);

UPDATE result_tab
SET norm_image = norm_score WHERE image_no = imgno;
end loop L2;
close norm_imagescores;
END

***CALLING THE STORED CBIRQuery PROCEDURE****
CALL CBIRQuery('AskoyBridge1')

***DOCQUERY PROCEDURE***
CREATE procedure docQuery(IN keyword1 VARCHAR(255))
```

```
LANGUAGE SQL

BEGIN

DECLARE docnr integer;
DECLARE rel_score decimal(10,5);
DECLARE min_score decimal(10,5);
DECLARE max_score decimal(10,5);
DECLARE norm_score decimal(10,5);
DECLARE docno integer;
DECLARE docscore decimal(10,5);
DECLARE stmt VARCHAR(510);
DECLARE SQLSTATE CHAR(5);
DECLARE norm_docscores CURSOR FOR
Select doc_no, doc_score from result_tab
ORDER BY doc_score ASC;

SET stmt = 'SELECT a.documentID, SCORE(document,''' || 'IS ABOUT EN_US '''||keyword1||''''' || ''') FROM
DB2EXT.document_tab a';

PREPARE selectStmt FROM stmt;
BEGIN
DECLARE getDoc CURSOR FOR selectStmt;
OPEN getDoc;

L1: LOOP
FETCH getDoc INTO docnr,rel_score;
IF SQLSTATE = '02000' THEN LEAVE L1; END IF;

UPDATE result_tab
SET doc_score = rel_score
WHERE doc_no=docnr;
END LOOP L1;
CLOSE getDoc;
END;

select max(doc_score) into max_score from result_tab;
select min(doc_score) into min_score from result_tab;

OPEN norm_docscores;
L2: LOOP
FETCH norm_docscores INTO docno,docscore;
IF SQLSTATE = '02000' THEN LEAVE L2; END IF;

SET norm_score = (docscore-min_score)/(max_score-min_score);

UPDATE result_tab
SET norm_text = norm_score WHERE doc_no = docno;
end loop L2;
close norm_docscores;
END

--***PROCEDURE FOR CLEARING THE TEMPORARY TABLE***
CREATE procedure clearTable
LANGUAGE SQL
BEGIN
DELETE FROM result_tab;
end

***COMBINATION QUERY ON image_tab***
```

```
SELECT imageTitle,
decimal(mmdbsys.qbscorefromstr(bilde,'averagecolor
file=<server,C:\DATABASECONTENT\Images\AskoyBridge1.bmp>
and histogram file=<server,C:\DATABASECONTENT\Images\AskoyBridge1.bmp>
and draw file=<server,C:\DATABASECONTENT\Images\AskoyBridge1.bmp>
and texture file=<server,C:\DATABASECONTENT\Images\AskoyBridge1.bmp>'),10,5)
AS score FROM image_tab
ORDER BY score
```

Appendix C – Java Code

This appendix contains the complete code used to implement this first version of the CAIRANK search engine and the functionality to submit and retrieve queries from the participating DBS'. These parts of the prototype were written in the Java programming language. The code is not commented in any other way than through comments inserted directly in the classes.

DbConnection class

```
/**
 * Class containing methods for creating connections to
 * the participating DBS'.
 * @author Christian Hartvedt
 */
package distributedIR;

import java.sql.*;
import java.util.*;
import oracle.jdbc.*;
import java.io.*;
import oracle.jdbc.driver.*;
import oracle.jdbc.driver.OracleResultSet;
import java.lang.Object;
import distributedIR.DbConnection;

public class DbConnection {
private static Connection Ocon = null;
private static Connection Icon = null;

/**
 * Constructor for IBM
 * Calls to connect to the IBM DBS
 * @param DBNavn the database name used for access
 */
public static Connection connect(String DBNavn)throws SQLException{
        if(Icon!=null){
        IBMdisConnect();
        } // end if
        try {
                Class.forName("COM.ibm.db2.jdbc.app.DB2Driver");
                Icon = DriverManager.getConnection("jdbc:db2:ibm","ch","*******");
                System.out.println("***Kontakt med IBM***");
        }//end try
        catch ( ClassNotFoundException cnfex) {
                System.out.println("Feilet med å laste IBM driveren: " + cnfex.getMessage());
                System.exit(1);
        }//end catch
        catch ( SQLException sqlex) {
                System.out.println("Tilkobling til IBM ikke mulig: " + sqlex.getMessage());
                System.exit(1);
        }//end catch
        return Icon;
} // end connect()

/**
 * Constructor for Oracle
```

```
 * Calls to connect to the Oracle DBS
 * @param usr the username used for access
 * @param pass the password used for accsess
 * @param service the actual service used for accsess
 */
public static Connection connect(String usr, String pass, String service)throws SQLException{
if(Ocon!=null){
ORACLEdisConnect();
} // end if
try {
Class.forName("oracle.jdbc.driver.OracleDriver");
Ocon = DriverManager.getConnection( "jdbc:oracle:thin:@//localhost:1521/orakel", usr,pass);
Statement alter_date = Ocon.createStatement();
Statement alter_session = Ocon.createStatement();
alter_session.execute("alter session set NLS_LANGUAGE=ENGLISH");
alter_date.execute("alter session set NLS_DATE_FORMAT = 'dd.mm.yyyy'");
System.out.println("***Kontakt med Oracle***\n");
}//end try
catch ( ClassNotFoundException cnfex) {
System.out.println("Feilet med å laste Oracle driveren: " + cnfex.getMessage());
System.exit(1);
}//end catch
catch ( SQLException sqlex) {
System.out.println("Tilkobling til Oracle ikke mulig: " + sqlex.getMessage());
System.exit(1);
}//end catch
return Ocon;
} // end connect()

/*
 * Method that calls to disconnect the connection to the
 * IBM database
 */
public static void IBMdisConnect()throws SQLException{
        Icon.close();
        Icon = null;
} // end IBMdisConnect()
```

```
/*
 * Method that calls to disconnect the connection to the
 * Oracle database
 */
public static void ORACLEdisConnect()throws SQLException{
            Ocon.close();
            Ocon = null;
} // end ORACLEdisConnect()

/*
 * Method that establishes the connection to the
 * IBM database and returns it to the caller
 */
public static Connection getIbmCon(){
            return Icon;
} // end getIBMCon()

/*
 * Method that establishes the connection to the
 * Oracle database and returns it to the caller
 */
public static Connection getOracleCon(){
return Ocon;
} // end getOracleCon()

/*
 * Method used to check if the connection to IBM
 * is established
 */
public static boolean IbmisConnected(){
            return Icon != null;
} // end IbmisConnected()

/*
 * Method used to check if the connection to Oracle
 * is established
 */
public static boolean OracleisConnected(){
return Ocon != null;
} // end OracleisConnected()
} // end class DbConnection
```

StartPage class

```
/**
 * Class containing methods for creating the start
 * page of the CAIRANK prototype.
 * @author Christian Hartvedt
 */
package distributedIR;

import java.awt.event.*;
import javax.swing.*;
import java.awt.*;

public class StartPage extends JPanel{
private JLabel lblHeading=new JLabel("CAIRANK Prototype");
private JButton btnCAIRANKQuery = new JButton("CAIRANK Query");
private JButton btnExit = new JButton("Exit");
private JPanel panel = new JPanel();
private Font font;
Interface show_interface;
class Adapter implements ActionListener{
public void actionPerformed(ActionEvent event) {
Object object = event.getSource();
if (object==btnCAIRANKQuery){
show_interface = new Interface();
}//end if
else if (object==btnExit){
System.exit(0);
}//end else if
}//end actionPerformed()
}//end class Adapter

public StartPage() {
init_Menu();
}//end StartPage()

public void init_Menu(){
Adapter adapter = new Adapter();
font = new Font("Times New Roman", Font.BOLD, 25);
this.setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
btnCAIRANKQuery.addActionListener(adapter);
btnExit.addActionListener(adapter);
btnCAIRANKQuery.setMinimumSize(new Dimension(85, 12));
btnCAIRANKQuery.setMaximumSize(new Dimension(160, 28));
btnCAIRANKQuery.setPreferredSize(new Dimension(150, 20));
btnExit.setMinimumSize(new Dimension(85, 12));
btnExit.setMaximumSize(new Dimension(160, 28));
btnExit.setPreferredSize(new Dimension(150, 20));
lblHeading.setFont(font);
panel.add(Box.createRigidArea(new Dimension(250, 200)));
panel.add(lblHeading);
panel.add(Box.createRigidArea(new Dimension(150, 30)));
panel.add(btnCAIRANKQuery);
panel.add(Box.createRigidArea(new Dimension(0, 12)));
panel.add(btnExit);
this.add(panel);
}//end init_Menu()

public static void main (String args[]) {
```

```
JFrame f = new JFrame("Context Aware Image Ranking Prototype");
f.getContentPane().add(new StartPage());
f.setSize(1280,800);
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
f.show();
}//end main()
}//end class StartPage
```

Interface class

```
/**
 * Class containing methods for initiating a
 * search using the the CAIRANK prototype.
 * @author Christian Hartvedt
 */
package distributedIR;

import javax.swing.*;
import java.lang.*;
import java.awt.*;
import java.awt.event.*;

public class Interface extends JPanel{
private int imageNo;
JFrame frame = new JFrame("Context Aware Image Ranking Prototype");
RenderedImage image = null;
private JPanel panelNorth = new JPanel();
private JPanel panelCenter = new JPanel();
private JPanel panelSouth = new JPanel();
private JPanel informationItemList = new JPanel();
private JScrollPane informationItemListScrollPane = new JScrollPane(informationItemList);
private JPanel messagePane = new JPanel();
private JTextField textField0 = new JTextField();
private JTextField textField1 = new JTextField();
private JButton btnCAIRANKSearch = new JButton("Search");
private JButton btnClose = new JButton ("Close");
private JLabel lblQueryno=new JLabel("Query #");
private JLabel lblQuerytext=new JLabel("Query text");

Queryproc queryprocedure = new Queryproc(this);
class Adapter implements ActionListener{
public void actionPerformed (ActionEvent event){
Object object = event.getSource();
if (object == btnCAIRANKSearch){
try{
imageNo = Integer.parseInt(textField0.getText());
}//end try
catch (NumberFormatException e) {
  System.out.println("You have not entered a number " +" "+ e);
        queryprocedure.close();
}//end catch
if (textField1.getText().length() == 0){
        System.out.println("You have not entered query text");
        queryprocedure.close();
}//end if
else if (textField0.getText().length() == 0 || textField1.getText().length() == 0){
        System.out.println("Not all field are filled in");
        queryprocedure.close();
}//end else if
else {
queryprocedure.getResults(imageNo, textField1.getText());
```

```
}//end else
} // end if(object == btnCAIRANKSearch)
if (object == btnClose){
queryprocedure.close();
} // end else if
} // end actionPerformed()
} // end class Adapter

public Interface() {
frame.getContentPane().add(this);
init();
frame.setSize(1280,800);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.show();
} // end Interface();

public void init() {
Adapter adapter = new Adapter();
this.setLayout (new BoxLayout(this, BoxLayout.Y_AXIS));
panelNorth.setLayout(new BoxLayout(panelNorth, BoxLayout.X_AXIS));
panelCenter.setLayout(new CardLayout());
informationItemList.setLayout(new BoxLayout(informationItemList, BoxLayout.Y_AXIS));
panelSouth.setLayout(new CardLayout());
btnCAIRANKSearch.addActionListener(adapter);
btnClose.addActionListener(adapter);
textField0.setMinimumSize(new Dimension(20,16));
textField0.setMaximumSize(new Dimension(30,20));
textField0.setPreferredSize(new Dimension(25,18));
textField1.setMinimumSize(new Dimension(200,16));
textField1.setMaximumSize(new Dimension(300,20));
textField1.setPreferredSize(new Dimension(250,18));
panelNorth.add(Box.createRigidArea(new Dimension(250,0)));
panelNorth.add(lblQueryno);
panelNorth.add(Box.createRigidArea(new Dimension(5,50)));
panelNorth.add(textField0);
panelNorth.add(Box.createRigidArea(new Dimension(20,50)));
panelNorth.add(lblQuerytext);
panelNorth.add(Box.createRigidArea(new Dimension(5,20)));
panelNorth.add(textField1);
panelNorth.add(Box.createRigidArea(new Dimension(10,20)));
panelNorth.add(btnCAIRANKSearch);
panelNorth.add(Box.createHorizontalGlue());
panelNorth.add(btnClose);
panelCenter.add(messagePane, "message");
panelCenter.add(informationItemListScrollPane, "informationItemList");
this.add (panelNorth);
this.add (panelCenter);
} // end init();
}//end class Interface
```

Queryproc class

```
/**
 * Class containing methods for establishing connections to
 * the participating DBS', execute stored procedures, submit
 * queries and retrieve result sets from each database.
 * @author Christian Hartvedt
 */
package distributedIR;

import java.sql.*;
```

```
import java.text.DecimalFormat;
import java.util.*;
import oracle.jdbc.*;
import java.io.*;
import oracle.jdbc.driver.*;
import oracle.jdbc.driver.OracleResultSet;
import oracle.jdbc.driver.OracleTypes;
import java.lang.Object;
import java.math.BigDecimal;
import distributedIR.DbConnection;

public class Queryproc {
private int SQL_Stmt0;
private String SQL_Stmt1;
DbConnection dbConnection = new DbConnection();
private Interface myInterface;

/*
* Method that calls the connect method in order
* to establich a connection to the IBM and the
* Oracle databases.
*/
public Queryproc(Interface i) {
myInterface= i;
try{
        DbConnection.connect("ibm");
} // end try
catch (SQLException e){
e.printStackTrace();
e.getMessage();
System.err.println(e);
} // end catch
try{
DbConnection.connect("christian", "********", "orakel");
} // end try
catch (SQLException e){
e.printStackTrace();
e.getMessage();
System.err.println(e);
} // end catch
}// end Queryproc()

/**
* Methods taking in the text from the text field in the
* search engine interface and submit it to the participating
* DBS'
* @param sqlStatement0 is the query number
* @param sqlStatement1 is the text in the first text field in
* the search engine interface
* @param sqlStatement2 is the text in the second text field in
* the search engine interface
*/

public void getResults(int textField0, String textField1) {
        SQL_Stmt0 = textField0;
        SQL_Stmt1 = textField1;
try {
BufferedWriter out = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"CAIRANKResults.xls", true));
out.write("Query No:" +SQL_Stmt0+" \nImage No: \t Image Title: \tCAIRANK Score:\n");
```

```
out.close();

BufferedWriter out2 = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"RawScoreResults.xls", true));
out2.write("Query No:" +SQL_Stmt0+" \nImage No: \t Image Title: \tRaw Merge Score:\n");
out2.close();

}//end try
catch (IOException e) {
System.out.println(e);
}//end catch
try{
DbConnection.getIbmCon();
String Iimgquery=Integer.toString(SQL_Stmt0);
String idocquery = SQL_Stmt1;
String imagecall= new String("{ call CBIRQuery(?) }");
String doccall= new String("{ call docQuery(?) }");
CallableStatement IMGstmt = DbConnection.getIbmCon().prepareCall(imagecall);
IMGstmt.setString(1, (String) Iimgquery);
CallableStatement DOCstmt = DbConnection.getIbmCon().prepareCall(doccall);
DOCstmt.setString(1, (String) idocquery);
IMGstmt.execute();
IMGstmt.close();
DOCstmt.execute();
DOCstmt.close();
}//end try
catch(SQLException ex){
System.err.println("SQLException: " +ex.getMessage());
}//end catch

try{
DbConnection.getIbmCon();
String iquery= "SELECT distinct r.image_no,r.sim_score,r.norm_image, r.img_title,r.doc_score,r.norm_text
FROM result_tab r order by r.sim_score asc";
PreparedStatement pstmt = DbConnection.getIbmCon().prepareStatement(iquery);
ResultSet rs = (ResultSet)pstmt.executeQuery();
while (rs.next()){
int Iimagenumber =(rs.getInt(1));
String Iimagescore =rs.getString(2);
double Inorm_image = rs.getDouble(3);
String Iimagetitle =rs.getString(4).trim();
String Idocscore =(rs.getString(5));
double Inorm_text =rs.getDouble(6);
double ARaw_Score=Inorm_image;
String BRaw_Score = Double.toString(ARaw_Score );
String IRaw_Score = BRaw_Score.replace ( '.', ',');
double IscoresToComb= (0.468223394 * Inorm_image)+(0.5 * Inorm_text);
double Iglobalscore= 5.7506 * IscoresToComb;
String ItotalCombscore = Double.toString(Iglobalscore );
String Istring_score = ItotalCombscore.replace ( '.', ',');
System.out.print(Idocscore+" - ");
System.out.print(Inorm_text+" - ");
System.out.print(Istring_score+"\n");

try {
BufferedWriter out = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"CAIRANKResults.xls", true));
out.write(Iimagenumber+"\t"+Iimagetitle+ "\t"+Istring_score+"\n");
out.close();
```

```
BufferedWriter out2 = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"RawScoreResults.xls", true));
out2.write(Iimagenumber+"\t"+Iimagetitle+ "\t"+IRaw_Score+"\n");
out2.close();
}//end try
catch (IOException e) {
        System.out.println(e);
}//end catch
} // end while

System.out.print("\n"+"End of IBM results"+ "\n\n");
}//end try
catch(SQLException ex){
System.err.println("SQLException: " +ex.getMessage());
}//end catch

try{
DbConnection.getOracleCon();
String odocquery=SQL_Stmt1;
int threshold=100;
String querycall= new String("{ call docQuery(?) }");
String imagecall= new String("{ call CBIRQuery(?,?) }");
CallableStatement IMGstmt = DbConnection.getOracleCon().prepareCall(imagecall);
IMGstmt.setInt(1, SQL_Stmt0);
IMGstmt.setInt(2,threshold);
CallableStatement DOCstmt = DbConnection.getOracleCon().prepareCall(querycall);
DOCstmt.setString(1,odocquery);
IMGstmt.execute();
IMGstmt.close();
DOCstmt.execute();
DOCstmt.close();
}//end try
catch(SQLException ex){
System.err.println("SQLException: " +ex.getMessage());
}//end catch

try{
DbConnection.getOracleCon();
String oquery= "SELECT distinct r.image_no,r.sim_score,r.norm_image,r.img_title,r.doc_score,r.norm_text
FROM result_tab r order by r.sim_score asc";
PreparedStatement pstmt = DbConnection.getOracleCon().prepareStatement(oquery);
OracleResultSet rs = (OracleResultSet)pstmt.executeQuery();

while (rs.next()){
int Oimagenumber =(rs.getInt(1));
String Oimagescore = (rs.getString(2));
String Onorm_imageString = (rs.getString(3));

if(Onorm_imageString.charAt(0)== ',')
        Onorm_imageString="0".concat(Onorm_imageString);
        String new_Onorm_imageString = Onorm_imageString.replace ( ',', '.');

double Onorm_image = Double.parseDouble(new_Onorm_imageString);
String Oimagetitle =rs.getString(4).trim();
String Odocscore =(rs.getString(5));
String Onorm_textString =(rs.getString(6));

if(Onorm_textString.charAt(0)== ',')
Onorm_textString="0".concat(Onorm_textString);
Onorm_textString = Onorm_textString.replace ( ',', '.');
```

```
double Onorm_text =Double.parseDouble(Onorm_textString);
double OscoresToComb= (0.5 * Onorm_image)+(0.5 * Onorm_text);
double Oglobalscore= 4.9898 * OscoresToComb;
String OtotalCombscore = Double.toString(Oglobalscore );
String Ostring_score = OtotalCombscore.replace ( '.', ',');
try {
BufferedWriter out = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"CAIRANKResults.xls", true));
out.write(Oimagenumber+"\t"+Oimagetitle+ "\t"+Ostring_score+"\n");

out.close();

BufferedWriter out2 = new BufferedWriter(new FileWriter("C:\\Documents and Settings\\Christian\\Mine
dokumenter\\Mastergradsprosjekt\\Datainnsamling\\" +SQL_Stmt0+"RawScoreResults.xls", true));
out2.write(Oimagenumber+"\t"+Oimagetitle+ "\t"+Onorm_imageString+"\n");

out2.close();
}//end try
catch (IOException e) {
System.out.println(e);
}//end catch

} // end while
System.out.print("\n"+"End of Oracle results"+ "\n");
}//end try
catch(SQLException ex){
System.err.println("SQLException: " +ex.getMessage());
}//end catch

try {
DbConnection.getOracleCon();
DbConnection.getIbmCon();
String ibmclearcall= new String("{ call clearTable() }");
CallableStatement ibmCLEARstmt = DbConnection.getIbmCon().prepareCall(ibmclearcall);
ibmCLEARstmt.execute();
ibmCLEARstmt.close();
System.out.println("\n***Clearing DB2 result table Finished***");
String oracleclearcall= new String("{ call clearTable() }");
CallableStatement oracleCLEARstmt =  DbConnection.getOracleCon().prepareCall (oracleclearcall);
oracleCLEARstmt.execute();
oracleCLEARstmt.close();
System.out.println("***Clearing Oracle result table Finished***\n");
}//end try
catch(SQLException ex){
        System.err.println("SQLException: " +ex.getMessage());
}//end catch
}//end getResults()

/*
 * Method that close the connection to the databases
 */
public void close(){
System.exit(0);
}//end close()
}//end class Queryproc
```

Appendix D – Image Collection

This appendix contains the names of all 84 bridges used as the image collection in this project. Different images of all bridges were stored in the two participating databases. The actual images of all the bridges used, are available on the enclosed CD-ROM.

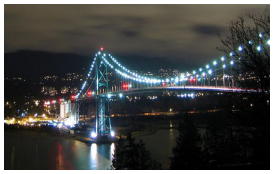
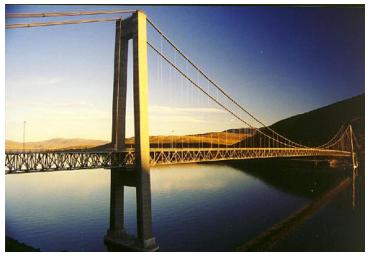
1	Akashi Kaikyo Bridge
2	Ambassador Bridge
3	Aquitaine Bridge
4	Askoy Bridge
5	Astoria Bridge
6	Bear Mountain Bridge
7	Benjamin Franklin Bridge
8	Bosphorus Bridge
9	Breivikstrommen Bridge
10	Brooklyn Bridge
11	BudapestChainBridge
12	Clark Bridge
13	CliftonBridge
14	DamesPointBridge
15	FirthOfForthBridge
16	FredHartmanBridge
17	George WashingtonBridge
18	Gjemnessundet Bridge
19	GoldenGate Bridge
20	Gothenborg Bridge
21	Great Belt Bridge
22	Great Seto Bridge
23	Great Seto Ohashi Bridge - Alternative name for Great Seto Bridge
24	Hakucho Bridge
25	Hawthorne Bridge
26	Hercilio Luz Bridge
27	Hoga Kustenbron
28	Humber Bridge
29	Jacques Cartier Bridge
30	Kap Shui Mun Bridge
31	Lantau Link Bridge: It comprises the Tsing Ma suspension bridge, the Ma Wan Viaduct, and the Kap Shui Mun cable-stayed bridge
32	Little Belt Bridge
33	London Tower Bridge
34	Mac Donald Bridge

35	Mackinac Bridge
36	Manhattan Bridge
37	Marquam Bridge
38	Menai Bridge
39	Mile High Swinging Bridge
40	Millau Bridge
41	Mississippi River Bridge
42	Murray Mackay Bridge
43	Natcher Bridge
44	New Tacoma Bridge
45	Nordhordlands Bridge
46	Normandie Bridge
47	OldLisbon Bridge 25th April Bridge
48	Oresund Bridge
49	Pasco Kennewick Bridge
50	Ponte Vasco da Gama
51	Quebec Bridge
52	Queen Isabella Causeway
53	Queensboro Bridge
54	Rainbow Bridge Tokyo
55	Rama7 Bridge
56	Reubling Bridge
57	Rhein Bridge
58	Richmond San Rafael Bridge
59	Rion Antirion Bridge
60	Royal Gorge Bridge
61	Runyang Bridge
62	San Francisco Bay Bridge
63	Seaway International Bridge
64	Second Orinoco River Bridge
65	Seri Wawsan Bridge
66	Severn Bridge
67	Silver Bridge
68	Skarnsundet Bridge
69	St Johns Bridge
70	Sultan Mehmet Bridge
71	Swietokrzyski Bridge
72	Tacoma Bridge
73	Tasman Bridge
74	Tatara Bridge
75	Tay Bridge
76	Ting Kau Bridge
77	Tjeldsund Bridge

78	Triniti River Bridge
79	Tsing Ma Bridge
80	Verrazano Bridge
81	Walt Whitman Bridge
82	Williamsburg Bridge
83	Yangtze River Bridge
84	Zakim Bridge

Appendix E – Seed Images

This appendix contains the images used as example images for the queries used in the experiment. These images were also used in the process of determining database weights for the participating databases, described in appendix F.

1		2		3	
	STORD BRIDGE NORWAY		OLD LISBON BRIDGE PORTUGAL		YANGLUO BRIDGE CHINA
4		5		6	
	AKASHI KAIKYO BRIDGE JAPAN		LIONS GATE BRIDGE VANCOUVER		AMBASSADOR BRIDGE
7		8		9	
	SEMIPALATINSK BRIDGE KAZAKHSTAN		GREAT BELT BRIDGE		KVALSUND BRIDGE NORWAY
10		11		12	
	SAN FRANCISCO BAY BRIDGE		GOLDEN GATE BRIDGE USA		SEVERN BRIDGE

Appendix F – Determining DB Weights

This appendix contains the complete process of determining the weights assigned to the two participating databases used in this project. Each query provided a weight based on the results from that query. Final weights were determined by multiplying the weights from each query and dividing this amount by the number of queries. An illustrated overview of the process of determining relevance in the result sets is available on the enclosed CD-ROM.

Ibm Query 1

608	SevernBridge4	1	1,0000
384	MillauBridgeFog3	0	0,0000
442	OldLisbonBridge25AprilBridge4	1	0,3333
214	GoldenGateConstruction6	1	0,2500
460	PascoKennewickBridge1	1	0,2000
430	NormandieBridge2	1	0,1667
686	TasmanBridgeConstruction2	0	0,0000
340	ManhattanBridge4	1	0,1250
46	AstoriaBridge1	0	0,0000
712	TingKauBridgeConstruction6	0	0,0000
	Sum:		2,0750
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**4,2330**

Ibm Query 2

614	SevernBridgeNight1	1	1,0000
774	WilliamsburgBridge6	1	0,5000
588	SanFranciscoBayBridgeLightning4	1	0,3333
292	KapShuiMunBridge1	0	0,0000
540	RionAntirionBridgeConstruction5	0	0,0000
208	GoldenGate9	0	0,0000
498	RainbowBridgeTokyo2	1	0,1429
524	RoeblingBridgeNight3	1	0,1250
640	StJohnsBridge11	0	0,0000
126	BudapestChainBridgeNight4	1	0,1000
	Sum:		2,2012
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**4,4904**

Ibm Query 3

4	AkashiKaikyoBridge4	1	1,0000
130	ClarkBridge3	0	0,0000
606	SevernBridge2	1	0,3333
746	VerrazanoBridge1	1	0,2500
420	NewTacomaBridgeConstruction1	1	0,2000
636	StJohnsBridge7	1	0,1667
306	LittleBeltBridge2	1	0,1429
204	GoldenGate5	1	0,1250
330	MackinacBridge6	1	0,1111
374	MillauBridge3	0	0,0000
	Sum:		2,3290
	Relevant:	8	0,8000
	Weight (= 3,4*Sum*R/10):		**6,3348**

Ibm Query 4

178	GeorgeWashingtonBridge11	1	1,0000
86	BosphorusBridge6	1	0,5000
190	GeorgeWashingtonBridgeConstruction9	1	0,3333
400	MurrayMackayBridge5	1	0,2500
756	VerrazanoBridgeConstruction3	1	0,2000
120	BudapestChainBridge1a	1	0,1667
436	NormandieBridgeConstruction2	0	0,0000
582	SanFranciscoBayBridgeConstruction7	0	0,0000
668	Tacoma8	0	0,0000
566	SanFranciscoBayBridge6	1	0,1000
	Sum:		2,5500
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**6,0690**

Ibm Query 5

718	TingKauBridgeNight3	1	1,0000
394	MississippiRiverBridgeNight1	1	0,5000
122	BudapestChainBridgeFog1	1	0,3333
716	TingKauBridgeNight1	1	0,2500
796	ZakimBridge5	1	0,2000
228	GoldenGateFog9	0	0,0000
720	TjeldsundBridge1	1	0,1429
532	RionAntirionBridge2	0	0,0000
44	AskoyBridgeNight3	1	0,1111
488	QueensboroBridgeLightning1	1	0,1000
	Sum:		2,6373
	Relevant:	8	0,8000
	Weight (= 3,4*Sum*R/10):		**7,1735**

Ibm Query 6

216	GoldenGateConstruction8	1	1,0000
788	YangtzeRiverBridgeConstruction5	1	0,5000
740	TsingMaBridgeFog1	1	0,3333
672	TacomaBridgeConstruction1	1	0,2500
218	GoldenGateConstruction10	1	0,2000
210	GoldenGateConstruction2	1	0,1667
560	RunyangBridgeConstruction3	1	0,1429
484	QueenIsabellaCauseway3	0	0,0000
408	NatcherBridge3	1	0,1111
330	MackinacBridge6	1	0,1000
	Sum:		2,8040
	Relevant:	9	0,9000
	Weight (= 3,4*Sum*R/10):		**8,5801**

Ibm Query 7

710	TingKauBridgeConstruction4	1	1,0000
758	VerrazanoBridgeConstruction5	1	0,5000
6	AkashiKaikyoBridge6	1	0,3333
50	AstoriaBridgeFog1	1	0,2500
302	LantauLinkBridge3	1	0,2000
738	TsingMaBridgeConstruction7	1	0,1667
104	BrooklynBridge10	0	0,0000
88	BosphorusBridgeFog1	1	0,1250
612	SevernBridgeConstruction1	1	0,1111
248	GreatBeltBridgeFog1	1	0,1000
	Sum:		2,7861
	Relevant:	9	0,9000
	Weight (= 3,4*Sum*R/10):		**8,5255**

Ibm Query 8

486	QueenIsabellaCauseway5	0	0,0000
434	NormandieBridge6	0	0,0000
188	GeorgeWashingtonBridgeConstruction7	0	0,0000
312	LondonTowerBridge2	0	0,0000
514	Rama7BridgeConstruction8	0	0,0000
366	MenaiBridgeConstruction2	0	0,0000
448	OldLisbonBridge25thAprilBridgeFog3	1	0,1429
686	TasmanBridgeConstruction2	0	0,0000
450	OldLisbonBridge25thAprilBridgeFog6	1	0,1111
214	GoldenGateConstruction6	1	0,1000
	Sum:		0,3540
	Relevant:	3	0,3000
	Weight (= 3,4*Sum*R/10):		**0,3610**

Ibm Query 9

196	GjemnessundetBridge2	1	1,0000
314	LondonTowerBridge4	0	0,0000
604	SeriWawsanBridge2	1	0,3333
628	SkarnsundetBridgeFog1	1	0,2500
674	TacomaBridgeLightning2	1	0,2000
328	MackinacBridge4	1	0,1667
624	SkarnsundetBridge3	1	0,1429
300	LantauLinkBridge1	1	0,1250
246	GreatBeltBridge3	1	0,1111
272	HercilioLuzBridge1	0	0,0000
	Sum:		2,3290
	Relevant:	8	0,8000
	Weight (= 3,4*Sum*R/10):		**6,3348**

Ibm Query 10

308	LittleBeltBridge4	1	1,0000
340	ManhattanBridge4	1	0,5000
450	OldLisbonBridge25thAprilBridgeFog6	1	0,3333
20	AmbassadorBridge7	1	0,2500
656	SultanMehmetBridge6	1	0,2000
264	HawthorneBridge1	1	0,1667
180	GeorgeWashingtonBridge13	0	0,0000
482	QueenIsabellaCauseway1	0	0,0000
148	DamesPointBridge5	0	0,0000
734	TsingMaBridgeConstruction3	0	0,0000
	Sum:		2,4500
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**4,9980**

Ibm Query 11

432	NormandieBridge4	1	1,0000
196	GjemnessundetBridge2	1	0,5000
772	WilliamsburgBridge4	1	0,3333
538	RionAntirionBridgeConstruction3	0	0,0000
56	BearMountainBridge4	1	0,2000
46	AstoriaBridge1	0	0,0000
732	TsingMaBridgeConstruction1	1	0,1429
604	SeriWawsanBridge2	0	0,0000
328	MackinacBridge4	1	0,1111
314	LondonTowerBridge4	0	0,0000
	Sum:		2,2873
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**4,6661**

Ibm Query 12

660	SwietokrzyskiBridgeFog1	1	1,0000
234	GoldenGateLightning3	1	0,5000
90	BosphorusBridgeNight1	1	0,3333
456	OresundBridge5	1	0,2500
744	TsingMaBridgeNight4	1	0,2000
488	QueensboroBridgeLightning1	0	0,0000
444	OldLisbonBridge25AprilBridgeNight1	1	0,1429
418	NewTacomaBridge1	1	0,1250
594	SanFranciscoBayBridgeNight5	1	0,1111
360	MarquamBridgeFog2	0	0,0000
	Sum:		2,6623
	Relevant:	8	0,8000
	Weight (= 3,4*Sum*R/10):		**7,2415**

Global IBM Database Weight:	**5,7506**

Oracle Query 1

785	YangtzeRiverBridgeConstruction2	1	1,0000
515	Rama7BridgeConstruction9	1	0,5000
191	GeorgeWashingtonBridgeConstruction10	1	0,3333
257	HakuchoBridge3	1	0,2500
213	GoldenGateConstruction5	1	0,2000
271	HawthorneBridgeFog5	0	0,0000
109	BrooklynBridgeConstruction2	1	0,1429
715	TingKauBridgeFog3	1	0,1250
297	KapShuiMunBridgeConstruction3	0	0,0000
705	TingKauBridge3	0	0,0000
	Sum:		2,5512
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**6,0718**

Oracle Query 2

797	ZakimBridge6	1	1,0000
407	NatcherBridge2	0	0,0000
523	RoeblingBridgeNight2	1	0,3333
125	BudapestChainBridgeNight3	1	0,2500
591	SanFranciscoBayBridgeNight2	1	0,2000
237	GoldenGateLightning6	1	0,1667
631	StJohnsBridge2	1	0,1429
223	GoldenGateFog4	0	0,0000
137	CliftonBridge3	0	0,0000
59	BearMountainBridge7	1	0,1000
	Sum:		2,1929
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**5,2190**

Oracle Query 3

327	MackinacBridge3	1	1,0000
783	YangtzeRiverBridge5	1	0,5000
487	QueenIsabellaCauseway6	0	0,0000
7	AkashiKaikyoBridgeConstruction	1	0,2500
693	TataraBridge5	1	0,2000
537	RionAntirionBridgeConstruction2	1	0,1667
529	RichmondSanRafelBridge4	0	0,0000
541	RionAntirionBridgeNight1	1	0,1250
455	OresundBridge4	1	0,1111
259	HakuchoBridgeNight2	0	0,0000
	Sum:		2,3528
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**5,5996**

Oracle Query 4

213	GoldenGateConstruction5	1	1,0000
191	GeorgeWashingtonBridgeConstruction10	1	0,5000
397	MurrayMackayBridge2	1	0,3333
139	CliftonBridge5	1	0,2500
215	GoldenGateConstruction7	1	0,2000
67	BenjaminFranklinBridge2	1	0,1667
785	YangtzeRiverBridgeConstruction4	1	0,1429
109	BrooklynBridgeConstruction2	1	0,1250
515	Rama7BridgeConstruction9	0	0,0000
155	FirthOfForthBridge4	0	0,0000
	Sum:		2,7179
	Relevant:	8	0,8000
	Weight (= 3,4*Sum*R/10):		**7,3926**

Oracle Query 5

577	SanFranciscoBayBridgeConstruction2	0	0,0000
741	TsingMaBridgeNight1	1	0,5000
687	TasmanBridgeNight2	1	0,3333
769	WilliamsburgBridge1	0	0,0000
317	LondonTowerBridgeNight2	1	0,2000
489	QueensboroBridgeLightning2	1	0,1667
657	SultanMehmetBridgeNight1	1	0,1429
793	ZakimBridge2	1	0,1250
407	NatcherBridge2	0	0,0000
79	BenjaminFranklinBridgeNight4	1	0,1000
	Sum:		1,5679
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**3,7315**

Oracle Query 6

23	AmbassadorBridgeConstruction2	1	1,0000
291	JacquesCartierBridgeConstruction2	0	0,0000
73	BenjaminFranklinBridgeConstruction2	1	0,3333
613	SevernBridgeConstruction2	1	0,2500
35	AquitaineBridge5	0	0,0000
401	MurrayMackayBridgeFog1	1	0,1667
193	GeorgeWashingtonBridgeConstruction12	1	0,1429
159	FirthOfForthBridgeConstruction2	0	0,0000
731	TsingMaBridge1	1	0,1111
509	Rama7BridgeConstruction3	1	0,1000
	Sum:		2,1040
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**5,0074**

Oracle Query 7

671	Tacoma11	1	1,0000
99	BrooklynBridge5	0	0,0000
731	TsingMaBridge1	1	0,3333
713	TingKauBridgeFog1	0	0,0000
509	Rama7BridgeConstruction3	0	0,0000
245	GreatBeltBridge2	1	0,1667
321	MacDonaldBridge3	1	0,1429
255	HakuchoBridge1	1	0,1250
485	QueenIsabellaCauseway4	0	0,0000
677	TasmanBridge3	0	0,0000
	Sum:		1,7679
	Relevant:	5	0,5000
	Weight (= 3,4*Sum*R/10):		**3,0054**

Oracle Query 8

331	MacKinacBridge7	1	1,0000
311	LondonTowerBridge1	0	0,0000
677	TasmanBridge3	0	0,0000
245	GreatBeltBridge2	1	0,2500
433	NormandieBridge5	0	0,0000
647	StJohnsBridgeFog5	1	0,1667
305	LittleBeltBridge1	1	0,1429
713	TingKauBridgeFog1	0	0,0000
513	Rama7BridgeConstruction7	1	0,1111
403	MurrayMackayBridgeFog3	1	0,1000
	Sum:		1,7706
	Relevant:	6	0,6000
	Weight (= 3,4*Sum*R/10):		**3,6121**

Oracle Query 9

289	JacquesCartierBridge2	1	1,0000
409	NatcherBridge4	1	0,5000
343	ManhattanBridge7	1	0,3333
275	HumberBridge1	1	0,2500
81	BosphorusBridge1	1	0,2000
201	GoldenGate2	1	0,1667
299	KapShuiMunBridgeConstruction5	0	0,0000
619	Silverbridge1	0	0,0000
373	MillauBridge2	0	0,0000
129	ClarkBridge2	1	0,1000
	Sum:		2,5500
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**6,0690**

Oracle Query 10

309	LittleBeltBridge5	1	1,0000
557	RunyangBridge4	1	0,5000
773	WilliamsburgBridge5	1	0,3333
573	SanFranciscoBayBridge13	0	0,0000
483	QueenIsabellaCauseway2	0	0,0000
251	GreatSetoBridge3	1	0,1667
525	RheinBridge1	1	0,1429
697	TataraBridgeConstruction1	1	0,1250
385	MillauBridgeFog4	0	0,0000
777	WilliamsburgBridgeConstruction2	1	0,1000
	Sum:		2,3679
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**5,6355**

Oracle Query 11

79	BenjaminFranklinBridgeNight4	1	1,0000
619	Silverbridge1	0	0,0000
57	BearMountainBridge5	1	0,3333
639	StJohnsBridge10	1	0,2500
379	MillauBridgeConstruction2	1	0,2000
431	NormandieBridge3	1	0,1667
373	MillauBridge2	0	0,0000
201	GoldenGate2	0	0,0000
71	BenjaminFranklinBridgeConstruction2	0	0,0000
723	TjeldsundBridge4	0	0,0000
	Sum:		1,9500
	Relevant:	5	0,5000
	Weight (= 3,4*Sum*R/10):		**3,3150**

Oracle Query 12

465	PascoKennewickBridgeNight3	1	1,0000
489	QueensboroBridgeLightning2	0	0,0000
475	PonteVascodaGamaNight1	1	0,3333
13	AkashiKaikyoBridgeNight2	1	0,2500
233	GoldenGateLightning2	1	0,2000
687	TasmanBridgeNight2	1	0,1667
741	TsingMaBridgeNight1	1	0,1429
577	SanFranciscoBayBridgeConstruction2	0	0,0000
407	NatcherBridge2	0	0,0000
317	LondonTowerBridgeNight2	1	0,1000
	Sum:		2,1929
	Relevant:	7	0,7000
	Weight (= 3,4*Sum*R/10):		**5,2190**

Global Oracle Database Weight:	**4,9898**

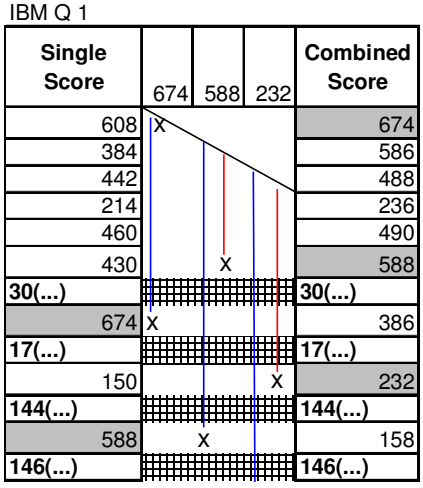
Appendix G – Distance measures

This appendix contains the Distance results for all queries. A total of 12 queries were created. Each of these queries was expressed using an example image accompanied by different query terms (QT). For each query, image results were ranked using both the Single Score merge approach and the Combined Score approach. Relevant images in the results are marked with grey shade.

Distance measures:

Ibm Queries:

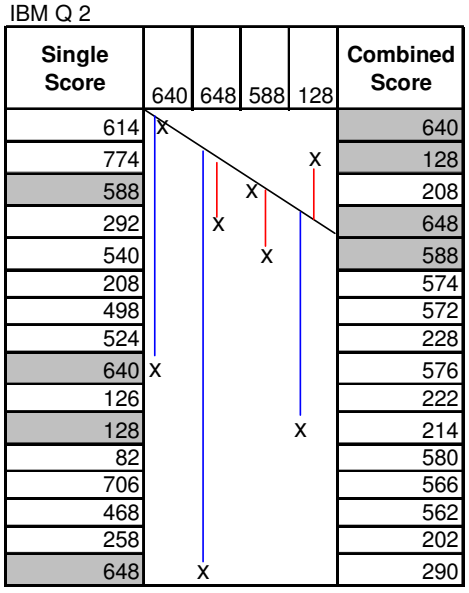
IBM Q 1



Single Score	674	588	232	Combined Score
608	X			674
384				586
442				488
214				236
460				490
430		X		588
30(...)				30(...)
674	X			386
17(...)				17(...)
150			X	232
144(...)				144(...)
588		X		158
146(...)				146(...)

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
674	36	0	36	0
588	198	4	198	4
232	344	52	344	52
		Sum:	578	56
		Average:	192,67	18,67

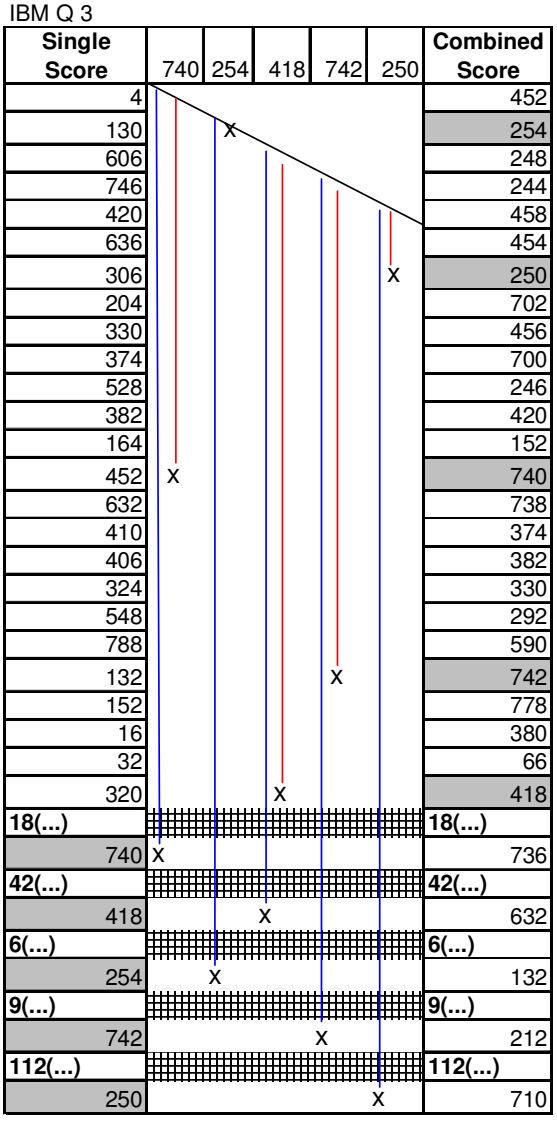
IBM Q 2



Single Score	640	648	588	128	Combined Score
614	X				640
774				X	128
588			X		208
292		X			648
540			X		588
208					574
498					572
524					228
640	X				576
126					222
128				X	214
82					580
706					566
468					562
258					202
648		X			290

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
640	8	0	8	0
648	14	2	14	2
588	0	2	0	2
128	7	-2	7	2
		Sum:	29	6
		Average:	9,67	2,00

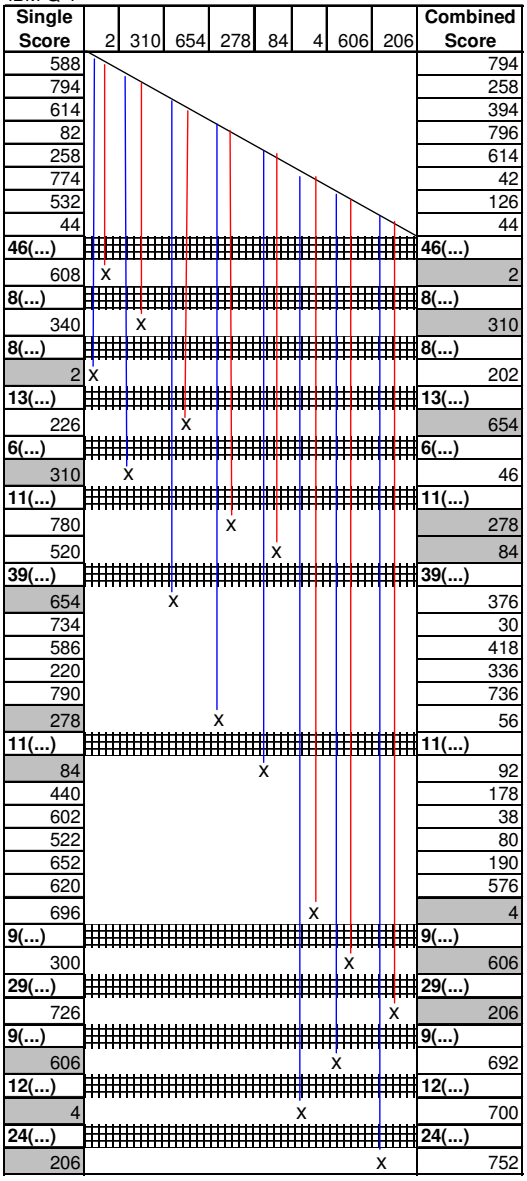
IBM Q 3



Single Score	740	254	418	742	250	Combined Score
4						452
130		x				254
606						248
746						244
420						458
636						454
306					x	250
204						702
330						456
374						700
528						246
382						420
164						152
452	x					740
632						738
410						374
406						382
324						330
548						292
788						590
132				x		742
152						778
16						380
32						66
320			x			418
18(...)						**18(...)**
740	x					736
42(...)						**42(...)**
418			x			632
6(...)						**6(...)**
254		x				132
9(...)						**9(...)**
742				x		212
112(...)						**112(...)**
250					x	710

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
740	42	13	42	13
254	92	0	92	0
418	84	22	84	22
742	100	17	100	17
250	211	2	211	2
		Sum:	529	54

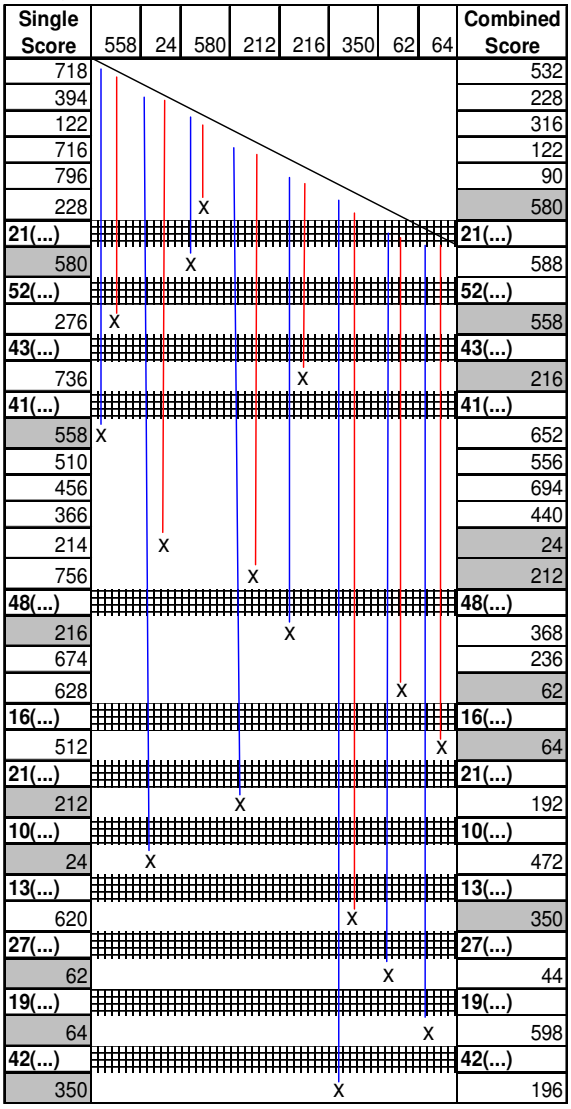
IBM Q 4



Single Score	2	310	654	278	84	4	606	206	Combined Score
588									794
794									258
614									394
82									796
258									614
774									42
532									126
44									44
46(...)									**46(...)**
608	x								2
8(...)									**8(...)**
340		x							310
8(...)									**8(...)**
2	x								202
13(...)									**13(...)**
226			x						654
6(...)									**6(...)**
310		x							46
11(...)									**11(...)**
780				x					278
520					x				84
39(...)									**39(...)**
654			x						376
734									30
586									418
220									336
790									736
278				x					56
11(...)									**11(...)**
84					x				92
440									178
602									38
522									80
652									190
620									576
696						x			4
9(...)									**9(...)**
300							x		606
29(...)									**29(...)**
726								x	206
9(...)									**9(...)**
606							x		692
12(...)									**12(...)**
4						x			700
24(...)									**24(...)**
206								x	752

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
2	72	54	72	54
310	92	62	92	62
654	144	84	144	84
278	148	102	148	102
84	159	102	159	102
4	227	164	227	164
606	213	173	213	173
206	250	202	250	202
		Sum:	1305	943
		Average:	435,00	314,33

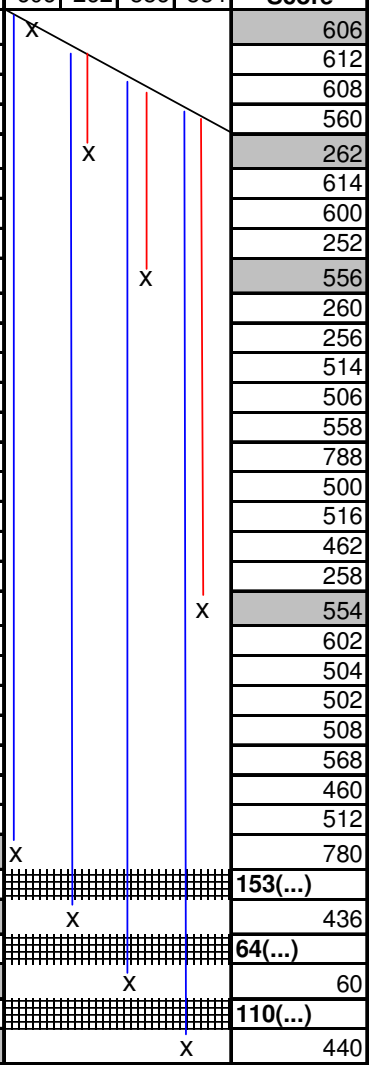
IBM Q 5



Single Score	558	24	580	212	216	350	62	64	Combined Score
718									532
394									228
122									316
716									122
796									90
228			x						580
21(...)									21(...)
580			x						588
52(...)									52(...)
276	x								558
43(...)									43(...)
736					x				216
41(...)									41(...)
558	x								652
510									556
456									694
366									440
214		x							24
756				x					212
48(...)									48(...)
216					x				368
674									236
628							x		62
16(...)									16(...)
512								x	64
21(...)									21(...)
212				x					192
10(...)									10(...)
24		x							472
13(...)									13(...)
620						x			350
27(...)									27(...)
62							x		44
19(...)									19(...)
64								x	598
42(...)									42(...)
350						x			196

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
558	166	80	166	80
24	271	169	271	169
580	25	3	25	3
212	258	168	258	168
216	216	120	216	120
350	372	281	372	281
62	308	216	308	216
64	327	232	327	232
		Sum:	1943	1269

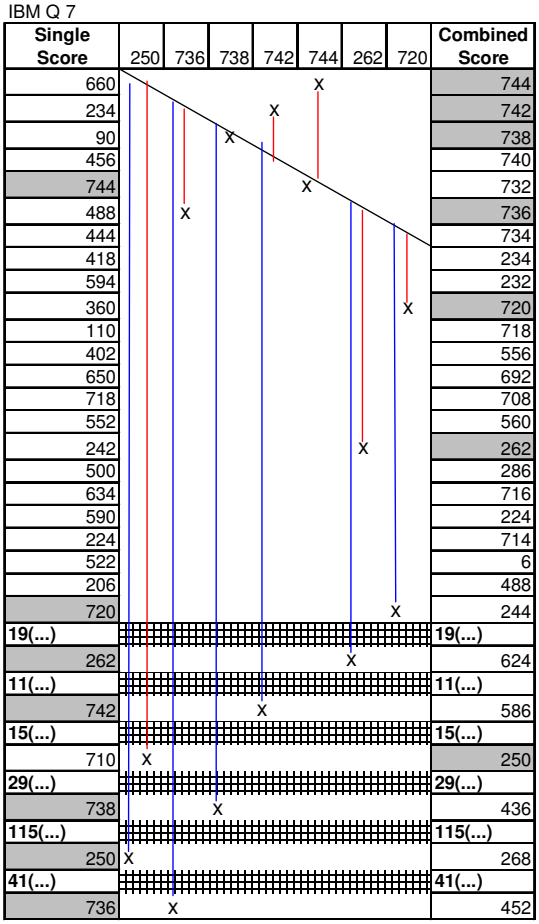
IBM Q 6



Single Score	606	262	556	554	Combined Score
106	x				606
32					612
478					608
408					560
16		x			262
746					614
50					600
528					252
320			x		556
380					260
302					256
738					514
266					506
132					558
752					788
58					500
200					516
458					462
642					258
622				x	554
452					602
672					504
134					502
210					508
330					568
218					460
788					512
606	x				780
153(...)					153(...)
262		x			436
64(...)					64(...)
556			x		60
110(...)					110(...)
554				x	440

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
606	27	0	27	0
262	180	3	180	3
556	244	6	244	6
554	354	16	354	16
		Sum:	805	25
		Average:	268,33	8,33

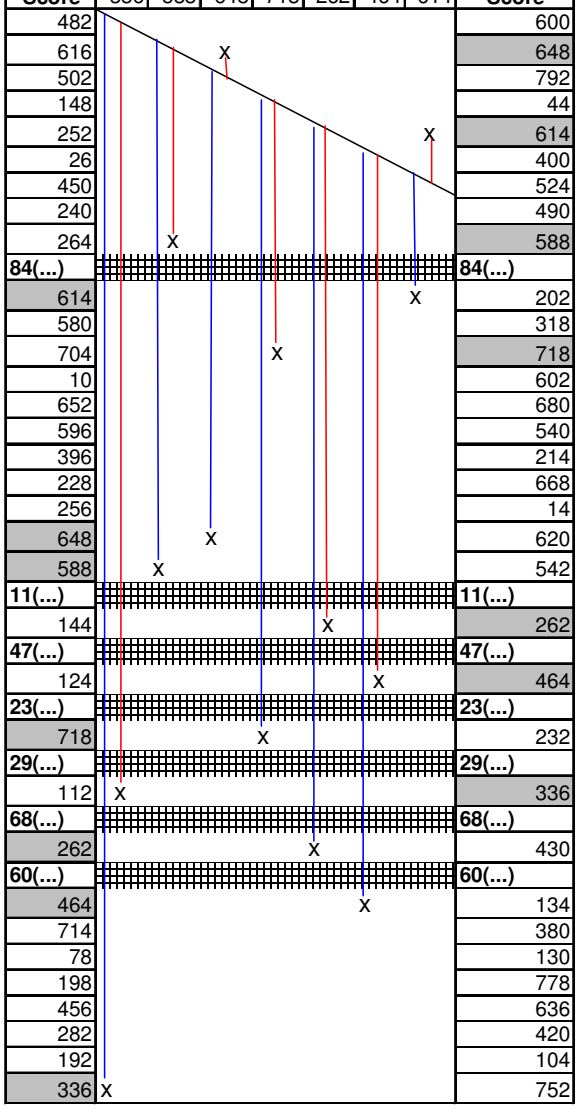
IBM Q 7



Single Score	250	736	738	742	744	262	720	Combined Score
660					x			744
234				x				742
90			x					738
456								740
744					x			732
488		x						736
444								734
418								234
594								232
360							x	720
110								718
402								556
650								692
718								708
552								560
242						x		262
500								286
634								716
590								224
224								714
522								6
206								488
720							x	244
19(...)								19(...)
262						x		624
11(...)								11(...)
742				x				586
15(...)								15(...)
710	x							250
29(...)								29(...)
738			x					436
115(...)								115(...)
250	x							268
41(...)								41(...)
736		x						452

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
250	216	70	216	70
736	257	4	257	4
738	98	0	98	0
742	51	-2	51	2
744	0	-4	0	4
262	36	10	36	10
720	16	3	16	3
		Sum:	674	93
		Average:	224,67	31,00

IBM Q 8



Single Score	336	588	648	718	262	464	614	Combined Score
482								600
616			x					648
502								792
148								44
252							x	614
26								400
450								524
240								490
264		x						588
84(...)								84(...)
614							x	202
580								318
704				x				718
10								602
652								680
596								540
396								214
228								668
256								14
648			x					620
588		x						542
11(...)								11(...)
144					x			262
47(...)								47(...)
124						x		464
23(...)								23(...)
718				x				232
29(...)								29(...)
112	x							336
68(...)								68(...)
262					x			430
60(...)								60(...)
464						x		134
714								380
78								130
198								778
456								636
282								420
192								104
336	x							752

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
336	354	217	354	217
588	102	7	102	7
648	100	-1	100	1
718	184	92	184	92
262	282	111	282	111
464	342	158	342	158
614	87	-2	87	2
		Sum:	1451	588
		Average:	483,67	196,00

IBM Q 9

Single Score	92	238	230	656	Combined Score
196					654
314					652
604					220
628					214
674					84
328	X				92
624				X	656
10(...)					**10(...)**
284			X		230
9(...)					**9(...)**
344		X			238
30(...)					**30(...)**
92	X				626
38(...)					**38(...)**
656				X	396
138(...)					**138(...)**
230			X		494
135(...)					**135(...)**
238		X			420

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
92	59	5	59	5
238	321	26	321	26
230	224	15	224	15
656	95	3	95	3
		Sum:	699	49
		Average:	233,00	16,33

IBM Q 10

Single Score	664	670	672	674	666	Combined Score
274				X		674
340						668
658					X	666
626		X				670
250	X					664
712			X			672
61(...)						**61(...)**
674				X		576
192(...)						**192(...)**
666					X	564
640						156
444						128
670		X				720
23(...)						**23(...)**
664	X					176
70						454
644						174
110						286
34						714
544						526
752						400
322						650
354						302
672			X			504

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |
664	286	4	286
670	262	2	262
672	294	3	294
674	64	-3	64
666	256	-2	256
		Sum:	1162
		Average:	387,33

IBM Q 11

Single Score	656	90	92	Combined Score
432				444
196				446
772				470
21(...)				**21(...)**
92			X	430
100(...)				**100(...)**
692	X			656
49(...)				**49(...)**
656	X			276
91(...)				**91(...)**
90		X		346
70(...)				**70(...)**
416			X	92
29(...)				**29(...)**
742		X		90

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
656	175	125	175	125
90	266	367	266	367
92	22	336	22	336
		Sum:	463	828
		Average:	154,33	276,00

IBM Q 12

Single Score	200	606	612	672	746	Combined Score
766						618
422		X				606
548					X	746
420						282
618				X		672
130						568
374	X					200
798						144
404						50
452						748
172						254
636			X			612
378						276
212						566
320						312
606		X				278
16(...)						**16(...)**
746					X	608
9(...)						**9(...)**
672				X		374
20(...)						**20(...)**
200	X					388
56(...)						**56(...)**
612			X			120

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
200	63	6	63	6
606	14	0	14	0
612	118	9	118	9
672	39	1	39	1
746	28	-2	28	2
		Sum:	262	18
		Average:	87,33	6,00

Oracle Queries:

Oracle Q1

Single Score	233	323	235	673	5	Combined Score
785						353
515		x				323
191						587
257				x		673
213						89
271						675
109						585
715						589
297			x			235
705						335
155						237
215						387
75					x	5
151						143
629						489
667	x					233
37(...)						**37(...)**
323		x				211
22(...)						**22(...)**
673				x		203
133(...)						**133(...)**
235			x			201
131(...)						**131(...)**
5					x	401
38(...)						**38(...)**
233	x					487

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
233	381	15	381	15
323	52	0	52	0
235	208	6	208	6
673	73	0	73	0
5	5	8	5	8
		Sum:	719	29
		Average:	239,67	9,67

Oracle Q 2

Single Score	133	409	591	795	Combined Score
797					797
407					407
523				x	795
125			x		409
591				x	523
237					631
631					799
223					793
137					129
59					95
95	x				133
627					317
409		x	x		591
205					801
745					413
545					237
317					223
343					637
655					205
543					411
795				x	325
56(...)					**56(...)**
133	x				97

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
133	76	10	76	10
409	11	2	11	2
591	2	10	2	10
795	17	-1	17	1
		Sum:	106	23
		Average:	35,33	7,67

Oracle Q 3

Single Score	775	247	777	739	Combined Score
327					455
783					457
487					453
7				X	739
693		X			247
537					745
529					735
541					459
455					773
259			X		777
427					737
277					731
17					153
287					245
19					743
85					159
637					733
471					741
325	X				775
21(...)					**21(...)**
739				X	85
19(...)					**19(...)**
247		X			539
109(...)					**109(...)**
777			X		729
108(...)					**108(...)**
775	X				291

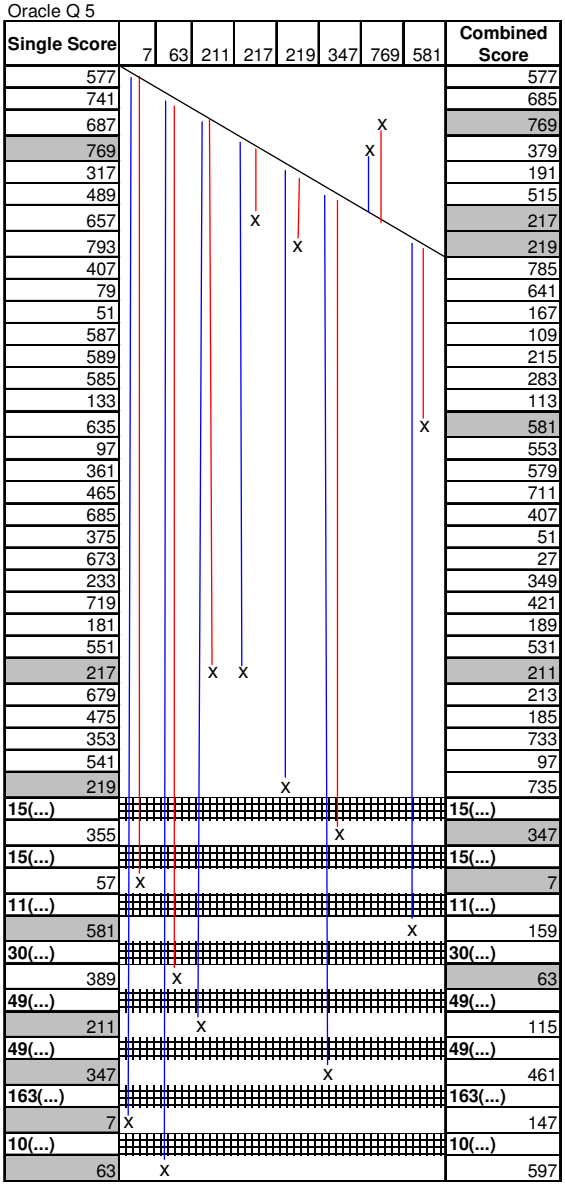
Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
775	279	17	279	17
247	59	3	59	3
777	168	7	168	7
739	37	0	37	0
		Sum:	543	27
		Average:	181,00	9,00

Oracle Q 4

Single Score	637	449	1	195	441	85	Combined Score
637	X					X	637
43							529
445							19
91							171
427		X					449
541							429
285							455
529							537
171							457
449		X	X				1
19				X			195
429							391
1			X				691
455							5
51							471
537							453
195				X			165
5							501
287							655
391							693
457					X		441
691							241
259							327
453							369
165						X	85
8(...)							**8(...)**
441					X		249
369							487
17							571
85						X	651

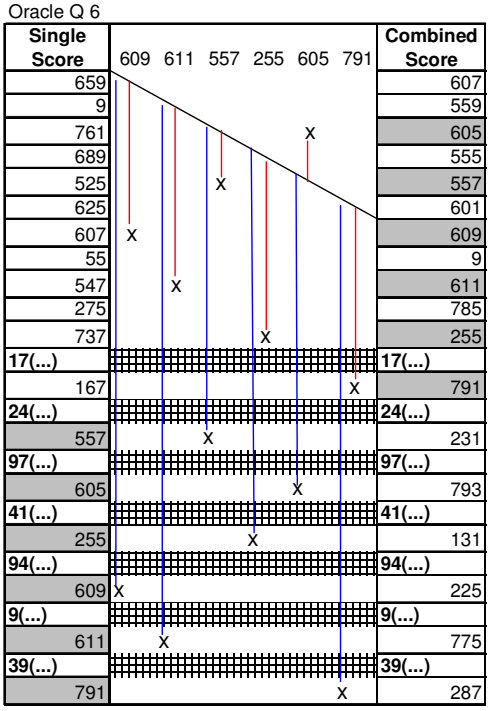
Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
637	0	0	0	0
449	8	3	8	3
1	10	7	10	7
195	13	7	13	7
441	29	16	29	16
85	31	19	31	19
		Sum:	91	52
		Average:	30,33	17,33

Oracle Q 5



Single Score	7	63	211	217	219	347	769	581	Combined Score
577									577
741									685
687								x	769
769							x		379
317									191
489									515
657									217
793									219
407									785
79									641
51									167
587									109
589									215
585									283
133									113
635								x	581
97									553
361									579
465									711
685									407
375									51
673									27
233									349
719									421
181									189
551									531
217			x	x					211
679									213
475									185
353									733
541									97
219					x				735
15(...)									15(...)
355						x			347
15(...)									15(...)
57	x								7
11(...)									11(...)
581								x	159
30(...)									30(...)
389		x							63
49(...)									49(...)
211			x						115
49(...)									49(...)
347						x			461
163(...)									163(...)
7	x								147
10(...)									10(...)
63		x							597

Oracle Q 6



Single Score	609	611	557	255	605	791	Combined Score
659							607
9							559
761					x		605
689							555
525			x				557
625							601
607	x						609
55							9
547		x					611
275							785
737				x			255
17(...)							17(...)
167						x	791
24(...)							24(...)
557			x				231
97(...)							97(...)
605					x		793
41(...)							41(...)
255				x			131
94(...)							94(...)
609	x						225
9(...)							9(...)
611		x					775
39(...)							39(...)
791						x	287

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
609	288	6	288	6
611	297	7	297	7
557	51	2	51	2
255	190	7	190	7
605	147	-2	147	2
791	333	23	333	23
		Sum:	1306	47
		Average:	435,33	15,67

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
7	370	63		63
63	380	105		105
211	154	24	154	24
217	23	3	23	3
219	27	3	27	3
347	201	41	201	41
769	-3	-4	3	4
581	68	8	68	8
		Sum:	476	251
		Average:	158,67	83,67

Oracle Q 7

Single Score	741	13	261	717	499	Combined Score
465	x					741
489		x				465
475		x				745
13						13
233						233
687						719
741	x			x		717
577			x			261
407						235
14(...)						**14(...)**
261			x			657
235						407
717				x		317
188(...)						**188(...)**
499					x	641
47(...)						**47(...)**
19					x	499

Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
6	0	6	0
2	2	2	2
21	5	21	5
22	3	22	3
210	258	210	258
	Sum:	261	268
	Average:	87,00	89,33

Oracle Q 8

Single Score	541	463	239	335	Combined Score
573	x				541
605					533
509					137
255					135
104(...)					**104(...)**
417				x	335
42(...)					**42(...)**
629		x			463
10(...)					**10(...)**
541	x				795
25(...)					**25(...)**
335				x	47
44(...)					**44(...)**
463		x			735
16(...)					**16(...)**
583			x		239
587					131
683					117
611					527
323					593
419			x		701
239					37

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
541	162	0	162	0
463	232	150	232	150
239	254	248	254	248
335	185	105	185	105
		Sum:	833	503
		Average:	277,67	167,67

Oracle Q 9

Single Score	237	235	221	239	81	Combined Score
289						231
409						201
343					x	81
275			x			221
81	x					237
201						223
299						289
619						653
373						207
129						655
101		x				235
7(...)						**7(...)**
317				x		239
747						213
125						203
557						657
221			x			215
11(...)						**11(...)**
237	x					91
84(...)						**84(...)**
235		x				389
93(...)						**93(...)**
239				x		191

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |
237	34	4	34
235	118	9	118
221	20	1	20
239	210	15	210
81	0	-2	0
		Sum:	382
		Average:	127,33

Oracle Q 10

Single Score	669	663	673	661	667	Combined Score
327						327
19						1
85						739
259		x				663
457				x		661
453						5
539						7
241						329
427						325
1						671
637						333
287			x			673
781					x	667
529	x					669
157(...)						**157(...)**
663		x				407
18(...)						**18(...)**
661			x			415
133(...)						**133(...)**
673			x			113
29(...)						**29(...)**
667					x	177
669	x					77

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
669	355	13	355	13
663	170	2	170	2
673	322	9	322	9
661	187	1	187	1
667	350	8	350	8
		Sum:	1034	25
		Average:	344,67	8,33

Oracle Q 11

Single Score	81	87	91	Combined Score
79				253
619				653
57		X		87
639				83
379	X			81
431				9
373				251
201				3
71				505
723				89
269				655
689				501
375			X	91
41(...)				**41(...)**
87		X		439
12(...)				**12(...)**
81	X			647
212(...)				**212(...)**
91			X	797

Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
81	67	4	67	4
87	53	1	53	1
91	278	10	278	10
		Sum:	398	15
		Average:	132,67	5,00

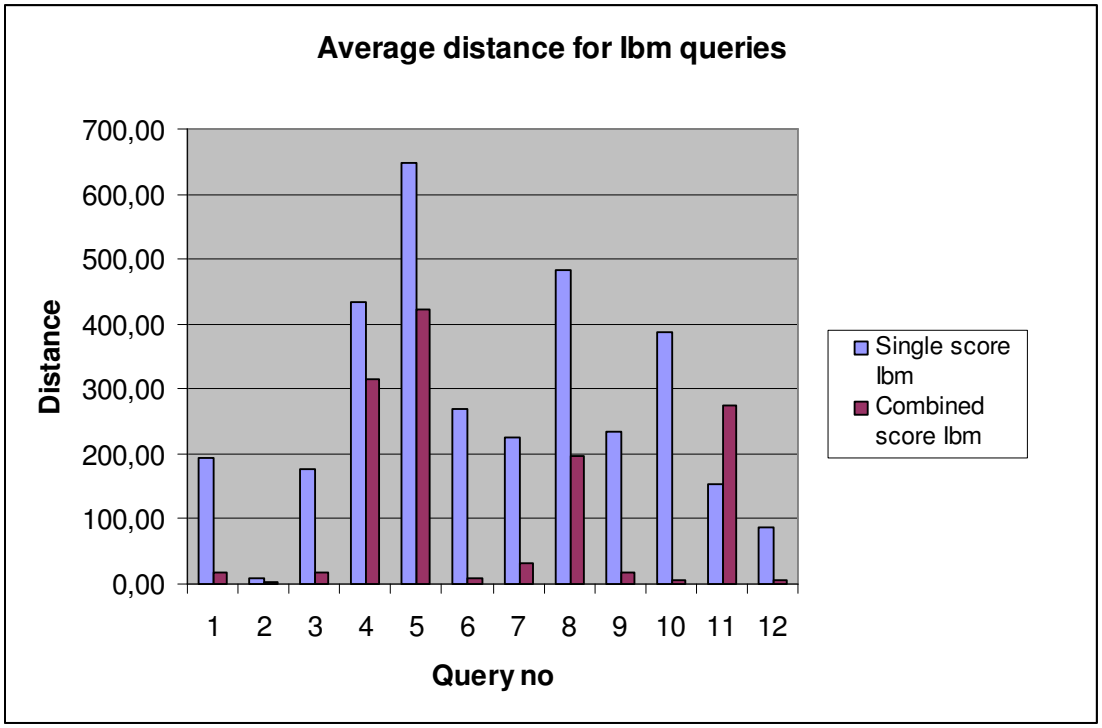
Oracle Q 12

Single Score	279	611	607	609	591	Combined Score
195						529
91						285
449						693
19						287
445						691
427						281
637						571
43					X	591
259		X				611
327						751
529				X		609
487						569
455						573
285						617
17						711
277			X			607
52(...)						**52(...)**
591					X	345
6(...)						**6(...)**
171	X					279
611		X				615
11(...)						**11(...)**
609				X		167
44(...)						**44(...)**
607			X			535
202(...)						**202(...)**
279	X					293

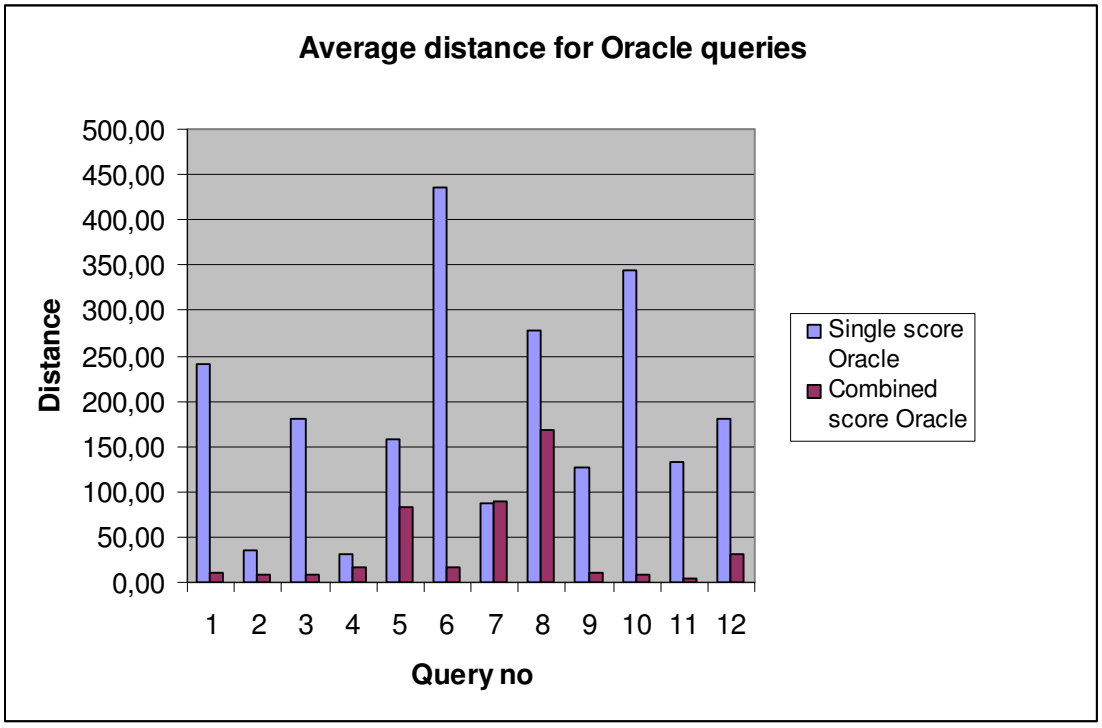
Ideal set	Distance Single Score (S)	Distance Combined Score (C)	| S |	| C |
279	336	75	336	75
611	75	7	75	7
607	131	13	131	13
609	85	7	85	7
591	64	3	64	3
		Sum:	542	95
		Average:	180,67	31,67

Average Distance:

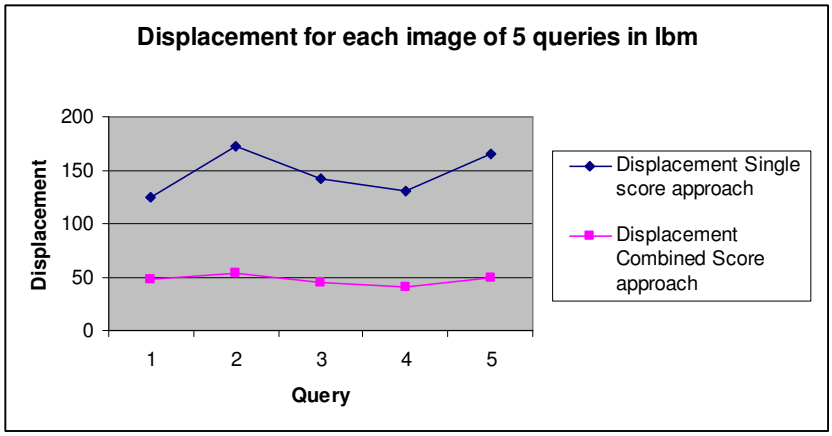
lbm					
Query no.	| S |	| C |	Relevant retrieved	Average | S |	Average | C |
1	578	56	3	192,67	18,67
2	29	6	4	9,67	2,00
3	529	54	5	176,33	18,00
4	1305	943	8	435,00	314,33
5	1943	1269	8	647,67	423,00
6	805	25	4	268,33	8,33
7	674	93	7	224,67	31,00
8	1451	588	7	483,67	196,00
9	699	49	4	233,00	16,33
10	1162	14	5	387,33	4,67
11	463	828	3	154,33	276,00
12	262	18	5	87,33	6,00
Total	9900	3943	63	157,14	62,59



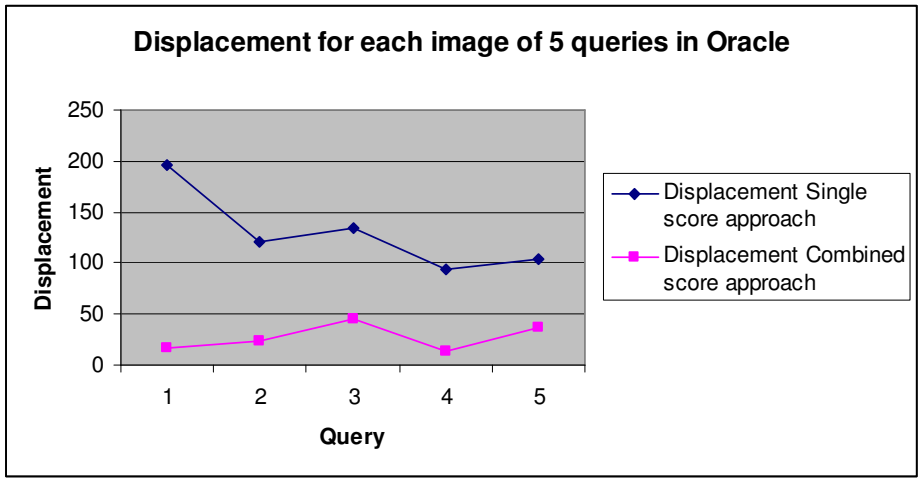
Oracle					
Query no.	| S |	| C |	Relevant retrieved	Average | S |	Average | C |
1	719	29	5	239,67	9,67
2	106	23	4	35,33	7,67
3	543	27	4	181,00	9,00
4	91	52	6	30,33	17,33
5	476	251	8	158,67	83,67
6	1306	47	6	435,33	15,67
7	261	268	5	87,00	89,33
8	833	503	4	277,67	167,67
9	382	29	5	127,33	9,67
10	1034	25	5	344,67	8,33
11	398	15	3	132,67	5,00
12	542	95	5	180,67	31,67
Total	6691	1364	60	111,52	22,73



Ibm			
Ideal position	Number of queries with ideal set >=5	Displacement Single score approach	Displacement Combined score approach
1	12	125,3333333	47,83333333
2	12	172,4166667	53,83333333
3	12	141,4166667	44,41666667
4	10	130	40,6
5	7	164,5714286	49



Oracle			
Ideal position	Number of queries with ideal set >=5	Displacement Single score approach	Displacement Combined score approach
1	12	196,1666667	17,25
2	12	121,4166667	24,25
3	12	134,9166667	45,33333333
4	11	94,72727273	13,54545455
5	8	104	37,5



Appendix H – Precision measures

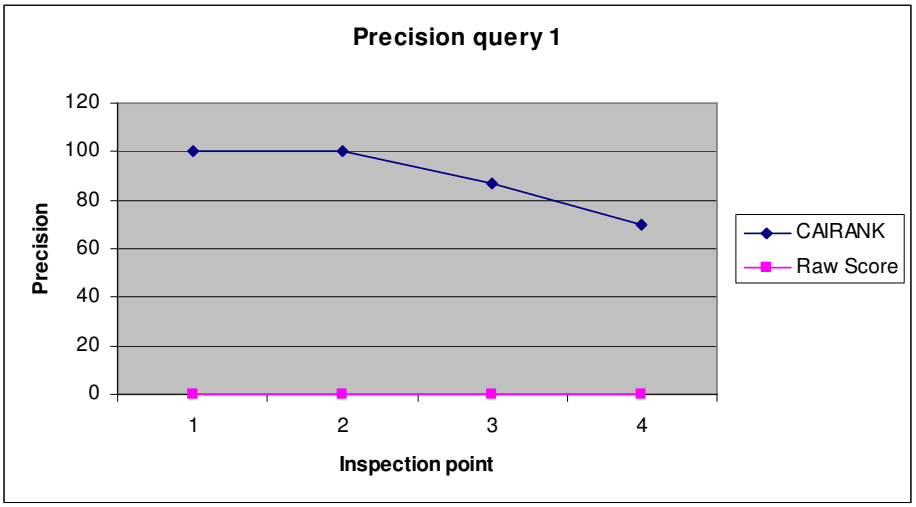
This appendix contains the Precision results for all queries. A total of 12 queries were created. Each of these queries was expressed using an example image accompanied by different query terms (QT). For each query, image results were ranked using both the Raw Score merge approach and the functionality implemented in the CAIRANK prototype. Relevant images in the results are marked with grey shade.

Precision measures:

Query No:1
QT:
Lightning, thunderstorm, electric discharge

Query image 1

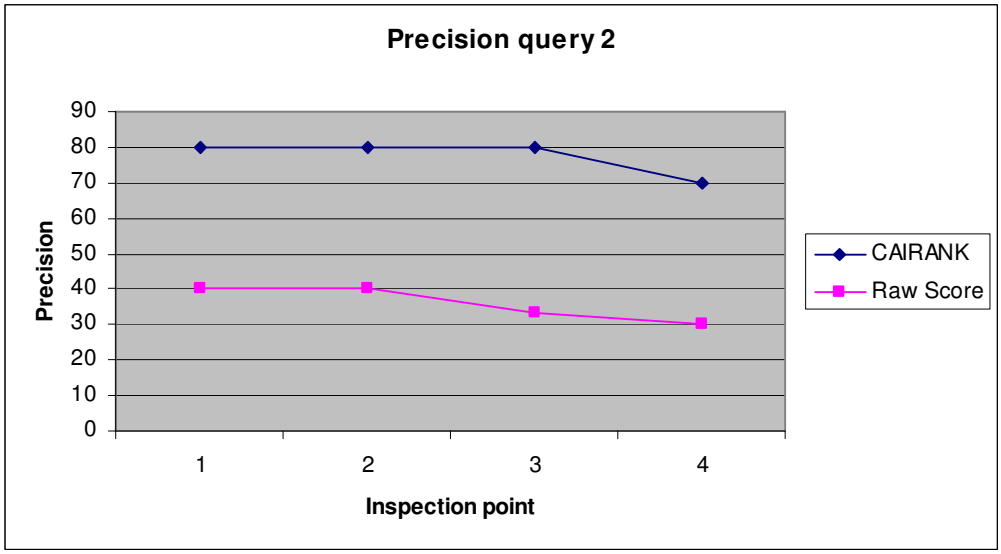
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
353	4,70213803		608	1	
323	4,696848842		785	1	
587	4,675068365	100	384	0,99504	0
673	4,60708234		515	0,99129	
89	4,570581953		442	0,9891	
675	4,362432446		191	0,98868	
585	4,337583242		214	0,98799	
589	4,333616351	100	460	0,98776	0
674	4,298027854		430	0,98444	
235	4,123021842		257	0,98442	
586	3,982681948		686	0,98388	
335	3,97262927		340	0,98382	
488	3,903573151	86,66666667	46	0,98102	0
490	3,877509118		712	0,98007	
236	3,877095819		412	0,97901	
588	3,8264593		274	0,97824	
234	3,53770346		20	0,97718	
237	3,45992732	70	220	0,97678	0
232	3,384146452		750	0,97584	
387	3,328296396		213	0,9743	



Query No:2 **Query image 2**

QT: U.S.
Boston,
Portland, San
Francisco,
St. Louis,
Cincinnati,
Rockport

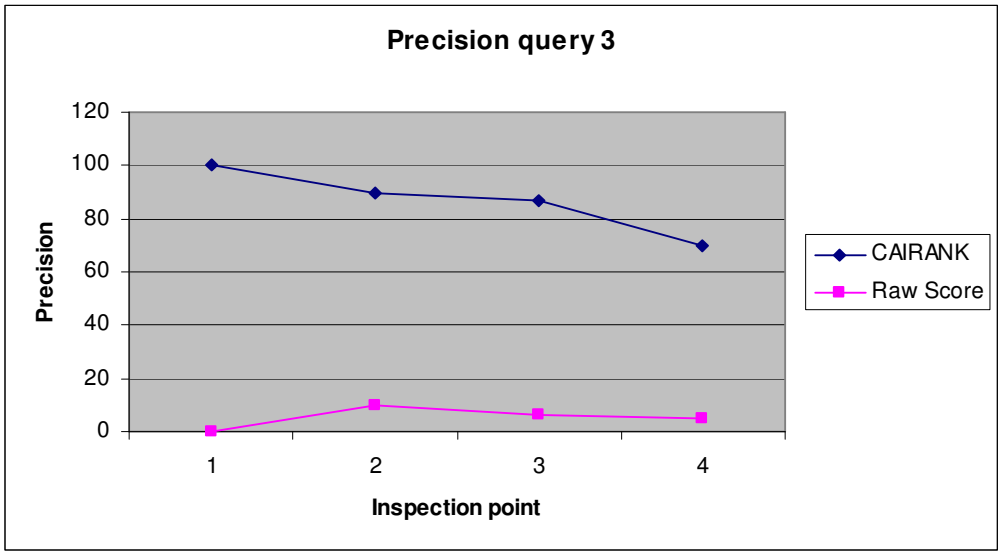
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
797	4,9898		614	1	
407	4,597402128		797	1	
795	4,506063839	80	774	0,9958	40
409	4,403872735		588	0,9853	
523	4,386682874		292	0,98494	
631	4,281946972		540	0,98346	
799	4,265854867		208	0,98337	
793	4,241928776	80	498	0,98322	40
129	4,221520494		524	0,98278	
640	4,219580895		640	0,98064	
95	4,218152379		126	0,97795	
208	4,212836879		128	0,97605	
133	4,210093852	80	82	0,97102	33,33333333
128	4,207808581		706	0,97075	
588	4,170625583		468	0,96285	
317	4,166882184		258	0,96216	
648	4,159805942		648	0,95844	
574	4,14195211	70	254	0,95736	30
572	4,138694106		574	0,95714	
591	4,134523331		572	0,95593	



Query No:3
QT: rail,
train

Query image 3

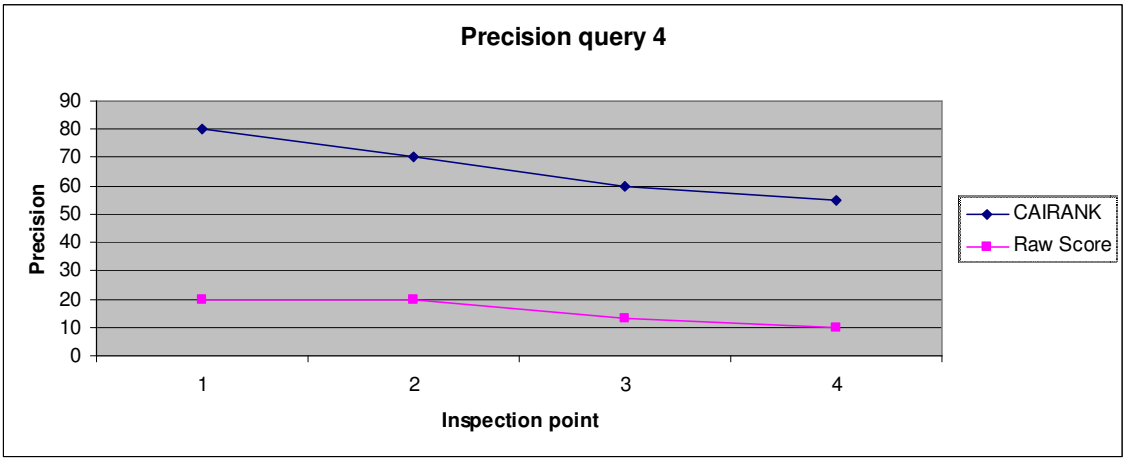
Image No:	CAIRANK Score:		Image No:	Raw Merge Score:	Precision:
455	4,810765976		4	1	
457	4,606907697		327	1	
453	4,54396137	100	130	0,99521	0
739	4,460606761		606	0,99255	
247	4,349658558		783	0,98049	
745	4,15001666		746	0,97898	
452	4,022303941		420	0,97444	
254	3,908225523	90	487	0,97387	10
458	3,90358873		636	0,97104	
454	3,850033604		306	0,97066	
248	3,81856886		204	0,97017	
735	3,772887576		330	0,96803	
244	3,753678033	86,66666667	374	0,96385	6,666666667
250	3,640019079		528	0,96189	
459	3,383483584		382	0,96129	
773	3,373978015		164	0,96108	
702	3,317677136		452	0,95765	
777	3,313651333	70	632	0,95456	5
456	3,312328283		410	0,9527	
700	3,276130851		7	0,95244	



Query No:4 **Query image 4**

QT: civil day sun

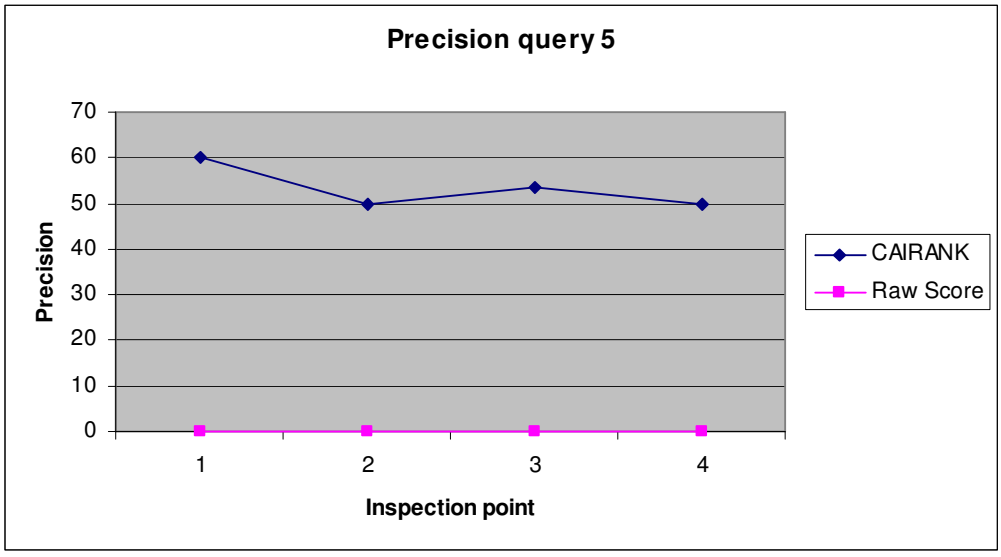
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
637	4,9898		588	1	
529	4,752959143		637	1	
19	4,750114957	80	43	0,99781	20
171	4,749441334		794	0,99705	
449	4,745075259		614	0,99049	
429	4,712591661		82	0,98787	
455	4,699717977		258	0,97961	
537	4,68267781	70	774	0,97929	20
457	4,677463469		532	0,97657	
1	4,675143212		44	0,97552	
195	4,661895293		706	0,97176	
391	4,650917733		128	0,97057	
691	4,632729912	60	796	0,96967	13,33333333
5	4,628513531		648	0,96911	
471	4,628164245		445	0,96782	
453	4,621078729		42	0,9646	
165	4,620978933		490	0,96372	
501	4,611024282	55	228	0,96339	10
655	4,598075751		318	0,96242	
693	4,597052842		316	0,96106	



Query No:5
QT:
Construction,
Building,
assembly

Query image 5

Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
577	4,9898		718	1	
769	4,704158899		577	1	
685	4,598150598	60	394	0,98599	0
217	4,520184973		122	0,98024	
219	4,467068552		716	0,98009	
379	4,352702336		796	0,97818	
532	4,305483325		228	0,97811	
191	4,268399665	50	720	0,97523	0
215	4,172370964		741	0,97397	
580	4,152347421		687	0,9677	
516	4,112123146		532	0,96629	
515	4,098821312		44	0,95908	
686	4,072291686	53,33333333	488	0,95763	0
784	4,061490797		42	0,95637	
642	4,028825285		490	0,94434	
132	3,991749667		316	0,9429	
348	3,988432476		238	0,94055	
110	3,977655059	50	90	0,94029	0
540	3,970116264		242	0,93984	
785	3,95990528		704	0,93848	

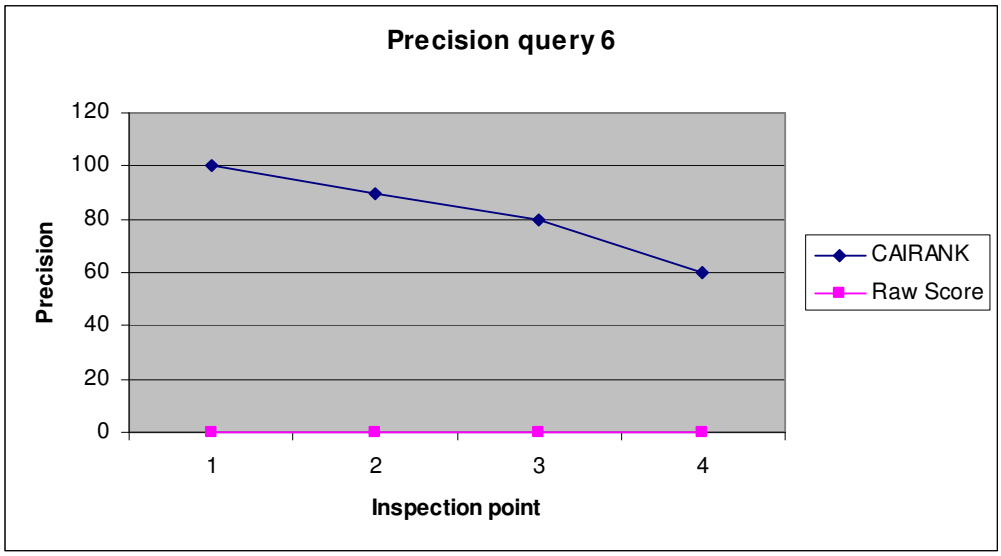


Query No:6

QT: hollow box girder, steel girder

Query image 6

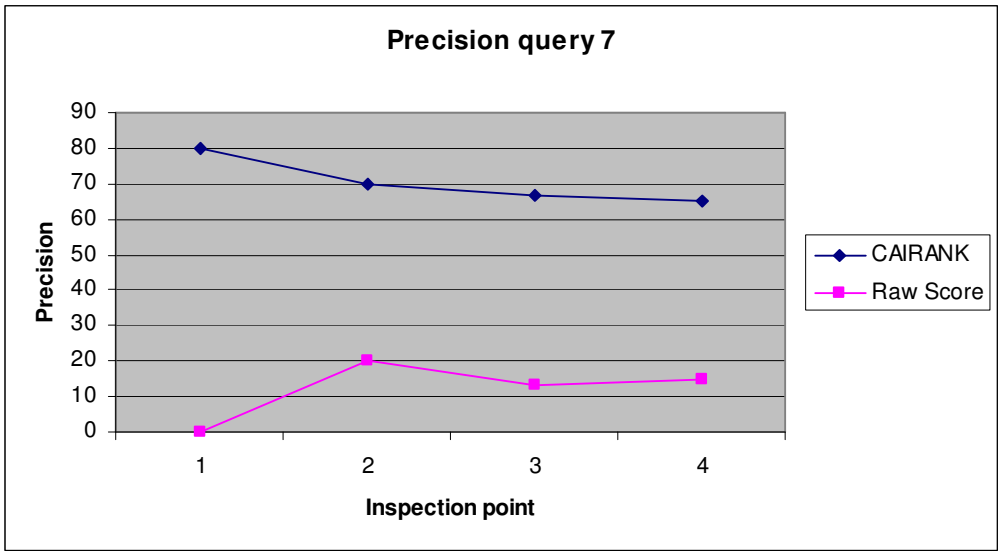
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
607	4,790731929		106	1	
559	4,374657456		659	1	
605	4,329275225	100	32	0,99654	0
606	4,304199965		478	0,9942	
555	4,270195993		408	0,99416	
612	4,248975447		16	0,99315	
557	4,165909173		746	0,99149	
601	4,005337409	90	50	0,99148	0
609	3,958458238		528	0,98697	
9	3,904867786		320	0,98376	
611	3,875003833		380	0,98153	
785	3,870737554		302	0,98144	
255	3,858263054	80	738	0,98144	0
525	3,84139753		266	0,9802	
560	3,794130203		132	0,9752	
608	3,756505226		752	0,97444	
461	3,719546614		58	0,97406	
157	3,701059405	60	200	0,97371	0
789	3,694098634		458	0,97153	
503	3,671419993		9	0,97139	



Query No:7
QT: Norway, China, Japan

Query image 7

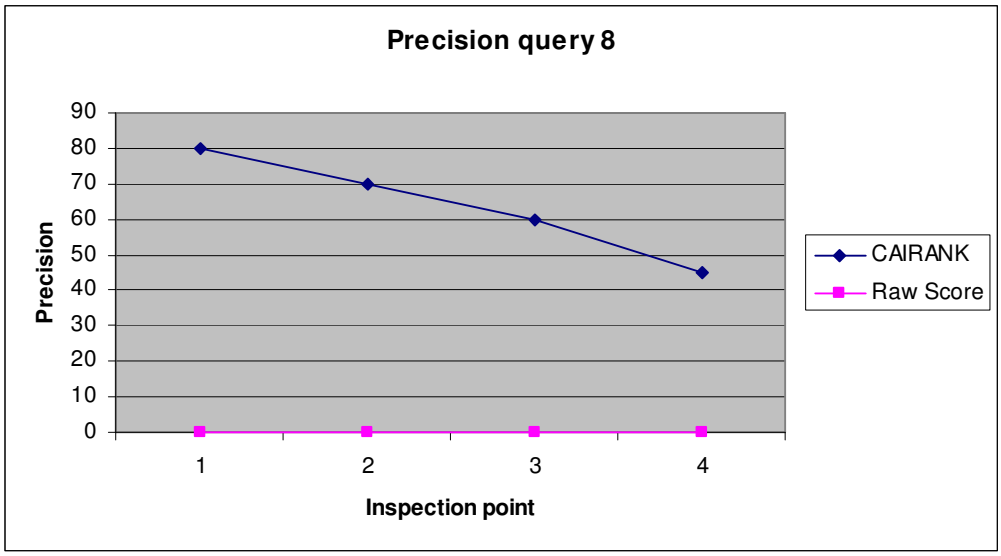
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
741	4,564120162		660	1	
744	4,268523472		465	1	
465	4,113940406	80	234	0,98748	0
742	4,034054873		489	0,97598	
738	3,892183599		90	0,97585	
745	3,881365828		475	0,9682	
13	3,872683576		13	0,9565	
233	3,777328498	70	456	0,95453	20
740	3,639513258		233	0,95019	
732	3,52709865		744	0,94458	
736	3,467808359		488	0,94338	
734	3,465600455		444	0,94233	
234	3,431407386	66,66666667	418	0,93976	13,33333333
719	3,428691172		594	0,93844	
717	3,172564738		360	0,92381	
261	3,041183304		110	0,92268	
718	3,022486068	65	402	0,92216	
720	3,006391382		650	0,92181	15
235	3,002537303		718	0,91939	
232	2,999762219		552	0,91671	



Query No:8 **Query image 8**

QT:
illuminated,
illumination

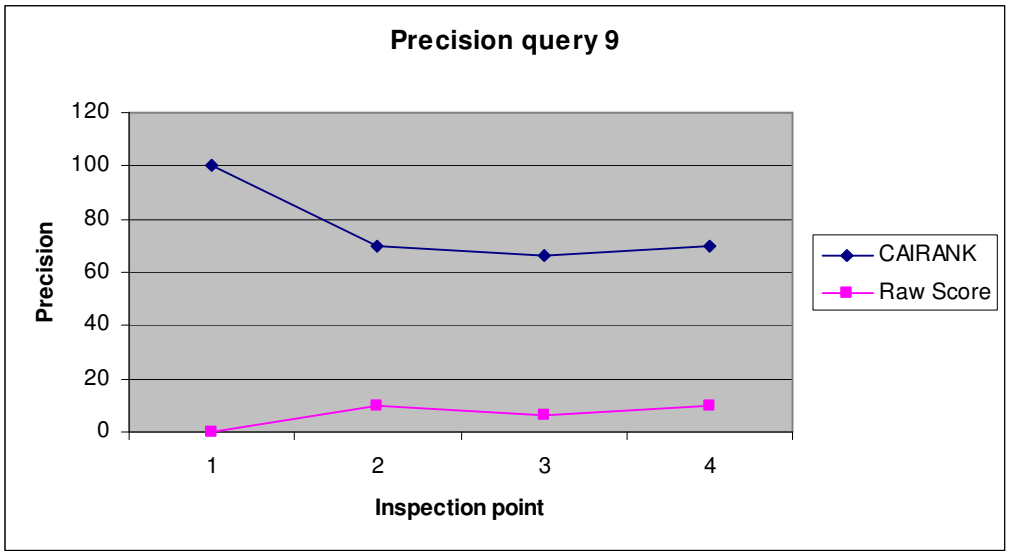
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
600	4,125198213		482	1	
541	3,968512685		573	1	
648	3,931872759		616	0,98647	0
792	3,879726229	80	502	0,98645	
44	3,822378432		605	0,97338	
614	3,81000374		148	0,97044	
400	3,754770855		252	0,95681	
533	3,712585843	70	26	0,95373	0
588	3,668082742		450	0,95187	
490	3,64108254		240	0,95018	
524	3,638146987		264	0,9497	
137	3,59490141		656	0,94904	
135	3,561594495	60	510	0,94878	0
497	3,529984112		188	0,94708	
716	3,499068071		504	0,94622	
252	3,476420716		312	0,94506	
616	3,472196934		558	0,94273	
240	3,457085353	45	308	0,94173	0
510	3,437927155		332	0,93755	
308	3,431521131		434	0,93684	



Query No:9 **Query image 9**

QT: commit suicide

Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
231	4,789509428		196	1	
201	4,775363345		289	1	
81	4,747894496	100	409	0,99746	0
221	4,6978967		314	0,99556	
237	4,57414966		604	0,99371	
223	4,428098214		628	0,99185	
289	4,358565351		674	0,99123	
653	4,336285894	70	328	0,98986	10
207	4,335362781		624	0,98569	
655	4,322863332		300	0,98458	
654	4,322563261		246	0,98405	
235	4,283418963		272	0,98202	
229	4,271094157	66,66666667	732	0,98147	6,666666667
220	4,269815139		554	0,9803	
214	4,257483189		442	0,97934	
652	4,241301636		538	0,97845	
651	4,22960397		274	0,97734	
205	4,229080041	70	472	0,97721	10
84	4,212899837		696	0,97484	
92	4,212899837		284	0,97457	

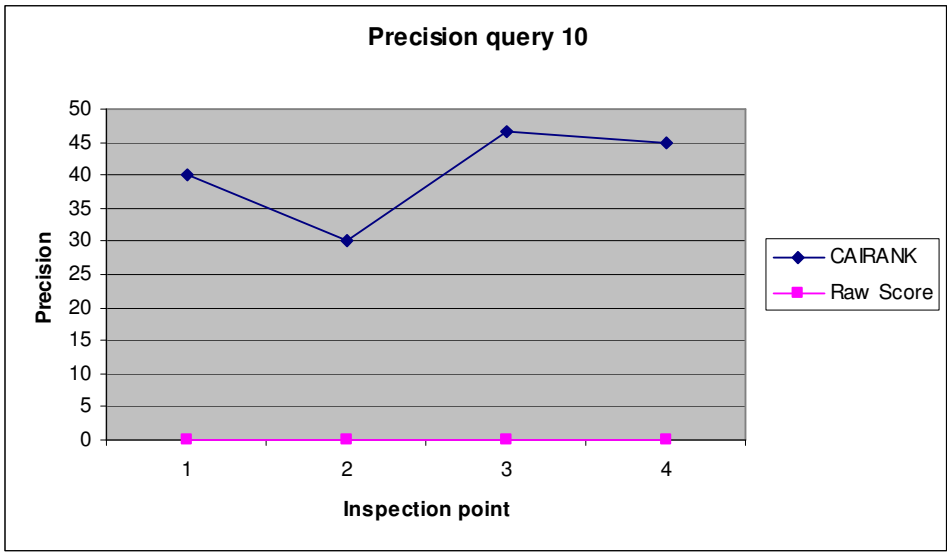


Query No:10

QT: collapse, galloping in the wind

Query image 10

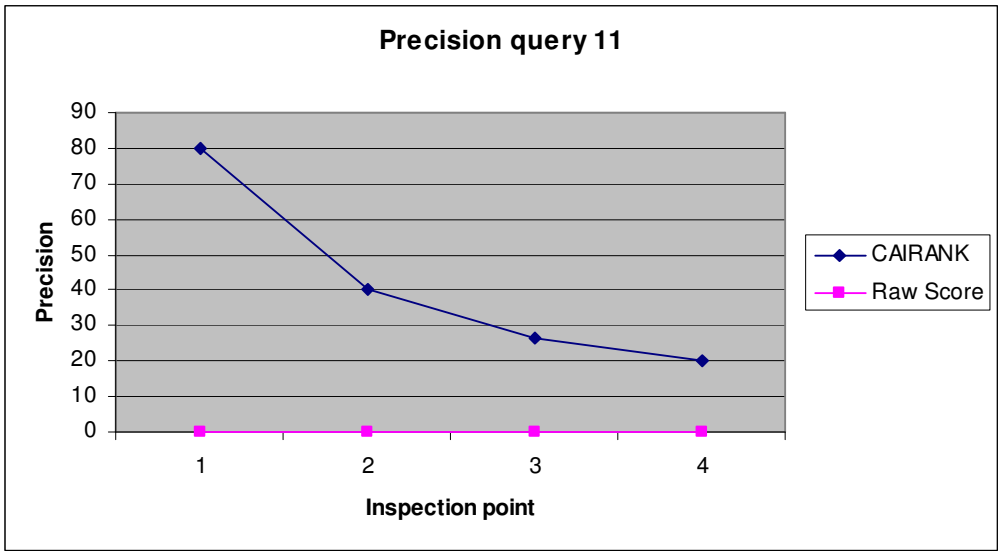
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
674	4,145820224		274	1	
668	4,017091711		327	1	
327	3,912452282	40	340	0,9935	0
734	3,769083588		658	0,99027	
732	3,742534892		626	0,9871	
712	3,740366542		250	0,98206	
1	3,712585843		712	0,97683	
739	3,678904693	30	654	0,97444	0
736	3,657934486		136	0,97336	
663	3,656974522		300	0,97289	
666	3,652249092		19	0,96984	
670	3,644090619		750	0,96793	
328	3,602161609	46,66666667	384	0,9673	0
661	3,595699778		734	0,96639	
664	3,593066504		430	0,96536	
5	3,584772116		272	0,96482	
7	3,562517608		40	0,9645	
672	3,56038663	45	386	0,96386	0
662	3,537357325		442	0,96334	
162	3,520533797		608	0,96289	



Query No:11 **Query image 11**

QT: Europe, Asia

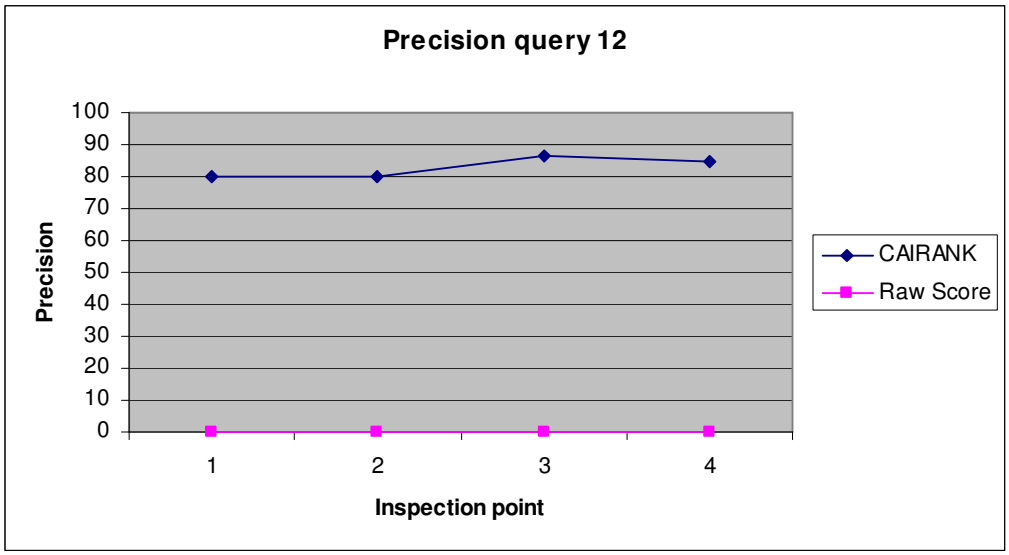
Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
253	4,630883686		432	1	
653	4,552868163		79	1	
87	4,485605659	80	196	0,99139	0
83	4,463276304		772	0,98827	
81	4,454369511		538	0,98812	
9	4,325757416		56	0,982	
251	4,320817514		619	0,9787	
446	4,185119347	40	57	0,97837	0
3	4,165609785		46	0,97746	
470	4,160483484		732	0,97492	
505	4,148070638		604	0,97464	
432	4,135747527		328	0,97344	
444	4,133949051	26,66666667	314	0,97222	0
196	4,115790625		674	0,96855	
472	4,109511289		624	0,96855	
440	4,090664151		246	0,9676	
56	4,084503809		554	0,96739	
46	4,067017763	20	294	0,96617	0
624	4,06685174		8	0,96595	
166	4,051002852		270	0,96343	



Query No:12
QT: tide, low tide, high tide

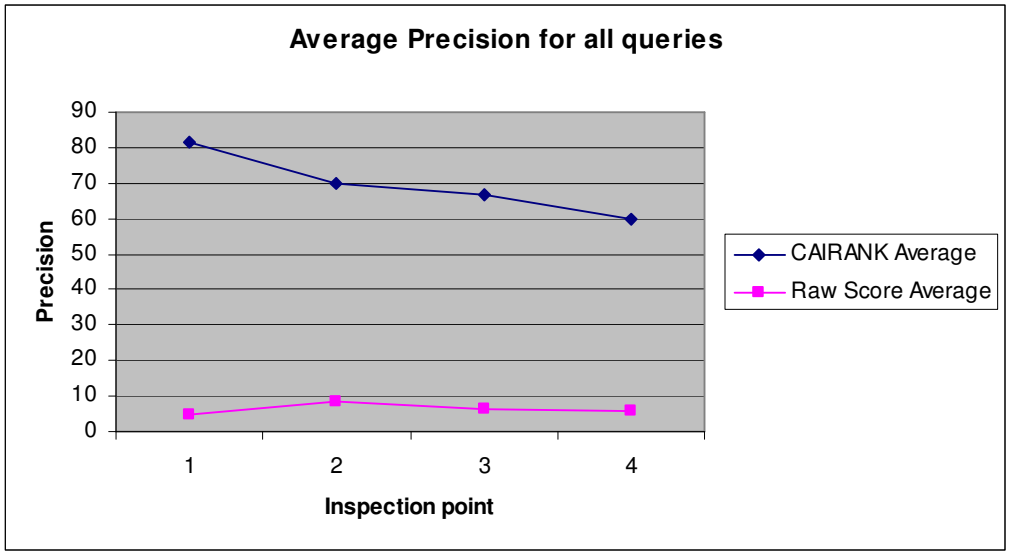
Query image 12

Image No:	CAIRANK Score:	Precision:	Image No:	Raw Merge Score:	Precision:
529	4,823440068		766	1	
285	4,785193251		195	1	
693	4,705481196	80	422	0,99995	0
287	4,624172405		548	0,99192	
691	4,514197213		91	0,98482	
281	4,366050051		420	0,98443	
618	4,347337432		449	0,97827	
571	4,275984161	80	618	0,97585	0
606	4,214159393		130	0,97265	
746	4,127509724		19	0,97245	
591	4,082454768		445	0,96785	
672	4,080549716		374	0,96568	
282	4,075412149	86,66666667	427	0,96556	0
568	4,069865544		637	0,96299	
200	4,00713122		798	0,96174	
611	3,981561012		43	0,95751	
144	3,981488085		259	0,95656	
50	3,97283349	85	404	0,95292	0
751	3,934806586		327	0,95175	
609	3,884259912		452	0,95017	



Average precision for the two approaches

	Average precision CAIRANK		Average precision Raw Score merge
Inspection point	**Precision**		**Precision**
5	81,66666667		5
10	70		8,333333333
15	66,66666667		6,111111111
20	59,58333333		5,833333333



Appendix I – Media CD

This appendix consists of an overview of the contents of an enclosed CD containing additional information. A list of the contents is given below. The CD itself is attached to the inside of the back cover.

CD contents:

Image Collection
Context descriptions
All results from manually submitted queries