Bayesian Approaches to Content-Based Image Retrieval

Georgios Andrea Stefanou

A Thesis presented for the Degree of
Doctor of Philosophy

Department of Statistics
Trinity College Dublin
Ireland

July 2005

Supervisor: Dr. Simon P. Wilson

Στους γονείς μου

Declaration

This thesis is entirely my own work.

The contents of this thesis have not been submitted as an exercise for a degree at this or any other University.

The contents of this thesis may be lent or copied by the Library at Trinity College Dublin.

Georgios Andrea Stefanou

Abstract

This thesis addresses some issues in the relatively new field of Content-Based Image Retrieval. Content-based image retrieval is a technique that uses the visual content of images to aid searches from large scale image databases. The field of content-based image retrieval is growing in importance as image archives grow in size in many fields, and as a means to search for visual content on the Web.

In this thesis we investigate Bayesian approaches to content-based image retrieval. Starting from the work of Cox et al. (1996) and Cox et al. (2000), we propose a retrieval system that attempts to capture properties of the visual content of images and how a user conducts a search. Decision theory is applied to the problem of selecting images to retrieve. The system is evaluated using a variety of tests with users.

Acknowledgnents

The first person I would like to thank is my supervisor, Dr. Simon P. Wilson. I have been working with him in the Department of Statistics since December 2000, when I started this project. During these years he has helped and assisted me in many ways. His enthusiastic engagement in my research and his never-ending stream of ideas have been absolutely essential for the results presented here. I am very grateful that he has spent so much time with me discussing different problems ranging from philosophical issues down to minute technical details.

This research was made possible by a European Commission grant through the MOUMIR Research Training Network. I would also like to thank the chief co-ordinator of MOUMIR project Dr. Anil Kokaram who gave me the opportunity to work in such an environment.

Also, I would like to thank my sister Xenia A. Stefanou for reminding me of the idea of simple statistical classification.

Finally I would like to thank my colleagues and family, particularly my parents, for their economic and emotional support throughout.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is CBIR?

Content-based image retrieval is a technique that uses the visual content of images to search in large scale image databases according to a user's interests. It has been an active and fast advancing research area since the 1990s. The earliest use of the term *content-based image retrieval* in the literature seems to have been by Kato (Eakins and Graham, 1999), to describe his experiments into automatic retrieval of images from a database by colour and shape characteristics. The term has since been widely used to describe the process of retrieving desired images from a large image collection on the basis of features (such as colour, texture) that can be automatically extracted from the images themselves.

1.2 The importance of CBIR

The twentieth century has witnessed uparalleled growth in the number, availability and importance of images. Image data is now important in many areas of life such as medicine, journalism, advertising, design, education and entertainment. Photograph libraries, art galleries and museums now see the commercial and artistic advantages of making their collections available in electronic form and many now do so. The creation of the World Wide Web in the early 1990s, enabling users to access data in a variety of media from anywhere on the planet, has provided a further massive stimulus to the exploitation of digital images.

The process of digitisation does not in itself make image collections easier to manage. Some form of cataloguing and indexing is still necessary; the only difference being that it is now possible to automatically derive much of the required information from the images without human intervention.

Efficient storage and retrieval of images is vital for the management of large image collections such as picture libraries and design archives. This message was reinforced by a workshop sponsored by the USA's National Science Foundation in 1992 that looked at the issues involved in managing visual information in some depth. It concluded that images were indeed likely to play an increasingly important role in electronically-mediated communication but that significant research advances, involving collaboration between a number of disciplines, was needed before image providers could take full advantage of the opportunities offered. In particular, research in data representation, feature extraction and indexing, image query matching and user interfacing was deemed important.

One of the main problems they highlighted was the difficulty of locating a desired image in a large and varied collection. While it is perfectly feasible to identify a desired image from a small collection simply by browsing, more effective techniques are needed with collections containing thousands of items. Journalists requesting photographs of a particular type of event, designers looking for materials with a particular colour or texture, and engineers looking for drawings of a particular type of part, all need some form of access by image content. It is this problem that forms the basis of the work in this thesis.

1.3 Research problems in Image Retrieval

In spite of remarkable progress in computer image processing in the last decade, there remain many challenging research problems in image retrieval that continue to attract researchers from many disciplines. One challenging research problem is scene recognition where humans can recognise objects in scenes, e.g. "There is a table". They can recognise aspects of images, e.g. "There is a smile on that person's face"; but computers simply cannot do this in all generality. If computers were given the facility that humans have, they would be able to process large number of images looking for ones satisfying given semantic clues.

The research leading to algorithms and data-structures that will finally make CBIR possible broadly will draw on a number of disciplines other than computer science. Obviously computer vision will have a significant role to play, as can image processing from a statistical point of view including appropriate forms of inference such as Bayesian inference. Input from cognitive neuroscience will shed light on how the brain "sees" and "retrieves", and psychology will provide knowledge of attentive processing.

All of these fields are required because the semantic content of images is not as obvious as it is with text. Interrogation of image databases (including video and film) is in many cases likely to be through natural language queries so the relationship between language and image will need to be understood. It is clear that the searching process will have to be highly interactive, including significant feedback. The balance between what can be left to the computer and where human intervention will remain essential is unknown. For example, it is clear that the human brain has the facility to bring into play background information when it is "seeing" and use it for retrieval. The development of models and collection of data on this issue will require substantial experimentation.

The research presented in this thesis is statistical. It concentrates on the learning algorithm that a computer uses to retrieve images from a database based on a simple form of interaction with the user. We present experiments that show how effective the algorithm is in modelling certain semantic content. In this way we hope to contribute to progress on solving these difficult research problems.

1.4 Applications

There are many applications where full visual retrieval is important. Some promising fields of application for still images include, see (Bimbo, 1999), (Eakins and Graham, 1999):

- Art galleries and museum management: a user may wish to retrieve all paintings that have some similarity with a reference painting, to look for commonalities and/or influences between artists as to the use of colours, spatial arrangement of forms and representation of subjects. Art historians may be interested in finding images based on their semantics. For example, they can consider the meaningful sensations that a painting provokes, according to the theory that the appropriate arrangement of colours on a canvas determines different psychological effects in the observer.

- architectural and engineering design: design documents can be queried for design details or a part of design,

- remote sensing and management of earth resources: images can be inspected in order to retrieve objects or regions with a certain property,
- geographic information systems: 2D maps can be inspected in order to retrieve all geographic entities in the surrounding of a certain location, or all geographic entities in a certain spatial relationship with respect to another entity,
- fabric and fashion design: fashion designers may want to inspect patterns from a large collection which look similar to a reference pattern to develop new ideas,
- trademark and copyright database management: to check if the same or a similar trademark already exists,
- law enforcement and criminal investigation: police may be interested in identifying criminals by matching a face image with those stored in the database,
- picture archiving: a user may want to inspect photos of a certain subject (horses, landscapes, etc.) in order to request a copy of them for publication. Media organisations to search large photograph databases for similar images of persons, events or objects.

1.5 About MOUMIR

The research described in this thesis was carried out within a European Commission Research Training Network called MOUMIR: MOdels for Unified Multimedia Information Retrieval. The MOUMIR network was composed of six university research laboratories and two industrial partners: Trinity College Dublin, the network coordinators; INRIA

Sophia-Antipolis; Ben Gurion University; Aristotle University Thessaloniki; Cambridge University; INESC Porto; RTP Lisbon and Bridgeman Art Library London.

MOUMIR's principal aim was to investigate methods for content based multimedia information retrieval. This covered a broad range of current research areas which were divided into three categories or Work Packages of audio, video and image processing. Various partners chose to concentrate on particular elements of the packages depending on their research expertise. Ben Gurion and a section of the Cambridge group principally focused on audio while Trinity (Department of Electronic and Electrical Engineering), INESC, RTP and Thessaloniki concentrated on video. INRIA and the remainder of the Cambridge and Trinity (Department of Statistics) performed still image analysis and it is within this area that the research described in this thesis falls.

1.6 Datasets

Two image databases were examined in the still image work package. The Institut Geographique National (IGN) provided a set of aerial images of the Ile-de-France region around Paris and the Bridgeman Art Library (BAL) supplied a database of fine art paintings. Much of the research is described in the context of the BAL database.

1.6.1 L'Institut Gèographique National

The first dataset we consider is a collection of 160 aerial images of the Ile-de-France region around Paris created by the Institut Gèographique National (IGN), the French Mapping Institute. An example is shown in Figure 1.1. Each of the images shows a land area of 4.6km by 4.6km, and is 500 by 500 pixels. There are many classes of statement that one could consider making about such images, but one of the most important con-

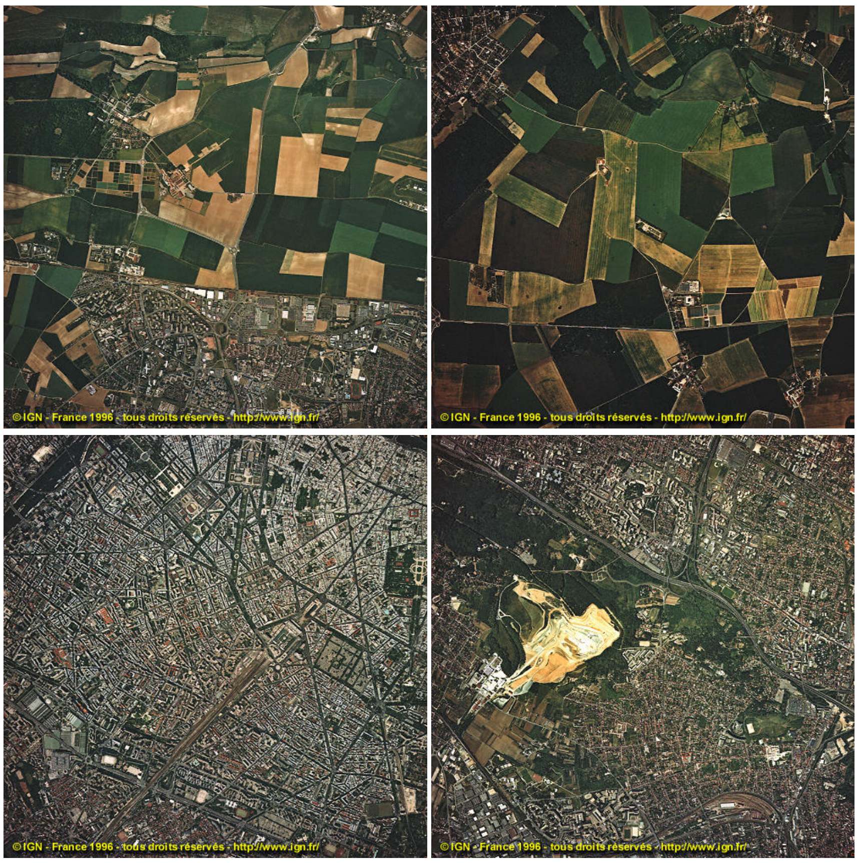

©IGN - France 1996 - tous droits réservés - http://www.ign.fr/
©IGN - France 1996 - tous droits réservés - http://www.ign.fr/
©IGN - France 1996 - tous droits réservés - http://www.ign.fr/
©IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 1.1: Examples of the IGN aerial images.

sists of statements about land use. Such statements essentially involve a map from the image domain into a finite set of classes: “forest”, “urban area”, “agricultural field” and so on. A set of such maps is available from the Institut d’Amènagement et d’Urbanisme de la Règion d’Ile-de-France (IAURIF), who compiled them (independently of the IGN images) using field studies and existing cartography.

1.6.2 Bridgeman Art Library

The second dataset is a collection of images of paintings from the Bridgeman Art Library (BAL). Based in the United Kingdom, BAL is a commercial art library supplying electronic and hard copy images to magazines, newspapers, designers and others. The images are realistic in intent, but in many cases the colours and forms do not correspond to "photographic realism". It is very hard to characterize the queries faced by the staff at BAL. The queries are often phrased at a very high semantic level, and the process of answering queries is complex, involving prolonged interaction with clients. This makes it a very challenging environment for CBIR systems. Example images are shown in Figure 1.2.

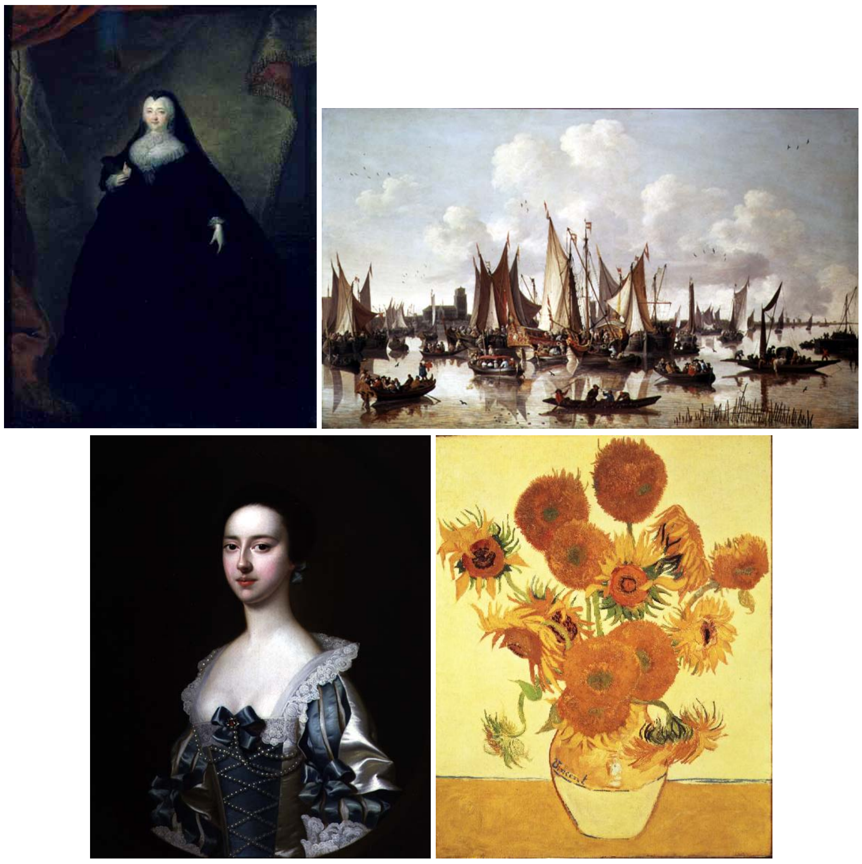


Figure 1.2: Examples of the BAL images.

1.6.3 Contributions of the Thesis

The new contributions of this thesis are:

1. Development and implementation of a Bayesian content-based image retrieval system, that extends the work of Cox et al. (2000), making use of relevance feedback.

2. Development of an open-ended browser, suitable for applications such as graphic design.

3. The use of decision theory to propose a solution to the problem of which images to display to the user. Several proposals, based on different priorities for the retrieval system, are proposed.

4. Evaluation of the retrieval system using the idea of target testing, and evaluation of the system's ability to capture emotional content of an art image database.

1.6.4 What is an Image?

A digital image is a matrix of pixel values $X = \{x_{ij} | i = 1, ..., n; j = 1, ..., m\}$. Each pixel represents the colour at that point in the image. Colour is defined by a 3-vector, most commonly by the intensities of red, green and blue in the colour, thus $x_{ij} = (R_{ij}, G_{ij}, B_{ij})$. However, other representations are possible and indeed useful; see Section 3.3.

1.7 Thesis Outline

The research discussed in the thesis builds upon this introduction. The main contributions of this dissertation are contained within Chapters 5, 6 and 7. The individual chapters are structured as follows:

Chapter 2

This chapter describes the first piece of work that was completed for the MOUMIR project. It represents a use of existing classification techniques applied to a particular image database. It is in many ways separate from the work described in succeeding chapters, when we consider image retrieval in more depth.

Chapter 3

This chapter presents the features that were extracted from images to perform CBIR search.

Chapter 4

This chapter describes the general mechanism of CBIR as well as the general issues arising from CBIR. It reviews some of the most well known CBIR systems and concludes with a description of the Bayesian Image Retrieval System Pichunter (Cox et al., 2000), on which this work is based. An implementation of Pichunter to our two image databases is also presented.

Chapter 5

In this chapter we propose some extensions based on the current PicHunter model as it appeared in Cox et al. (2000). Those proposals have to deal with inserting some new parameters to the current model. The computation of those new models are not trivial. Methods such as direct Monte Carlo and direct evaluation of a posterior on a grid were tried, with the latter approach found to be better.

Chapter 6

This chapter describes a decision-theoretic solution to the display strategy problem for the Bayesian image retrieval system that we are concerned with. The usual strategy is the most probable scheme e.g. display images with high posterior probability, which we are going to present from a decision- theoretic perspective. While the most probable scheme is a reasonable strategy for target-specific search, for category search or open-ended browsing, where users search through a database with a rather broad, nonspecific goal in mind, it may not be appropriate to display the most probable images to a user, because it does not allow us to capture a user's needs in a broader sense. For example, a user may not initially have the query image at hand or the ideal query may evolve during the retrieval process. In other words, by developing other display strategies we are proposing an open-ended search browser rather than a target-specific search system. This contribution fills the need for such a system in applications in graphics, photojournalistic and advertising.

Chapter 7

The first part of this chapter has to deal with image retrieval evaluation. Once a content-based image retrieval application has been developed, the next crucial problem is how to evaluate its performance. We also investigate the feasibility of using visual browsing for the retrieval of emotional content of BAL pictures using our proposed model.

Chapter 8

This chapter concludes the dissertation with a review of the key components of the research described. The important contributions of this work are highlighted with suggestions of how this body of work should proceed in the future.

Appendices

Two appendices are also included. They relate to Chapter 6 in the main body of the thesis.

Chapter 2

Image Classification

2.1 Introduction

This chapter describes the first piece of work that was completed for the MOUMIR project. It represents a use of existing classification techniques applied to a particular image database. It is in many ways separate from the work described in succeeding chapters, when we consider image retrieval in more depth. However, since image retrieval makes use of image segmentation and the work is an interesting application, we include it here.

Image Segmentation is the process of classifying the pixels of an image into distinct areas according to objects or activities in it. Segmentation is often the critical step in image analysis: the point at which we move from considering each pixel as a unit of observation to working with *objects* (or parts of objects) in the image, composed of many pixels. If segmentation is done well then many other image analysis procedures, that rely on a good segmentation, are made simpler. The following work is concerned with the classification of several IGN (Institut Géographique National) satellite images of the Ile de France area, that were supplied to MOUMIR.

In remote sensing, image classification is the process used to produce thematic maps from imagery. In this chapter we review some image classification techniques as well as their advantages, disadvantages and their limitations. Supervised classification is used, which requires a priori knowledge of the number of classes, as well as knowledge concerning statistical aspects of the classes (variability, size etc.). We identified 4 classes for training data: town and three different types of vegetation. Eighteen statistical features were extracted from them. After, a separability analysis was performed, helping to find those features that have the most separability between the 4 classes and as a result reducing the size of feature descriptors for each class to 10. Finally, the minimum distance algorithm was used to classify the image pixels into the 4 classes. We also present the thresholding method for image classification as well as a post classification procedure.

CBIR is one application of image segmentation. One example query for images such as those considered here, could be: "give me a satellite image with at least 30% urban area". Indeed, one could argue that a good segmentation of satellite images, with well-identified classes, provides a complete semantic description of the image and hence one should be able to retrieve an image very well based on any query about the classes in it. However, queries such as: "give me a satellite image with at least 30% green land area" are not directly addressed by a segmentation into thematic classes.

2.2 Image Classification

Digital image classification is the process of assigning pixels to classes. By comparing pixels to one another and to those of known identity, it is possible to assemble groups of similar pixels into classes that match the informational categories of interest to users of image data. These classes form regions on a map or an image. After classification, the digital image is presented as a mosaic of regions, which are typically identified by a colour or symbol. We can see in Figure 2.1 the classified image (right) is defined by examining the real image, then grouping together those pixels that have similar values. Here class A (sky) is formed from bright pixels, and class C (land) is formed from dark pixels. In principle, these classes are homogeneous; pixels within classes are more similar to one another than they are to pixels in other classes. In practice, of course,

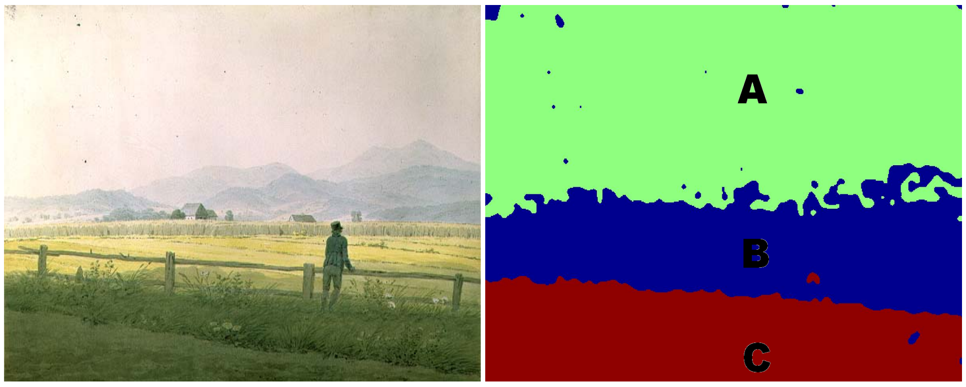


Figure 2.1: Real image (left) and classified image (right).

each class will display some diversity, as each scene will exhibit some variability within classes.

Image classification has formed an important part of the fields of remote sensing, image analysis, and pattern recognition. In some instances, the classification itself may form the object of the analysis. For example, classification of land use from remotely sensed data produces a map-like image that forms the final product of the analysis. In other instances, the classification may form only an intermediate step in a more elaborate analysis like image retrieval.

The term *classifier* refers loosely to a computer program that implements a specific procedure for image classification. Over the years many classification strategies have been devised. From these alternatives the analyst must select the classifier that will best accomplish a specific task. At present it is not possible to state that a given classifier is "best" for all situations because characteristics of each image and the circumstances for each study vary so greatly.

There are two basic approaches to classification. One requires a user to help by indicating examples of each type of ground cover to be used for the classification (supervised classification). The other attempts to divide the image into classes which have distinct properties (unsupervised classification); see Campbell (2002).

2.3 Supervised Classification

Supervised classification can be defined informally as the process of using samples of known identity (i.e., pixels already assigned to informational classes, for example, the different kinds of geological units, forest, or land) to classify pixels of unknown identity (i.e. to assign unclassified pixels to one of several informational classes). *Training areas* or *training fields* are identified, that the analyst believes to contain only a region of known identity. Samples of pixels are taken from these areas.

Such areas must be homogeneous in respect to the informational category to be classified. That is, training areas should not include unusual regions, nor should they straddle boundaries between categories. Size, shape, and position must favour convenient identification both on the image and on the ground. Clearly, the selection of these training data is a key step in supervised classification; see Campbell (2002).

2.3.1 Advantages

The advantages of supervised classification are:

- The analyst has control of a selected menu of informational categories tailored to a specific purpose and geographic region. This quality may be vitally important if it becomes necessary to generate a classification for the specific purpose of comparison with another classification of the same area at a different date or if the classification must be compatible with those of neighboring regions.
- Supervised classification is tied to specific areas of known identity, determined through the process of selecting training areas;
- The problem of matching patterns on the final map with the informational categories of interest does not arise because this task has been addressed during the process of selecting training data;
- Serious errors in classification may be detected by examining training data to determine if they have been correctly classified by the classification procedure. Inaccurate classification of training data indicates serious problems in the classification or selection of training data, although correct classification of training data does not always indicate correct classification of other data.

2.3.2 Disadvantages and Limitations

Supervised classification can be thought of as discriminant analysis for images. It shares the many disadvantages of discriminant analysis such as:

- classification structure is imposed by the analyst upon the data which may not match the natural classes that exist within the data, and therefore may not be distinct or well defined in the data space;

- In satellite imaging, training data are often defined primarily with reference to informational categories and only secondarily with reference to spectral properties. A training area that is "100 % forest" may be accurate with respect to the "forest" designation, but may still be very diverse with respect to density, age, shadowing, and the like, and therefore form a poor training area;

- Poor selection of training data by the analyst may lead to a bad representation of the class. This may be true despite the best efforts of the analyst, especially if the area to be classified is large, complex, or inaccessible;

- Conscientious selection of training data can be a time-consuming, expensive, and tedious undertaking, even if ample resources are at hand. The analyst may experience problems in matching prospective training areas as defined on maps and aerial photographs to the image to be classified;

- Special or unique categories can not be discovered because they are not in the training data; this can happen because they are not known to the analyst or because they occupy very small areas on the image.

2.3.3 Methods for Supervised Classification

A variety of different methods have been devised to implement the basic strategy of supervised classification. All use information derived from the training data as a means of classifying those pixels not assigned to training fields. The following sections outline only a few of the many methods of supervised classification.

- ***Minimum Distance Classification***

This method uses a vector of features of the pixel values such as mean value in a window centred on the pixel, of the image data that form the training data as a means of assigning pixels to informational categories. The feature vector of each pixel in the training fields can be plotted in feature space (Figure 2.2). Ideally,

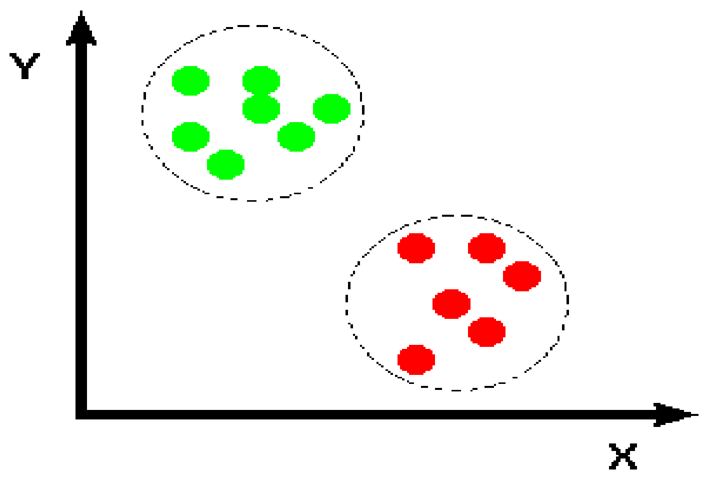


Figure 2.2: Two classes.

each class appears as a distinct cluster in feature space. These clusters may appear to be the same as those that would be defined for unsupervised classification. However, in unsupervised classification, these clusters are defined according to the "natural" structure of the data. Now, for supervised classification, these groups are formed by values of pixels within the training fields defined by the analyst.

Each cluster can be represented by its centroid, often defined as its mean value. As unassigned pixels are considered for assignment to one of the several classes, the distance to each cluster centroid is calculated and the pixel is then assigned to the closest cluster. Thus the classification proceeds by always using the "minimum distance" from a given pixel to a cluster centroid defined by the training data. Such distances could be the Euclidean distance between points x and y in a Euclidean space $\mathbb{R}^n$ which is given by,

$$d = |x - y| = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2}, \tag{2.1}$$

or the Mahalanobis distance which is given by,

$$d = \sqrt{(x - y)' S^{-1} (x - y)}, \tag{2.2}$$

where S is the sample variance-covariance matrix. Mahalanobis distance is a very useful way of determining the "similarity" of a set of values from an "unknown" sample to a set of values measured from a collection of "known" samples. It is superior to Euclidean distance because it takes into account not only the average value but also its variance and the covariance of the variables measured. Usually standarization of the data is needed; the Mahalanobis distance is preferred. Standardization of the data is needed if the range or scale of one variable is much larger or different from the range of others. This distance also compensates for intercorrelation among the variables. Euclidean distance is appropriate for variables that are uncorrelated and have equal variances.

- ***Maximum Likelihood Classification***

This kind of strategy uses the training data as a means of estimating means and variances of the classes, which are then used to estimate the probability of being in a particular class. Maximum likelihood classification assumes that a feature vector of a pixel in class i is MVN(μ_i, Σ_i). The training data for class i is used to estimate μ_i and Σ_i. Then a pixel with feature vector f has class

$$j = \arg\max_i \left(\frac{e^{-\frac{1}{2}(f - \hat{\mu}_i)' \hat{\Sigma}_i^{-1} (f - \hat{\mu}_i)}}{\sqrt{2\pi |\hat{\Sigma}_i|}} \right), \tag{2.3}$$

where $\hat{\mu}_i$ and $\hat{\Sigma}_i$ are the maximum likelihood estimates of μ_i and Σ_i. Like Mahalanobis distance, ML classification considers not only the mean, or average, values in assigning classification, but also the variability of brightness values in each class. It requires more intensive calculations, so it has the disadvantage of requiring more computer resources than the simpler technique mentioned above. Also, it is sensitive to variations in the quality of training data. Computation of the estimated probabilities is based on the assumption that both training data and the classes themselves display normal frequency distributions.

This method can of course be generalised to other likelihood functions.

- ***Bayes's Classification***

The use of a priori probabilities can be formalized using Bayesian probability theory to define the a posteriori probability that a pixel with feature vector f belongs in class c:

$$p(c|f) = \frac{p(f|c)p(c)}{p(f)} \tag{2.4}$$

where $p(f)$ is the probability of the feature vector occurring anywhere in the image. The classification process simply involves computing the a posteriori probability of the sampled feature vector being in each class and assigning pixels with that vector to the class that yields the highest a posteriori probability, Schott (1997). Note that if we assume $p(c)$ is uniform (equal a priori weight to each class) and that $p(f|c)$ is MVN then we obtain the ML classifier. This approach to classification is extremely useful and flexible, however this strategy is limited by the quality of the estimates of the probabilities required for the classification.

2.4 Unsupervised Classification

Unsupervised classification can be defined as the identification of natural groups, structures, or patterns within image data. For unsupervised classification, the analyst employs a computer algorithm that locates concentrations of feature vectors within a heterogeneous sample of pixels. These *clusters* are then assumed to represent classes in the image and are used to calculate class signatures. They remain to be identified (labeled). If we consider supervised classification to be discriminant analysis, then unsupervised classification is cluster analysis.

2.4.1 Advantages

The advantages of unsupervised classification are:

- No extensive prior knowledge of the properties of each class is required, although knowledge of the region is required to interpret the meaning of the results produced by the classification process.
- To conduct unsupervised classification, only the number of categories desired or possibly, minimum and maximum limits on the number of categories, need to be specified.
- Unsupervised classification can recognise classes with small area, that are easily missed by a supervised method, Campbell (2002).

2.4.2 Disadvantages and Limitations

The disadvantages and limitations of unsupervised classification arise primarily from a reliance upon "natural" groupings and difficulties in matching these groups to the informational categories that are of interest to the analyst.

- The classes that the method identifies do not necessarily correspond to the categories that are of interest to the analyst. As a result, the analyst is faced with the problem of matching classes generated by the classification to the classes that are required by the user of the information. Usually there is no simple one-to-one correspondence between the two sets of classes.

- The analyst has limited control over which classes are selected by the classification algorithm and their specific properties. If it is necessary to generate a specific menu of informational classes the use of unsupervised classification may be unsatisfactory.

2.4.3 Methods for Unsupervised Classification

- ***Thresholding*** Thresholding is the simplest and most commonly used method of classification (Glasbey and Horgan, 1995). Given a single threshold t, the pixel located at lattice position (i, j), with greyscale value g_{ij}, is allocated to category 1 if

$$g_{ij} \leq t. \qquad (2.5)$$

  Otherwise, the pixel is allocated to category 2. In many cases, t is chosen manually by the scientist, by trying a range of values of t and seeing which one works best at identifying the objects of interest. More than one threshold can be used, in which case more than two categories are produced.

- ***k-means clustering*** One other automatic method, which does not require a training set, is k-means clustering. The user must specify the number of classes (k) in the image. The k-means algorithm then attempts to locate the mean vector μ_i for each of the k classes. Normally the image data are subsampled by selecting every qth pixel in the image to reduce the data volume. Next, k initial estimates of the location of the mean vectors in the l-dimensional feature space are made at random. If we designate these initial estimates as μ_i'', then we can tentatively assign each pixel to a class based on how close (minimum distance to the mean) it is to the mean vectors. The mean of all the pixels tentatively assigned to the ith class becomes our new estimate of the class mean μ_i'. The sample pixels are tentatively reassigned using the new class means, and the procedure repeats in this fashion until the class means no longer change (i.e. the change is less than some threshold). At this point, the tentative means are assumed to be good estimates of the class mean vectors μ_i. All pixels in the image can then be assigned to a class using the minimum distance to the mean classifier. Alternatively the pixels assigned in the last iteration to the ith class, can be used to generate estimates of the other statistics for the class.

In practice, it is often useful to run an unsupervised k-means classifier with the initial estimate of k slightly greater than expected to "see" what types of groupings naturally occur in the image. These "extra" classes often identify features that would confuse a supervised classifier (e.g. clear and turbid water as two separate classes) or point out classes that will be very difficult to separate.

2.5 Supervised Classification for IGN images

This section describes an application of supervised classification to satellite images. Two hundred IGN greyscale images are classified using the minimum distance classification technique.

2.5.1 Greyscale Display

The natural way to display digital images is to use the pixel values to specify the brightness with which a pixel is illuminated on a computer screen or how bright the pixel appears on a printed page. We will work with *greyscale* images or *intensity* images in which we use levels of greyness to display its values. The term “grey” implies in effect that there is no colour present: all parts of the visible spectrum are equally represented. A physical measurement of the amount of the amount of light is its intensity.

Generally, there are a finite number of grey levels to which each pixel on a computer monitor can be set. On old monitors, just two levels were possible, meaning that only binary images could be displayed. On most modern monitors, at least 255 grey levels are possible (Glasbey and Horgan, 1995). There is no correct conversion from RGB to grayscale. We have used the standard NTSC (National Television Standards Committee) formula to convert a RGB image to a greyscale intensity image I:

$$I = 0.3 \times R + 0.6 \times G + 0.1 \times B \tag{2.6}$$

where R,G,B are the Red, Green and Blue colour components for each individual pixel. In Figure 2.3 , we can see the RGB and the greyscale image.

2.5.2 Training Data

Training fields are areas of known identity delineated on the digital image, usually by specifying the corner points of a rectangular area using line and column numbers within the coordinate system of the digital image. Specific training areas are identified for each class following the guidelines outlined above. The objective is to identify a set of pixels that accurately represents luminance variation present within each class (see Figures 2.4, 2.5). In this application we identify 4 classes, which we name field1, field2,

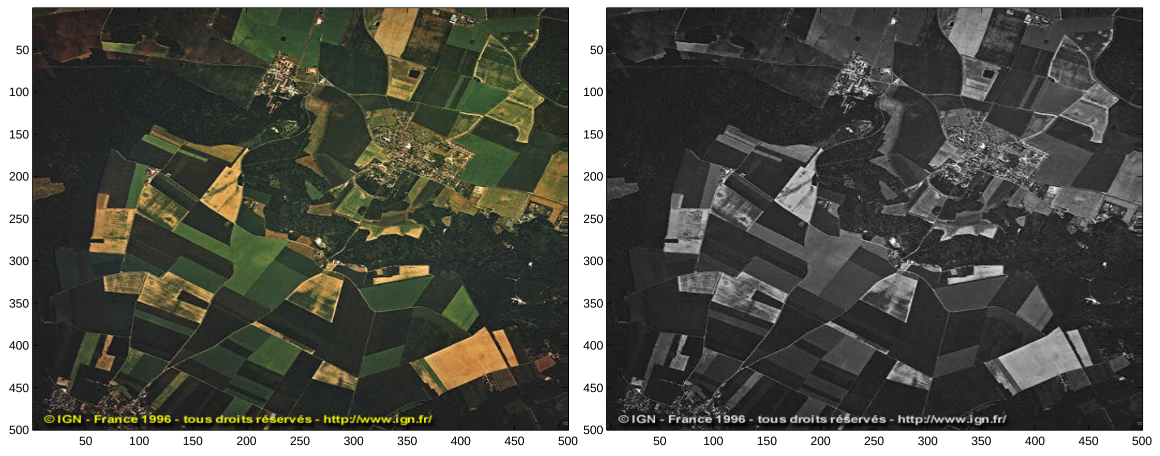

©IGN - France 1996 - tous droits réservés - http://www.ign.fr/
©IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 2.3: RGB image (left) and Greyscale image (right).

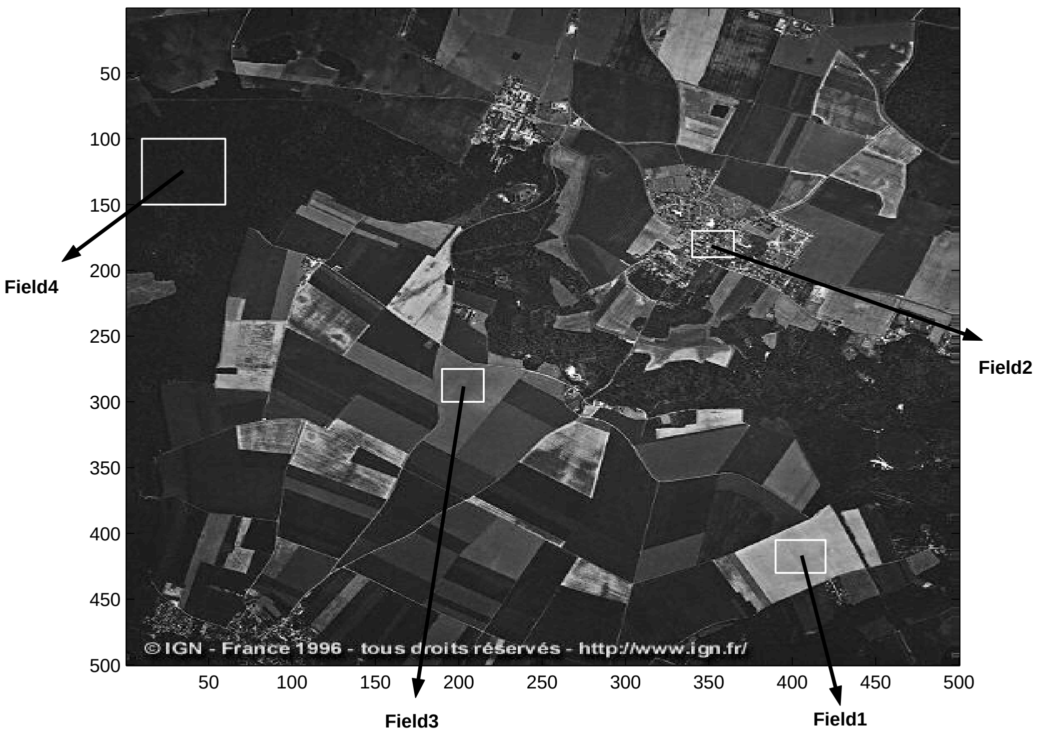
field3 and field4. The training data for these classes are composed of rectangular areas with size 26 by 26, 21 by 26, 26 by 26 and 51 by 51 pixels respectively.

An important concern is the overall **number of pixels** selected for each category. As a general guideline, the operator should ensure that several individual training areas for each category provide a total of at least 100 pixels for each category.

Sizes of training areas are important. They must each be large enough to provide accurate estimates of the properties of each class. Therefore, they must as a group include enough pixels to form reliable estimates of the characteristics of each class. Individual training fields should not, on the other hand, be too big so as to include undesirable variation.

Shapes of training areas are not important provided that shape does not prohibit accurate delineating and positioning of outlines of regions on digital images. Usually it is easiest to define rectangular, as we have already seen, or polygonal areas, as such shapes minimize the number of vertices that must be specified, which is usually the most bothersome task for the analyst.

Placement of training areas may be important. Training areas should be placed



50
100
150
200
250
300
350
400
450
500
Field4
Field2
Field3
Field1
© IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 2.4: Training fields

within the image in a manner that permits convenient and accurate location with respect to distinctive features, such as water bodies, or boundaries between distinctive features on the image. They should be distributed throughout the image so that they provide a basis for representation of diversity present within the scene. Boundaries of training fields should be placed well away from the edges of contrasting parcels so that they do not encompass edge pixels.

Perhaps the most important property of a good training area is its **uniformity**, or **homogeneity**. Data within each training area should exhibit a unimodal frequency distribution. Training areas that exhibit bimodal histograms should be discarded if their boundaries cannot be adjusted to yield more uniformity. Histograms of these training data have a single peak, indicating a degree of homogeneity (Figure 2.5).

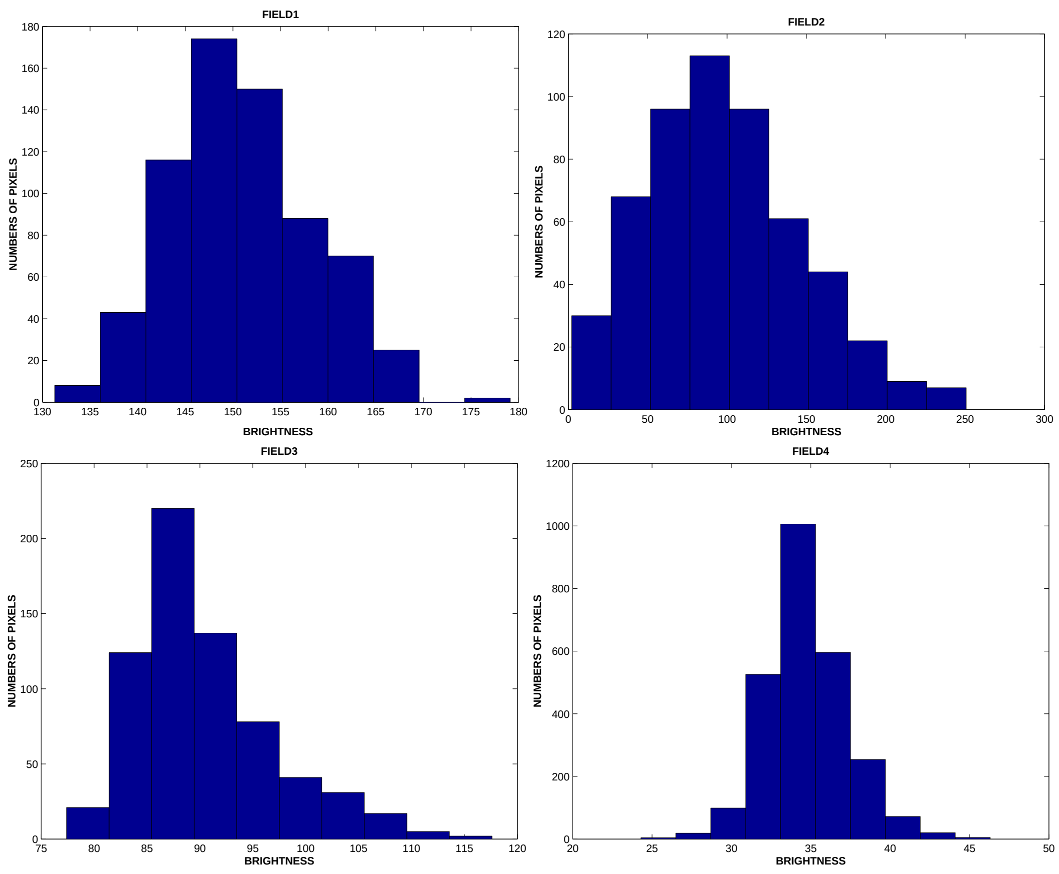


Figure 2.5: Uniformity of the training data for the 4 classes identified in the IGN image

Data from such training fields form a suitable basis for image classification.

2.5.3 Local Feature Descriptors

The aim now is to extract important *features* from each training class, from which a description, interpretation, or understanding of the scene within the class can be provided by the machine. A unique description of each classification category should be created. The description of training classes is an extremely important component of the classification process.

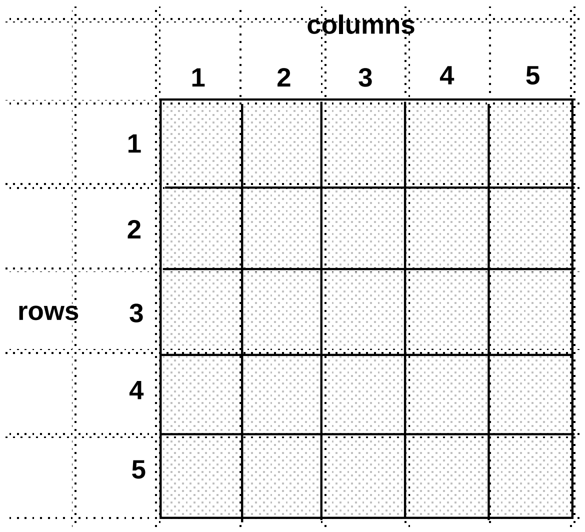


Figure 2.6: A 5×5 moving window w

These features are often calculated on a window of pixels centred on the pixel in question. In our situation we use a 5×5 moving window (w) (Figure 2.6). In what follows, w may refer to a window centered at pixel (i, j).

For each pixel, within the training area, we computed 18 features:

- **Four correlations** have been calculated using the following formula:

$$R(w1, w2) = \frac{cov(w1, w2)}{\sqrt{cov(w1, w1)cov(w2, w2)}} \tag{2.7}$$

  Vertical correlation between the window $(w1)$ with rows 1 to 4 and columns 1 to 5 and the window $(w2)$ with rows 2 to 5 and columns 1 to 5. **Horizontal correlation** between the window with rows 1 to 5 and columns 1 to 4 and the window with rows 1 to 5 and columns 2 to 5. Two diagonal correlations, **diagonal1 correlation** between the window with rows 2 to 5 and columns 1 to 4 and the window with rows 1 to 4 and columns 2 to 5. **Diagonal2 correlation** between the window with rows 2 to 5 and columns 2 to 5 and the window with rows 1 to 4 and columns 1 to 4.

- **minimum** and the **maximum** pixel grey value of the moving window.

- Another easily measured feature is **modulation**, M_{ij}, defined as,

$$M_{ij} = \frac{max(w) - min(w)}{max(w) + min(w)}; \tag{2.8}$$

  because pixel values are always positive, this definition insures that modulation is always between zero and one, and unitless (Schowengerdt, 1997).

- the **entropy**, defined as

$$H = - \sum_{p_w(x)>0} p_w(x) \log_2 p_w(x), \tag{2.9}$$

- the **median**,

- the **mad** (mean absolute deviation), defined as

$$mad = \frac{\sum_{i=1}^{n} |x_i - \bar{x}|}{n}, \tag{2.10}$$

  computes the average of the absolute differences between a set of data and the sample mean of that data,

- the **mean**, defined as

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}, \tag{2.11}$$

- the **standard deviation** defined as

$$\sigma = (\frac{1}{n-1} \sum_{i=1}^{n} -(x_i - \bar{x})^2)^{\frac{1}{2}}, \tag{2.12}$$

- the **variance** that is the square of the standard deviation (σ^2),

- the **skewness** defined as

$$S = \frac{E(x - \bar{x})^3}{\sigma^3}, \tag{2.13}$$

where $E((x-\bar{x})^3)$ represents the mean of the $(x_i-\bar{x})^3$. The skewness is a measure of the asymmetry of the data around the sample mean.

- Sample **kurtosis** is a measure of the extent to which observed data fall near the center of a distribution or in the tails, is defined as

$$K = \frac{E(x-\bar{x})^4}{\sigma^4}, \tag{2.14}$$

- the **fifth central moment** is defined as,

$$m_5 = E(x-\bar{x})^5, \tag{2.15}$$

- the **sixth central moment** is defined as,

$$m_6 = E(x-\bar{x})^6 \tag{2.16}$$

  has been calculated,

- finally the statistical **mode** which is the most common pixel value obtained in the moving window w was calculated.

The fifth and higher moments are not so easily related to histogram shape, but they do provide further quantitative discrimination of texture content. Some of these features like mean, variance, skewness, kurtosis, and mode are called *histogram features*, because they are based on the histogram of the window. Some other features, like as the image standard deviation, can be used as a measure of image contrast, since it is a measure of the histogram width, i.e. the spread in pixel values.

2.5.4 Separability analysis

Once the training pixels have been selected and the features that will be used by the classifier have been specified, a **separability** analysis can be performed to determine the combination of features that is best at distinguishing among the given classes.

A measure of separability is typically computed for all possible pairs of classes and for all combinations of q features, out of K total features, where $q = 2, 3, 4, ..., 17$ and $K = 18$. There are $M(M-1)/2$ possible pairs out of M classes which are in our case: (a,b)=$\{$(field1, field2), (field1, field3), (field1, field4), (field2, field3), (field2, field4), (field3, field4)$\}$. Because we will use Mahalanobis distance to classify the image, the average separability over all class pairs is computed also using the Mahalanobis distance:

$$D(a,b) = \left[(\mu_a - \mu_b)^T (\frac{C_a + C_b}{2})^{-1} (\mu_a - \mu_b)\right]^{1/2}, \tag{2.17}$$

where μ_a, μ_b are sample means of features in class a and b, and C_a and C_b are the sample variance-covariance matrices. The subset of features that produces the highest average separability is found.

Because of strong correlations between features, the variance-covariance matrix C may be close to being singular, which causes problems with evaluating the Mahalanobis distance by computer, and so the average separability. We address this problem by evaluating the reciprocal of the condition number of $C = \frac{1}{2}(C_a + C_b)$, and eliminating from consideration those combinations of features that show a condition number below a certain threshold. The reciprocal of the condition number (RCOND) for the matrix C is defined as,

$$y = \frac{1}{\kappa_1(C)} = \frac{1}{\|C\|_1 \cdot \|C^{-1}\|_1}, \tag{2.18}$$

where C is square and invertible and $\kappa_1(C)$ is the condition of C in 1-norm. The *matrix condition number* $\kappa(C)$ of a square matrix C is defined as,

$$\kappa(C) = \|C\| \cdot \|C^{-1}\|, \tag{2.19}$$

where $\|\cdot\|$ is a matrix norm induced by a vector norm.

We believe this to be reasonable since such combinations of features will show high correlations and so should not be good candidates for combinations with good separability. We ran the separability analysis with different threshold values for the reciprocal of the condition number and did obtain different results, as different combinations were allowed. However, we do seem to be converging to a set of best features as our threshold gets closer to 0 and we are only eliminating feature combinations with very high correlations.

We summarize these results in Table 2.2, which shows the best combination of features of different sizes and for different reciprocal of the condition number thresholds. In that table, we have assigned a number to each feature for simplicity (Table 2.1).

Named features	**Label**
correlation vertical	1
correlation horizontal	2
correlation diagonal1	3
correlation diagonal2	4
minimum	5
maximum	6
modulation	7
entropy	8
median	9
mean absolute deviation	10
mean	11
standard deviation	12
variance	13
skewness	14
kurtosis	15
fifth moment	16
sixth moment	17
mode	18

Table 2.1: Features and their labels

The maximum average separability was found **989.8709** and the best feature combination contains the following 10 features: **vertical correlation**, **horizontal correlation**, **diagonal1 correlation**, **diagonal2 correlation**, **maximum**, **modulation**, **mean absolute deviation**, **standard deviation**, **skewness and kurtosis**.

One may then use that subset for classification and save computation time in the classification stage. It is important to match the separability measure to the classifier for the feature subset analysis. For example, one should use Euclidean distance if the nearest-mean algorithm will be used for classification, or transformed divergence, if the maximum-likelihood algorithm will be used.

2.5.5 Classification Algorithm for IGN Images

The separability analysis helped us to reduce the feature space (from 18 features to 10 features) and save computation time also. Having the combination of the 10 features, we can continue to the final procedure that is the image segmentation or image classification. These features give us the most separability between the imformational classes (field1, field2, field3, field4). We are going to use the 5×5 moving window (Figure 2.6) on the satellite image (Figure 2.4).

Starting with the training data, a calculation of the 10 features is performed for each training field. In the second step, we calculate the 10 dimensional mean feature vector μ_i and the 10×10 variance-covariance matrix Σ_i for each field. In the third step, we start to calculate the 10 features for each pixel in the image using the 5×5 moving window. So for each pixel is produced a 10 dimensional *random vector* (r). In the fourth step, we calculate the Mahalanobis distance (Equation 2.20) between the random vector r and the mean vector μ_i for each field.

$$\mathrm{D}_i = (r - \mu_i)'\Sigma^{-1}(r - \mu_i), \quad i = 1, 2, 3, 4 \ . \tag{2.20}$$

The segmentation rule is to allocate each pixel in the image to the nearest category mean vector, as measured by the above distance (minimum distance classification). In Figure 2.7 we can see some results of the segmentation.

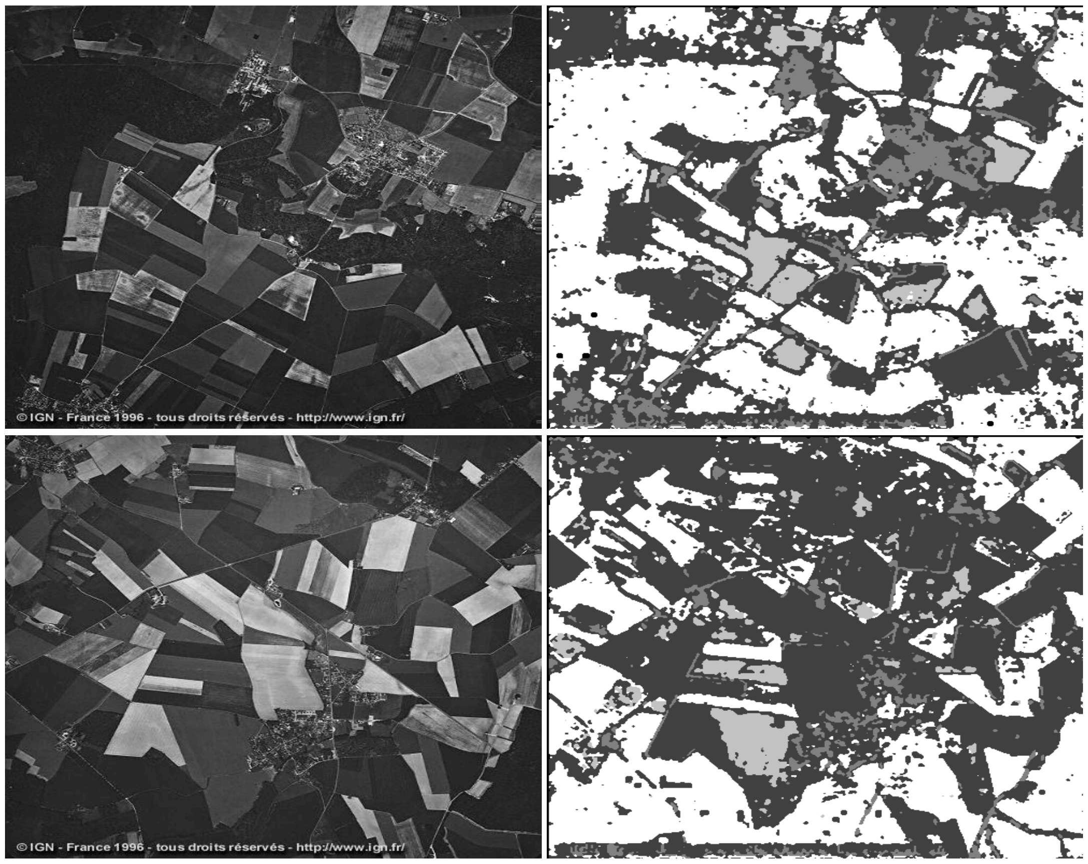

© IGN - France 1996 - tous droits réservés - http://www.ign.fr/
© IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 2.7: Original images (left) were segmented into 4 classes (right).

2.6 Unsupervised Classification for IGN and BAL images

We are going now to present some examples of unsupervised classification methods for the IGN and BAL images. We implement the thresholding method for the IGN and BAL images on a grayscale. Also we present some segmented BAL images as they provided from our project MOUMIR partner, which is the University of Cambridge.

2.6.1 Thresholding for IGN and BAL images

The simplest unsupervised scheme is thresholding, where the set of grey-scale values is partitioned into intervals. Firstly, we define the number of classes we want to segment the image. Several number of classes were used ($c = 4, 6, 12$). Then we calculated the range of the image grayscale pixel values which is defined as,

$$R = max(g_{ij}) - min(g_{ij}), \tag{2.21}$$

where g_{ij} is the grayscale pixel value in lattice in i^{th} row and j^{th} column.

Next, we compute the range of the class. The range is the same for every class and is defined as

$$\varepsilon = \frac{R}{c}. \tag{2.22}$$

The pixel located at lattice position (i, j), with greyscale value g_{ij}, is allocated to category c if

$$(c-1)\varepsilon \leq g_{ij} \leq c\varepsilon. \tag{2.23}$$

In Figure 2.8 we quoted some results for an IGN satellite image. Also some results for a BAL grayscale image are quoted in Figure 2.9. A comparison between the results from supervised classification and the results from thresholding shows that the supervised method works more precisely.

2.7 Post Classification Smoothing

Classified data often manifest a salt-and-pepper appearance due to the inherent variability encountered by a classification when applied on a pixel by pixel basis. It is often desirable to "smooth" the classified output to show only the dominant classification. One means of classification smoothing involves the application of a majority filter. In

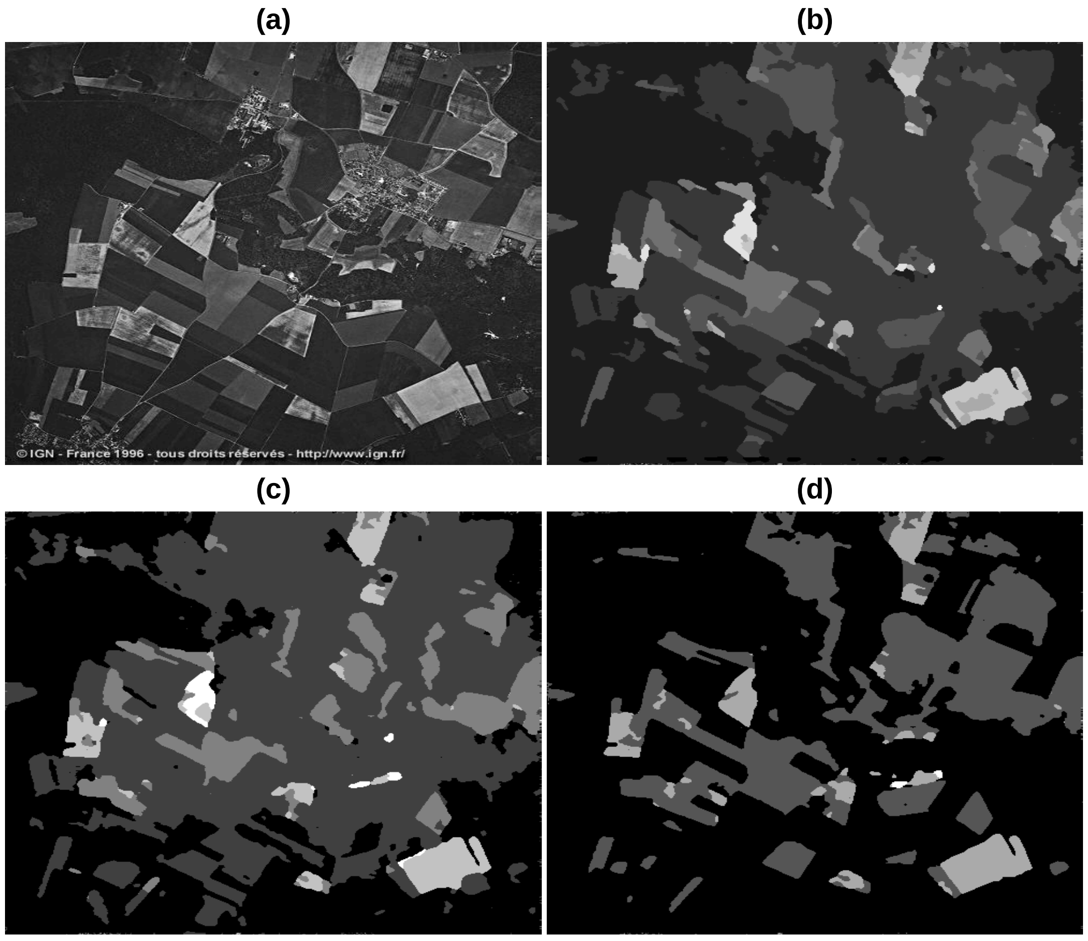

(a)
(b)
(c)
(d)
© IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 2.8: Three segmentations of the IGN satellite image in (a), obtained using manually selected thresholds of (b) 12 classes, (c) 6 classes, (d) 4 classes.

such an operation a moving window is passed through every classified pixel. For example, if pixel (i, j) has been labelled as category c_{ij} by a classification algorithm then a majority filter relabels (i, j) as the most common category in a $(2\mathrm{m} + 1)\times(2\mathrm{m} + 1)$ window centered on pixel (i, j) (Glasbey and Horgan, 1995). If there is no majority category in the window, the identity of the center pixel is not changed. In Figure 2.10 we show an example of the post classification smoothing process.

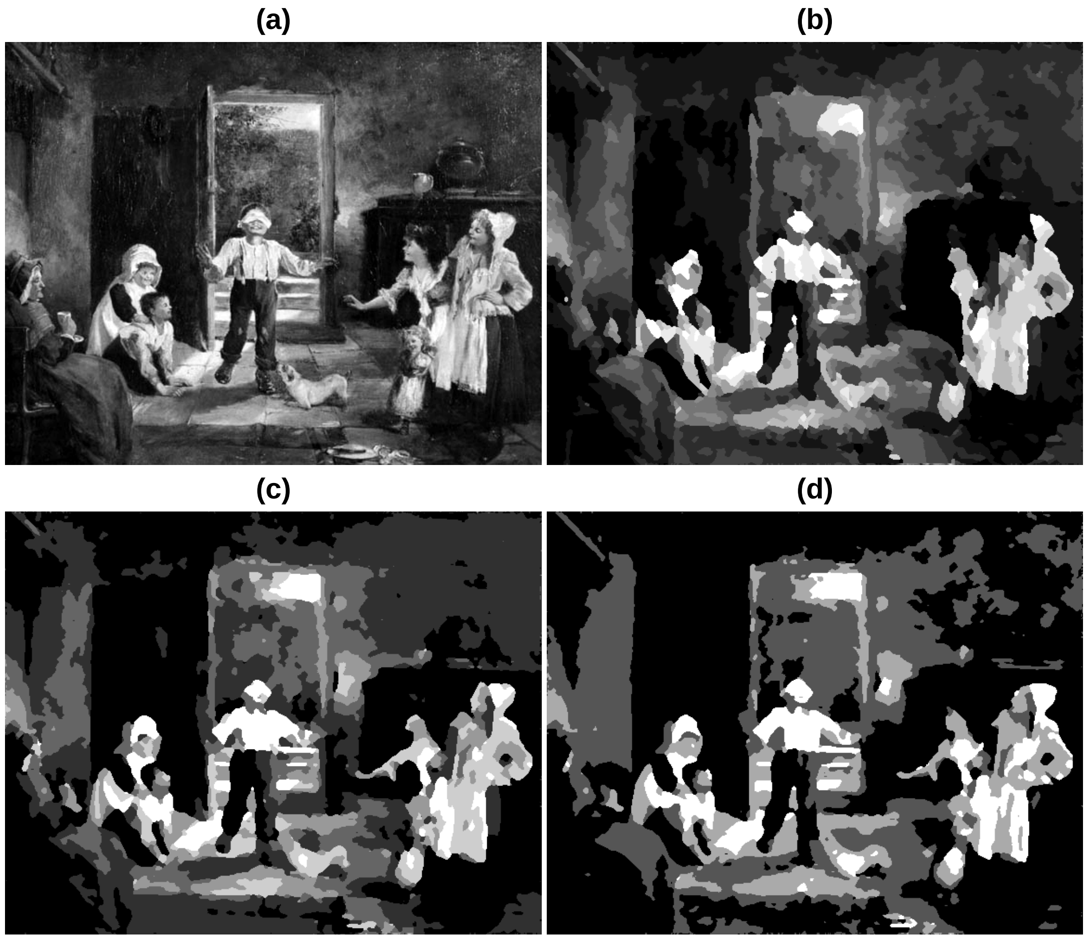


Figure 2.9: Three segmentations of the BAL image in (a), obtained using manually selected thresholds of (b) 12 classes, (c) 6 classes, (d) 4 classes.

2.8 Unsupervised Image Segmentation via Markov Trees and Complex Wavelets

This method was implemented from by MOUMIR project partner. They present an unsupervised segmentation technique in which colour and texture models are learned from the image prior to segmentation, and whose output (including the models) may subsequently be used as a content descriptor in a CBIR system, see Shaffrey et al.

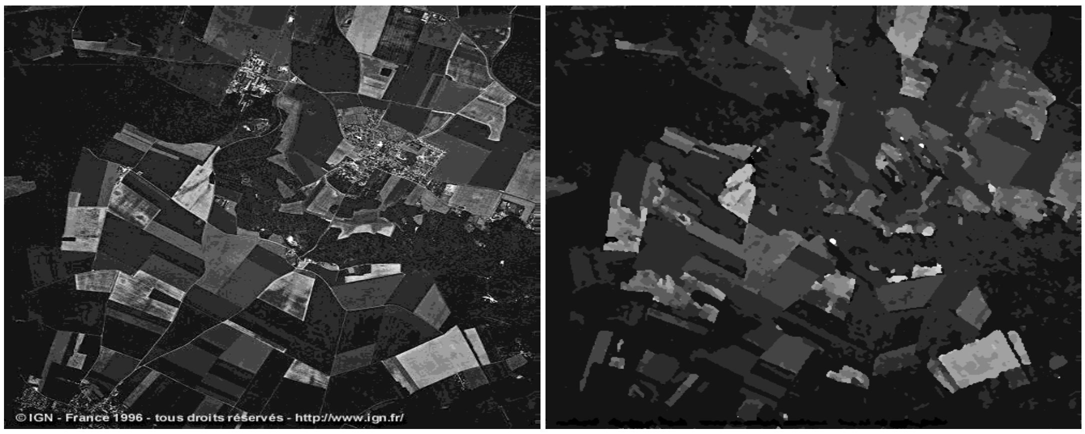

© IGN - France 1996 - tous droits réservés - http://www.ign.fr/

Figure 2.10: Classified image in 12 classes before smoothing (left) and classified image after smoothing by majority filter in a 3×3 window using (right).

(September 2002) for more details. Results are shown in Figure 2.11. In order to facilitate better comparisons with other methods developed within MOUMIR, segmentations obtained by this method are used later in this thesis.

2.9 Discussion

This chapter reviewed a few of the many classification strategies available. There are many choices as the classification process is considered. The supervised classification method was used to classify the IGN images as it is convenient, not very costly to implement and quite fast. Also, the classes should be known and ground truth is also available. We can evaluate statistical properties of the classes. Yet it does seem clear that it is unlikely that a single classification strategy will be "best" for all purposes. Rather, certain approaches may be more effective for certain classes of landscapes, while others may be more effective in different settings. The supervised classification is quite effective for classifying the class of town in the IGN images. The IGN image

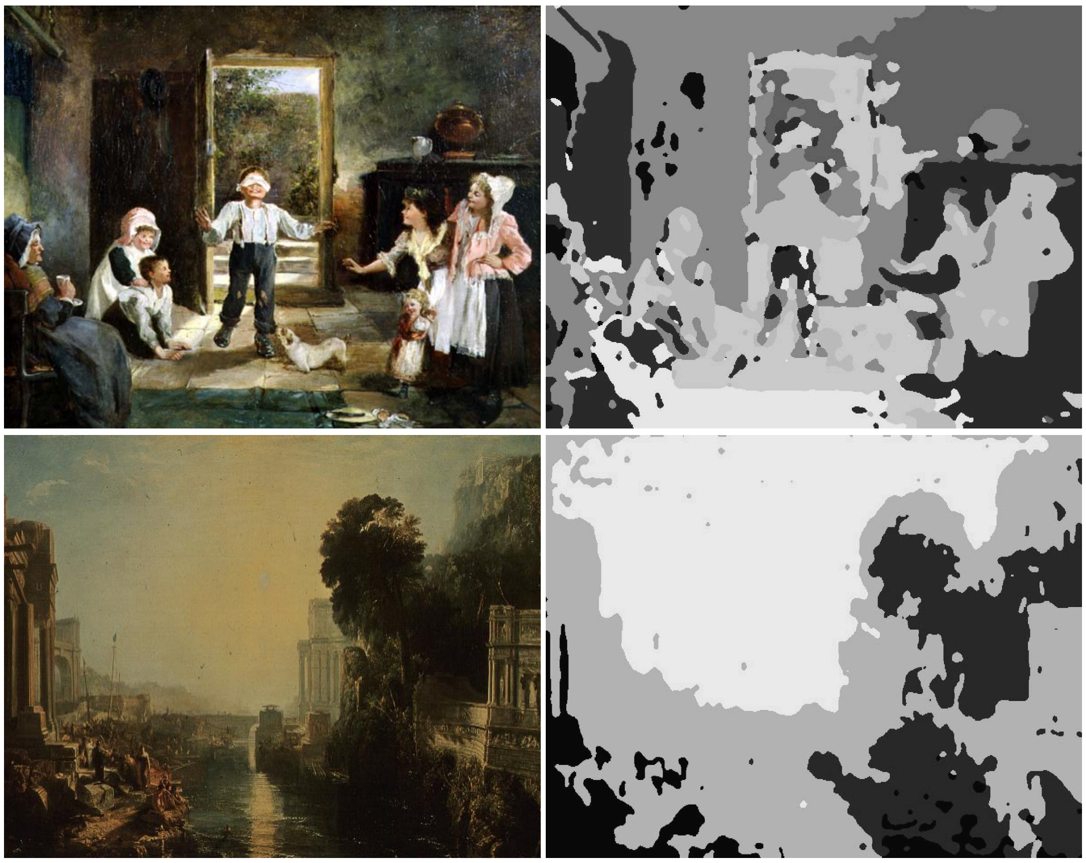


Figure 2.11: Original RGB images (left) and their segmentations (right).

database is a narrow domain, e.g. it has a limited and predictable variability in all relevant aspects of its appearance.

In content-based image retrieval, the image is often divided in parts before features are computed from each part. Partitionings of the image aim at obtaining more selective features by selecting pixels in a trade-off against having more information in features when no subdivision of the image is used at all. Classification is used in many retrieval systems, for example Smeulders et al. (2000). Different segmentation methods may have an effect on the quality of the retrieval process (particularly for object search), but this has not been investigated in this thesis. The classifications results will be used in the next chapter to extract quantitative information from images.

	RCOND = 0.1		RCOND = 0.001		RCOND = 0.0001		RCOND = 0.00001	
Possible combinations of features	Maximum Average Separability	Features selected	Maximum Average Separability	Features selected	Maximum Average Separability	Features selected	Maximum Average Separability	Features selected
2	743.6086	6,11	491.8827	8,11	357.9073	7,18	278.1070	5,7
3	65.0841	1,7,8	757.3230	5,6,8	622.2639	6,8,13	760.6159	5,6,7
4	57.5844	1,2,7,8	903.9120	1,5,6,9	734.1397	6,8,10,13	895.4607	1,5,6,7
5	6.7798	1,2,3,8,14	937.5132	1,2,6,9,12	906.7237	1,6,10, 12,15	928.4608	1,2,6, 7,12
6			940.2394	1,2,6,9, 12,15	943.1467	1,2,6,10, 12,15	943.5981	1,2,6,7, 12,14
7			904.7148	1,2,8,9, 10,14,15	963.6372	1,2,6,10, 12,14,15	955.5745	1,2,4,6, 7,12,14
8			906.8336	1,3,4,5,9, 10,14,15	969.7754	1,2,4,6,10, 12,14,15	971.1164	1,2,6,7,10, 12,14,15
9			3.4198	1,2,3,4, 8,10,12, 14,15	977.7450	1,2,3,4,6, 10,12,14,15	978.1203	1,2,4,6,7, 10,12,14,15
10					978.7308	1,2,3,4, 6,8,10, 12,14,15	**989.8709**	**1,2,3,4, 6,7,10, 12,14,15**
11					969.0229	1,2,3,6, 8,9,10, 12,14,15,18	846.3063	1,2,4,5, 6,10,12, 13,14, 15,18
12					931.7973	1,3,4,5, 6,9,10, 11,13, 14,15,18	852.3957	1,2,3,4, 5,6,10, 12,13,14, 15,18
13							854.1850	1,2,3,4, 5,6,9, 10,11, 12,13, 15,18
14							857.2247	1,2,3,4, 5,6,9, 10,11, 12,13, 14,15,18

Table 2.2: Separability analysis for different number of features and different threshold values for RCOND. Blank cells indicate that no combination of features produced a variance matrix that satisfied the RCOND threshold.

Chapter 3

Feature Extraction for CBIR

3.1 Introduction

In a large number of image processing applications, including content based image retrieval, *features* must be extracted from image data, from which a description, interpretation, or understanding of the scene can be provided by the machine; see Jain (1989). Representation of perceptual features of images like colour, texture, shape, image structure and spatial relationships is a fundamental problem in visual information retrieval. Image analysis and pattern recognition algorithms provide the means to extract values which give a quantitative measure of these features. Certain basic semantic properties, or semantic primitives of an image, such as objects, actions, events or narrative structures, can be also derived by analysing combinations of low-level features according to suitable models. Visual primitives also convey meaning to the observer. Colours, for example, cause particular sensations according to their chromatic properties and spatial arrangement. Bimbo (1999) discusses this in more detail.

Content based image retrieval retrieves stored images from a collection by comparing features automatically extracted from the images themselves. The commonest features used are mathematical measures of colour, texture or classification (spatial relationships of objects in the image). All current CBIR systems, whether commercial or experimental, operate according to the above principle. A typical system allows users to formulate queries by submitting an example of the type of image being sought. The system then identifies those stored images whose feature values match those of the query most closely, and displays thumbnails of these images on the screen. In this chapter, we introduce the representation of perceptual pictorial features that we have used in order to derive the feature vector **F** for every image i in the database.

3.2 Features Based on Classification

In Chapter 2 some versions of image classification were presented. Classification features play an important role in content based image retrieval. The result of classification is a set of homogeneous regions in the image, defined by some criterion. In terms of the semantics of an image, we hope that the different regions correspond to different objects or activities in the image, bearing in mind that automatic classification techniques usually result in a fuzzy, blobby description of objects, rather than a precise classification. However, features from those segments (classes) often capture useful information of the object. At least, they provide some information about the image scene.

Features which are based on classification have often been motivated by content-based retrieval (Smeulders et al., 2000). In the following, we introduce the features we calculated from the classified images.

3.2.1 Number of Classes

Once we have the classified image, or labeled image, it is very easy to count the number of classes or categories or segments that are being classified. The labeled image is a matrix L of positive integers $1, 2, ...$ where each number represents each class. So,

$$\text{Number of classes} = \max(L). \tag{3.1}$$

3.2.2 Area

The *area* of a class c is the number of pixels in the class:

$$area_c = \sum_i \sum_j I(L_{ij} = c). \tag{3.2}$$

If we divide the size of area of each class by the total number of pixels in the image, we get relative area, which has the advantage of being invariant to image size. For example, in Figure 3.1 the two images are different sizes but the semantics are the same. In this work, we use the following features of relative area, that convey

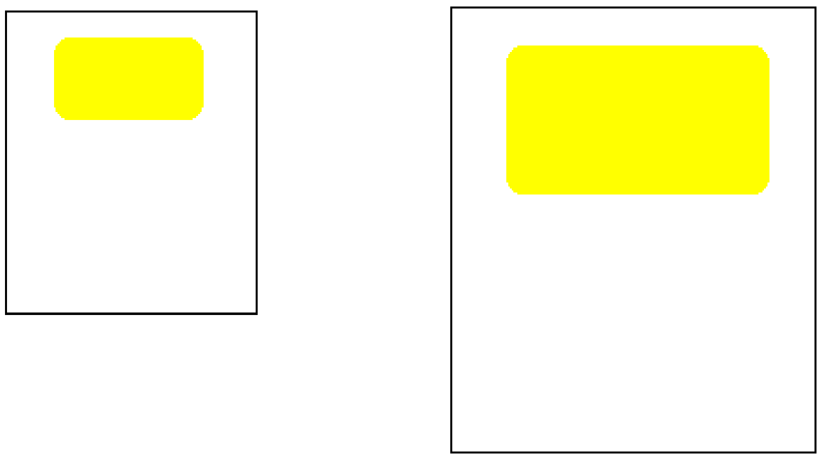


Figure 3.1: Two images with the same semantics but different sizes.

information on the average size and variability in size of the image classes:

- *Minimum relative area*
- *Maximum relative area*
- *Median relative area*
- *Standard deviation of relative areas.*

3.2.3 Centroid

The centroid of a class is the x and y coordinates of the centre of mass of the class. This is a measure of the class' location in the image. The x and y coordinates are expressible in terms of zeroth and first order moments by

$$centroid = \left(\frac{\mu_{10}}{\mu_{00}}, \frac{\mu_{01}}{\mu_{00}} \right), \tag{3.3}$$

where the definition of the $(k,\ l)$th-order moment is:

$$\mu_{kl} = \sum_i \sum_j I(L_{ij} = c) i^k j^l \quad \text{for} \quad k, l = 0, 1, 2, ... \tag{3.4}$$

We calculated the following features based on centroids, which provide information on the positions of objects in the image.

- *Standard deviation of x-coordinates*
- *Standard deviation of y-coordinates*
- *Distance of closest centroid to image centre;* We should point out that the image was being rescaled in the terms of its rows and its columns between 0 and 1, so that the feature is invariant to image size. In Figure 3.2 there are two images with different sizes but with the same semantics. After rescaling, the distance between the class and the image centre (black dot) is the same for both images.

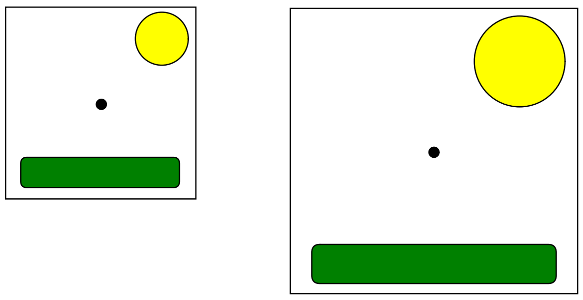


Figure 3.2: Two images with the same spatial properties but different sizes.

- *Distance of furthest centroid to image centre*
- *Shortest distance between centroids*
- *Largest distance between centroids*
- *Centroid of class of maximum area*
- *Centroid of class of minimum area*
- *Number of centroids in each of 9 equal blocks; Distinct blocks* are rectangular partitions that divide a matrix into m-by-n sections. Distinct blocks overlay the image matrix starting in the upper-left corner, with no overlap. Then an $m \times n$ matrix Q is defined, where Q_{ij} is the number of centroids in that block of the image. Figure 3.3 shows a matrix divided into 3-by-3 blocks. The $m \times n$ matrix elements are then used as features.

3.2.4 Eccentricity

A statistic often used to describe shape is the eccentricity of the ellipse. The eccentricity is the ratio of the maximum length of line or chord that spans the class to the minimum length chord. The value is between 0 and 1. An ellipse whose eccentricity is 0 is actually

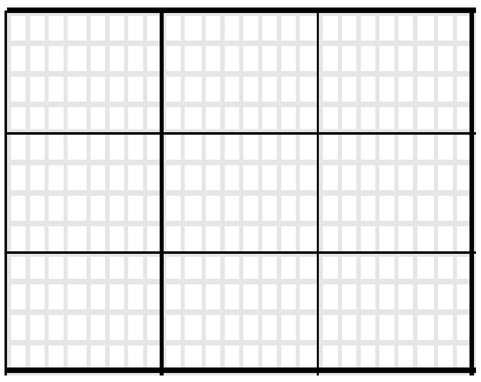


Figure 3.3: An image divided into 9 distinct blocks.

a circle, while an ellipse whose eccentricity is 1 is a line segment. We can also define eccentricity using moments as

$$\text{eccentricity} = \frac{\mu_{20} + \mu_{02} + \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}}{\mu_{20} + \mu_{02} - \sqrt{(\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2}} \tag{3.5}$$

The following features which are based on eccentricity were calculated:

- *Eccentricity of class of minimum area*
- *Eccentricity of class of maximum area*

3.2.5 Orientation

The angle (in degrees) between the x-axis and the major axis of the ellipse. Figure 3.4 shows the axes and orientation of the ellipse. The left side of the figure shows an image region and its corresponding ellipse. The right side shows the same ellipse, with features indicated graphically. The solid blue lines are the axes, the red dots are the foci, and the orientation is the angle between the horizontal dotted line and the major axis. The following features were calculated:

- *Orientation of class of minimum area*

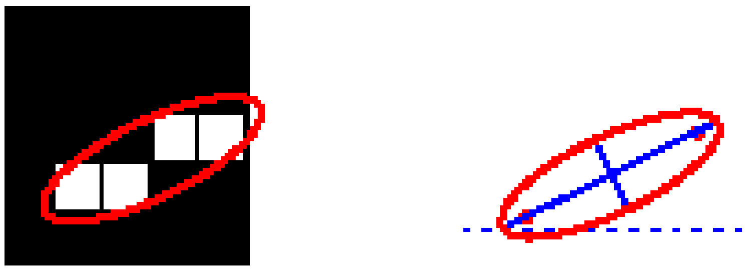


Figure 3.4: Illustration of the axes and orientation of the ellipse.

- *Orientation of class of maximum area*

3.2.6 Local Colour Statistics

Based on the original RGB (Red, Green, Blue) image, we calculate for the maximum-area class the following statistical measures:

- *Mean of red channel*
- *Mean of green channel*
- *Mean of blue channel*
- *Standard deviation of red channel*
- *Standard deviation of green channel*
- *Standard deviation of blue channel*
- *Skewness of red channel*
- *Skewness of green channel*
- *Skewness of blue channel*

3.2.7 Edge length

An edge is a transition between adjacent classes. Given the labeled matrix L we can scan this matrix horizontally and vertically and so we can calculate the length of the edges. The edges can give us information about the image domain. In Figure 3.5 we can see that the small edge length indicates a narrow domain, when a big edge length indicates a broad domain. A narrow domain has a limited and predictable variability in all relevant aspects of its appearance. A broad domain has large and unpredictable variability in its appearance even for the same semantic meaning.

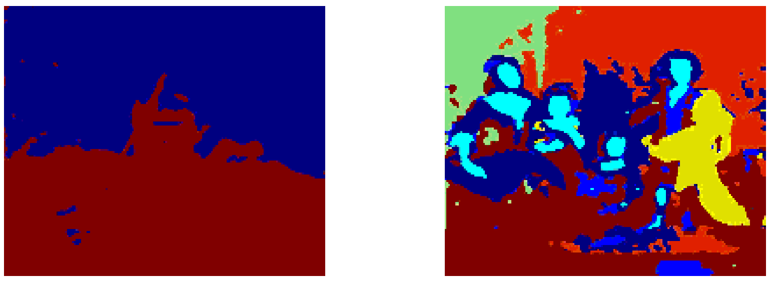


Figure 3.5: An image with small edge length (left) and with big edge length (right).

- *Horizontal edge length;* Let HE be a matrix of the size of L. Then define

$$HE(i,j) = \begin{cases} 1, & \text{if } L(i,j) \neq L(i,j+1), \\ 0, & \text{otherwise.} \end{cases} \tag{3.6}$$

  The length of horizontal edges in the image is then $\sum_i \sum_j HE(i,j)$.

- *Vertical edge length;* Similarly, let

$$VE(i,j) = \begin{cases} 1, & \text{if } L(i,j) \neq L(i+1,j), \\ 0, & \text{otherwise.} \end{cases} \tag{3.7}$$

  then $\sum_i \sum_j VE(i,j)$ is the length of vertical edges.

- *Total edge length;* Also once we have the horizontal edge length and the vertical edge length we can calculate the total edge length as the sum of them.

3.3 Features based on Global Colour

Colour is a visual feature which is immediately perceived when looking at an image. Retrieval by colour similarity requires that models of colour stimuli are used, such that distances in the colour space correspond to human perceptual distances between colours. Moreover, colour patterns must be represented in such a way that salient chromatic properties are captured. It is well known that the standard representation of a colour, as a combination of intensities of red, green and blue, is not the best representation for measuring differences in colour as perceived by humans.

3.3.1 Colour Spaces

Colour is represented as a point in a three-dimensional colour space. Several geometric colour models are available, which support a precise and quantitative representation of colour stimuli.

Colour is the brain's reaction to a specific visual stimulus. Although we can precisely describe colour by measuring its spectral power distribution (the intensity of the visible electro-magnetic radiation at many discrete wavelengths) this leads to a large degree of redundancy. The reason for this redundancy is that the eyes' retinas sample colour using only three bands, roughly corresponding to red, green and blue light. The signals from the colour sensitive cells in the retina (cones), together with those from the rods (sensitive to intensity only), are combined in the brain to give several different "sensations" of the colour. These sensations have been defined by the CIE (Commission Internationale de l' Eclairage) and are quoted from Hunt (1998):

- Brightness: the human sensation by which an area exhibits more or less light.
- Hue: the human sensation according to which an area appears to be similar to

one, or to proportions of two, of the perceived colours red, yellow, green and blue.

- Colourfulness: the human sensation according to which an area appears to exhibit more or less of its hue.
- Lightness: the sensation of an area's brightness relative to a reference white in the scene.
- Chroma: the colourfulness of an area relative to the brightness of a reference white.
- Saturation: the colourfulness of an area relative to its brightness.

The tri-chromatic theory of light states three separate light, red, green and blue, can match any visible colour based on the eye's use of three colour sensitive sensors. This is the basis on which photography and printing operate, using three different coloured dyes to reproduce colour in a scene. It is also the way that most computer colour spaces operate, using three parameters to define a colour. The spaces are usually denoted by 3 letters, each letter representing one of the 3 variables in the colour space, for example RGB for the standard representation of red, green and blue. The spaces can be divided into two groups, hardware and user-oriented. Because, the complete presentation of all these colour spaces is out of the scope of this thesis, we refer to the references Castleman (1996), Galbiati (1990), Bimbo (1999), Hunt (1998), Jain (1989), Lewis (1990), Petrou and Bosdogianni (1999), which form a rich source.

Colour models for representing colour stimuli can be distinguished as Bimbo (1999):

- **Hardware-oriented** models. These are defined according to properties of optical devices used to reproduce colours, such as the TV monitor, the computer screen and the colour printer. Hardware-oriented colour models are the **RGB** (red, green, blue), **CMY** (cyan, magenta, yellow), **YIQ** (Y component represents the intensity, the I and Q component represents the color information). It is difficult for the user to deal with these models, in that they do not directly refer to intuitive notions of hue, saturation and brightness.

- **User-oriented** models. These are based on human perception of colours. User-oriented colour models are the **HSL** (Hue, Saturation, Lightness), **HSV** (Hue, Saturation, Value), **L*u*v*** (L* component defines the luminancy, and u*, v* define chrominancy), **L*a*b*** (the model represent the luminance of the color L, a is the colour's position between red and green and b is the colour's position between yellow and blue) and **L*C*h*** (Luminosity, Chroma, Hue).

We are going to focus only on **RGB** and **HSV** because we worked with them. Different colour spaces are better for different applications, for example some equipment has limiting factors that dictate the size and type of colour space that can be used. Some colour spaces are perceptually linear, i.e. a 10 unit change in stimulus will produce the same change in perception wherever it is applied. Many colour spaces, particularly in computer graphics, are not linear in this way. Some colour spaces are intuitive to use, i.e. it is easy for the user to navigate within them and creating desired colours is relatively easy. Other spaces are confusing for the user with parameters with abstract relationships to the perceived colour. Finally, some colour spaces are tied to a specific piece of equipment (i.e. are device dependent) while others are equally valid on whatever device they are used.

3.3.2 RGB Colour Space

RGB is the most commonly used hardware-oriented scheme for digital images. It maintains compatibility with devices that originate or display images and is somewhat based on the physiology of human retina. Colours in RGB are obtained as the addition of the three primaries Red, Green and Blue. The RGB colour space is a solid having the shape of a unit cube, as shown in Figure 3.6. In this cube, the grey scale goes from

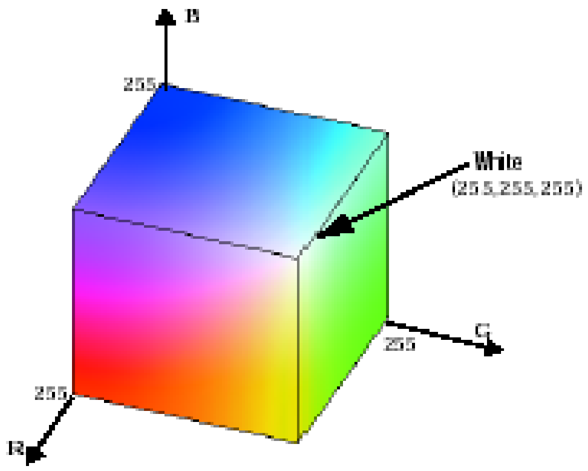


Figure 3.6: RGB colour space.

(0, 0, 0) (Black) to (255, 255, 255) (White). Maximally saturated colours are placed at the corners of the cube, Red at (255, 0, 0), Green at (0, 255, 0), Blue at (0, 0, 255), Cyan at (0, 255, 255), Magenta at (255, 0, 255) and Yellow at (255, 255, 0), [Bimbo (1999)].

3.3.3 HSV Colour Space

The **HSV** (Hue, Saturation, Value) model defines a colour space in terms of three constituent components:

- **Hue** is the colour type such as red, blue, or yellow. It ranges from 0-360. Hue is measured by an angle with Red starting at 0 degrees, Green at 120 degrees and Blue at 240 degrees. Complementary colours are in-between: Yellow is at 60 degrees, Cyan is at 180 degrees, and Magenta is at 300 degrees.

- **Saturation** is the "vibrancy" of the colour and ranges from 0-100%. The lower the saturation of a colour, the more "grayness" is present and the more faded the colour will appear. In other words the saturation is the lack of whiteness.

- **Value** is the brightness of the colour, the quantity of light. It ranges from 0%(dark) to 100%(bright).

In Figure 3.7 we can see a diagram of HSV colour space. It can be derived from the

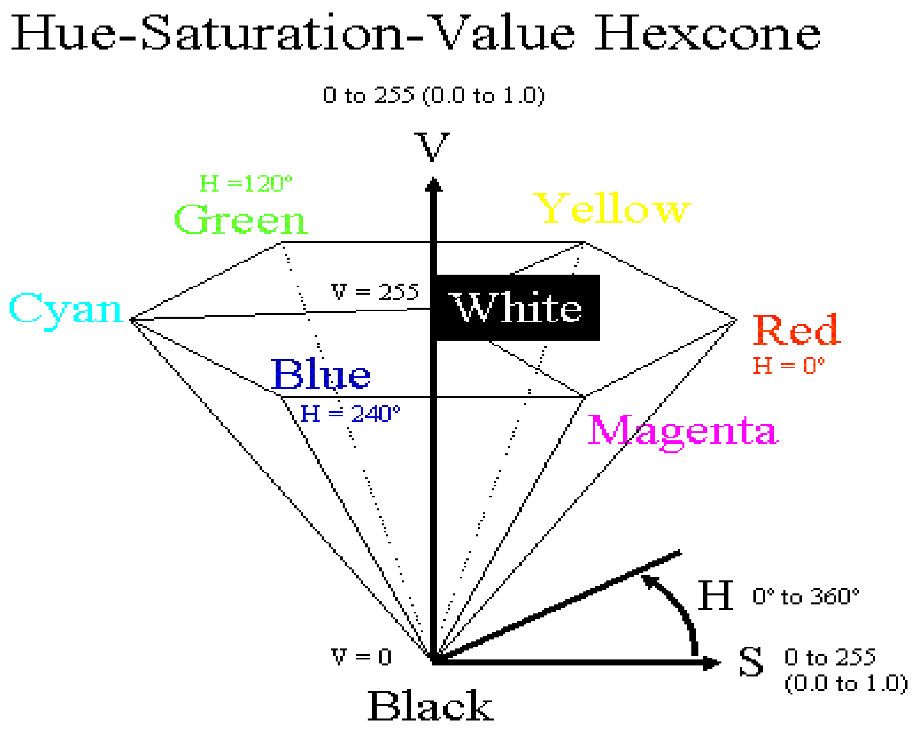


Figure 3.7: A diagram of HSV colour space.

RGB space according to the following transformation:

$$V = \frac{R+G+B}{3} \tag{3.8}$$

$$S = \left[1 - \frac{3}{R+G+B}\min(R,G,B)\right] * 100 \tag{3.9}$$

$$H = 180\frac{0.5(R-G)+(R-B)}{((R-G)^2+(R-B)(G-B))^{1/2}}, \quad if\ \frac{B}{V} \leq \frac{G}{V} \tag{3.10}$$

$$H = undefined, \quad if\ S = 0 \tag{3.11}$$

$$H = 360 - H, \quad if\ \frac{B}{V} > \frac{G}{V} \tag{3.12}$$

These allow any RGB image to be converted to an HSV image. The HSV colour space is used because it is much more efficient for colour recognition. The HSV colour space is a popular choice for manipulating colour. It was developed to provide an intuitive representation of colour and to approximate the way in which humans perceive and manipulate colour, see Manjunath et al. (2001).

There are two main occasions when RGB is inconvenient. The first is when defining a type of colour. The hue component of HSV explicitly defines colour type e.g. green, pink, purple, whereas the definition of a colour type in terms of RGB values is complicated. The second is colour matching or determining if one colour is similar to another colour. The RGB colour space is conceptually a cube with one axis representing red, one representing green, and one representing blue, as shown in Figure 3.6. Where the axes meet at (0,0,0), we have black, and at (255,255,255) we have white. A measure of colour similarity is to take the Euclidean distance between the two colours. Colours are similar if this distance is less than a threshold. In RGB space, all points less than or equal to the threshold form a sphere inside the RGB cube. The user probably thinks of matching a colour by choosing "all the bluish tones", or some similar perceptual way. But the sphere that one obtains in the RGB cube does not include many of the values that would meet this criteria, and it does include many that probably would not. There is no obvious transform to get more of those colours which do match. Another example is purple colours that run along the diagonal between the red and blue axes.

3.3.4 HSV Histogram

We compute a 64-element-long histogram of the HSV values of the image's pixels, by converting the RGB colour image to HSV color space image, and then quantizing it into $4 \times 4 \times 4 = 64$ colour bins.

Colour quantization

Colour quantization is the term used to describe the process of reducing the number of colours in an image by selecting an optimized set of representative colours and then re-applying this reduced set to the original image. In uniform colour quantization each channel of the colour space is treated independently. Each channel is then divided into equal sized segments. In this work we divided the Hue, Saturation and Value components into 4 segments of equal size. Once the colour space is divided, each of the original colours is then mapped to the region in which it falls. Figure 3.8) illustrates this for the Hue component. The same process is followed for Saturation and Value. We then count the number of pixels for the 64 possible colour combinations to get the histogram. The representative colours for each region is then the average of all the

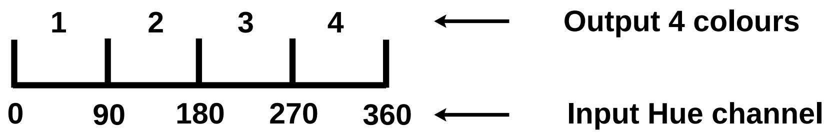


Figure 3.8: A uniform quantizer for the Hue channel.

colours mapped to that region.

In non-uniform quantization, the colour space is divided according to the distribution of colours in the image. Image quantization is an important problem in computer graphics, that arises when displaying high-colour images on non-truecolour devices, see [Buhmann et al. (1998)]. We have used the uniform colour quantization algorithm because it is quicker and easier to implement and usually gives the same amount of information. Non-uniform quantization may be better if many of the images being analysed have a small range of colours, which is not the case for the database we use in this thesis, see for example [Jain (1989)], [Smith and Chang (1996a)].

In Figure 3.9 we demonstrate an RGB colour image and its conversion to quantized HSV image. The quantized HSV image has dominant colour regions emphasized. The 64-element-long HSV histogram, for the same image, is shown in Figure 3.10 and can be represented as a vector $< h_1, h_2, ..., h_{64} >$, in which each bucket (bin) h_j contains the percentage of pixels of HSV colour combination j in the image.

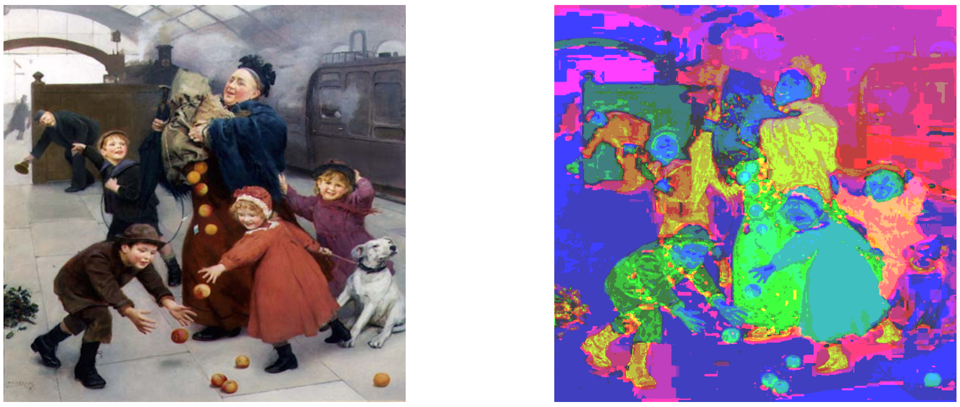


Figure 3.9: RGB original image (left) and the quantized HSV colour image (right).

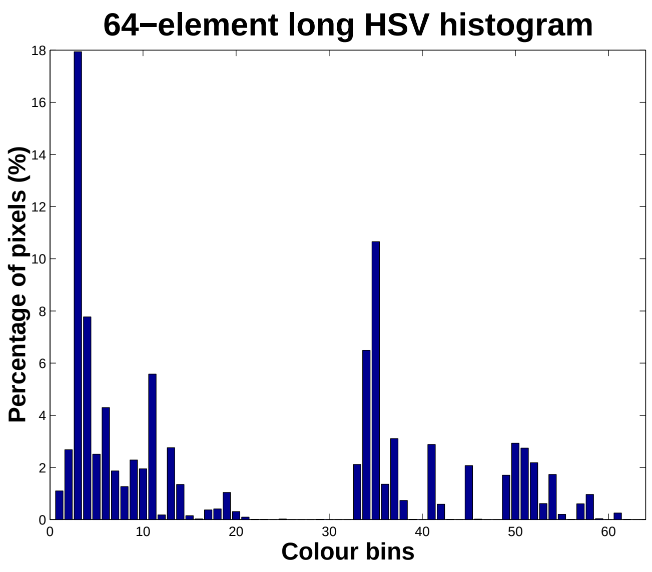


Figure 3.10: A 64-element-long histogram. The x-axis represents all the combinations of colour and the y-axis represents the percentage of pixels which have a particular colour combination. For instance the 1 represents the combination (H,S,V)=(1,1,1) and the 10 represents the combination (H,S,V)=(1,3,2).

3.3.5 Median Intensity

An intensity image is a data matrix I whose values represents intensities within some range (commonly 0-255). Figure 3.11 we can see an intensity or grayscale image and its corresponding histogram with the median value denoted with the solid line.

3.3.6 Mean Saturation

As we described earlier, saturation refers to the purity of colour. As saturation increases, colours appear more “pure”. As saturation decreases, colours appear more “washed-out”. Once we convert the RGB colour image to HSV colour space, we can isolate the Saturation channel and calculate its mean value.

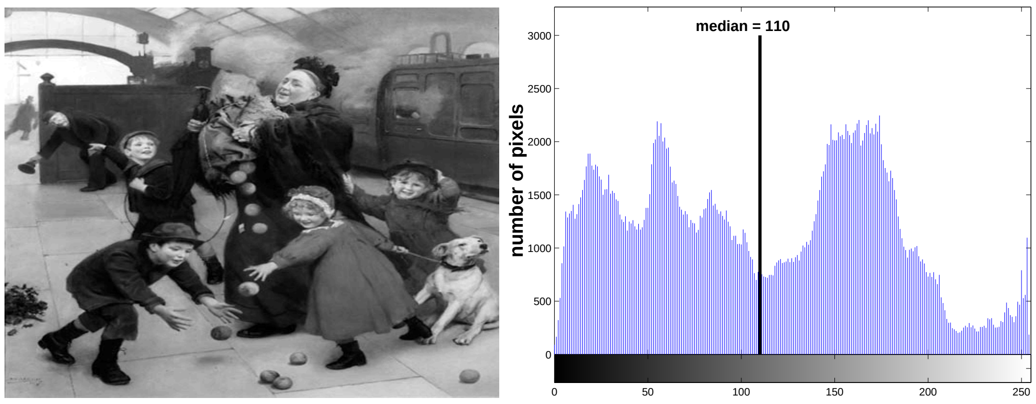


Figure 3.11: An intensity image and its histogram.

3.3.7 Percentages of Colours

Another feature that we use is the percentages of pixels that fall into the ranges of HSV colourspace defined in Table 3.1.

Colours	**Hue(^{o})**	**Saturation(%)**	**Value(%)**
black	0...360	0...100	0...3
grey	0...360	0...15	2...85
white	0...360	0...15	80...100
red	-70...25	10...100	5...100
orange	15...50	10...100	2...100
yellow	25...80	10...100	8...100
green	75...185	10...100	2...100
blue	175...260	2...100	2...100
purple	255...300	10...100	2...100
brown	-50...80	5...85	1...40
pink	-70...25	10...60	2...100

Table 3.1: Percentages of pixels that fall into the following ranges of HSV colourspace.

3.4 Features based on Texture

Texture has been one of the most important characteristics which have been used to classify and recognize objects and scenes (Aksoy and Haralick, June 1998). Texture is a broad term used in pattern recognition to identify image patches that are characterized by differences in brightness. Texture is actually a very nebulous concept, often attributed to human perception, as either the feel or the appearance of (woven) fabric. Everyone has their own interpretation as to the nature of texture. According to Nixon and Aguado (2002) there is no mathematical definition for texture, it simply exists. From a psychological point of view, texture features that strike a human observer are granularity, directionality and repetitiveness. In the context of image databases, statistical methods can generate features that often can be arranged in numerical vectors, giving access to a plethora of techniques for defining and manipulating texture similarity; (Santini, 2001).

Together with colour, texture is a powerful discriminating feature, present almost everywhere in nature. Feature spaces based on texture are particularly interesting for image retrieval. Like colours, textures are connected with psychological effects. In this section, we address texture representation and algorithms for texture feature extraction; (Bimbo, 1999).

3.4.1 RGB Colour Coherence Vector

The colour-coherence vector is a global measure of texture in an image. We quantize the RGB image into $4 \times 4 \times 4 = 64$ colour bins. This procedure is exactly the same as the one we described earlier in Subsection 3.3.4, the only difference being that we have to quantize the Red, Green and Blue channels. In Figure 3.12, we demonstrate this with an example of two RGB quantized images. It is obvious that the quality of these quantized images has been decreased. Pass et al. (1996) define a colour's *coherence* as the degree to which pixels of that colour are members of large similarly-coloured regions. They refer to these significant regions as *coherent regions*, and observe that they are of significant importance in characterizing images. For example, the images in Figure 3.12 have similar colour histograms, despite their rather different appearances. The colour red appears in both images in approximately the same quantities. In the image with the flowers the red pixels are widely scattered, while in the portrait image the red pixels (from the man's trouser) form a quite big single coherent region. So, if we isolate these areas we can see different textures. That is why we assert that the colour coherence vector is a descriptor of texture.

The coherence measure classifies pixels as either coherent or incoherent. Coherent pixels are a part of some sizable contiguous region, while incoherent pixels are not. A *colour coherence vector* represents this classification for each color in the image. This notion of coherence allows us to make fine distinctions that cannot be made with simple colour histograms.

The next step is to classify the pixels within a given quantized colour as either coherent or incoherent. A coherent pixel is part of a large group of pixels of the same colour, while an incoherent pixel is not. We can determine the pixel groups by computing connected components. A connected component C is a maximal set of pixels of the same quantized colour such that for any two pixels $p, p' \in C$, there is a path in C between p and p'. Formally, a path in C is a sequence of pixels $p = p_1, p_2, ..., p_n = p'$ such that each pixel p_i is in C and any two sequential pixels p_i, p_{i+1} are adjacent to each other, where we consider two pixels to be adjacent if one pixel is among the eight closest neighbors of the other. We only compute connected components within a given discretized colour bucket. Connected components can be computed in a single pass over the image (see for a description of how to do this in Glasbey and Horgan (1995)). When this is complete, each pixel will belong to exactly one connected component. The pixels are then classified as either coherent or incoherent depending on the number of pixels in the connected component to which it belongs. A pixel is coherent if the size of its connected component exceeds a fixed value τ, otherwise the pixel is incoherent. For a given discretized colour, some of the pixels with that colour will be coherent and some will be incoherent. Let us call the number of coherent pixels of the j^{th} discretized colour α_j and the number of incoherent pixels β_j. Clearly, the total number of pixels with that colour is $\alpha_j + \beta_j$, and so a colour histogram would summarize an image as

$$\langle \alpha_1 + \beta_1, ..., \alpha_n + \beta_n \rangle.$$

Instead, for each colour we compute the pair

$$(\alpha_j, \beta_j)$$

which is called the *coherence pair* for the j^{th} colour. For a quantization into $4 \times 4 \times 4 = 64$ colours, the 128-element long colour-coherence vector for the quantized image consists of

$$\langle (\alpha_1, ..., \alpha_{64}), (\beta_1, ..., \beta_{64}) \rangle^{\mathsf{T}}.$$

This vector is the concatenation of two 64-bin histograms: one for coherent pixels and one for incoherent pixels. In our calculations we have used $\tau = 150$ pixels as the threshold for defining a pixel to be coherent/non-coherent.

3.4.2 HSV Colour Autocorrelogram

This is another feature for image indexing and comparison. This feature distills the spatial correlation of colours, and is both effective and inexpensive for content-based image retrieval according to Huang et al. (1997). A colour correlogram is a table indexed by colour pairs, where the k^{th} entry for (i, j) specifies the probability of finding a pixel of colour j at a distance k from a pixel i in the image. Due to the high complexity and large computational cost, the autocorrelogram is used instead, which captures spatial correlation between identical colours only. A 256-element long HSV colour autocorrelogram at distances 1, 3, 5 and 7 pixels is calculated in this work, because we want to capture some reasonable long range textures and do not compromise the quality of the feature, see Figure 3.13. The pixel values are subjected to the same preprocessing (conversion from RGB colour space to HSV colour space and quantization into $4 \times 4 \times 4$=64 colour bins) as the HSV-histogram in Subsection 3.3.4. The 256 (64+64+64+64)-element long HSV colour autocorrelogram consists of

$$\langle (a_1^1, ..., a_{64}^1), (a_1^3, ..., a_{64}^3), (a_1^5, ..., a_{64}^5), (a_1^7, ..., a_{64}^7) \rangle.$$

The first 64 bins are the number of times each pixel of a given colour had neighbours of the same colour at distance 1. The next 64 bins are for distance 3, and similarly for distance 5 and 7 pixels.

3.4.3 Image Contrast

The last feature we have calculated for every image in the database is the image contrast. Contrast is the basis for image perception. It allows the human observer to differentiate between regions. The image contrast has been calculated over a grayscale image as follows. First, we define Y0 to be the brightness value that 1/3rd of the grayscale pixel values are below, and second the Y1 is a brightness value that 2/3rds of the pixels are below. Then, the image contrast can be calculated as the difference Y1-Y0, see (Cox et al., 1996). In Figure 3.14 we can see an example of an image with a high and low contrast.

3.5 Feature Normalisation

The goal is to make all features have approximately the same effect in the computation of similarity by independently normalizing each feature component to the [0,1] range. We can do this by linear scaling to unit range. Given a lower bound l and an upper bound u for a feature component x,

$$f = \frac{x - l}{u - l} \tag{3.13}$$

results in f being in the [0,1] range. A summary of the entire set of global image feature descriptors is provided in Table 3.2. Other normalisations are possible. One could normalise by sub-vectors by each feature or set of features for example or use normalisations other than linear, such as linear scaling to unit variance, whitening transform, transformation to a uniform[0,1] random variable, rank normalisation and normalisation after fitting distributions, see (Aksoy and Haralick, 2000).

3.5.1 Energy Normalisation

It is evident from the entries in Table 3.2 that the number of texture features is considerably larger than either colour or spatial. Later, we normalise features so that colour, texture and segmentation features should carry equal weight. Features in a particular grouping (e.g. colour features) are normalized by dividing by the square root of the number of elements in that grouping. This makes sense if we view user feedback as being made on the basis of a combination of similarities in these 3 feature sets. Component normalisation gives more weight to texture, that contains more features, whereas normalisation by set gives them an equal weight when computing distance.

Features	Description	Elements
Segmentation	• Number of classes, $\langle f_1 \rangle$	1
	• Minimum relative area, $\langle f_2 \rangle$	1
	• Maximum relative area, $\langle f_3 \rangle$	1
	• Median relative area, $\langle f_4 \rangle$	1
	• Standard deviation of relative areas, $\langle f_5 \rangle$	1
	• Standard deviation of x-coordinates, $\langle f_6 \rangle$	1
	• Standard deviation of y-coordinates, $\langle f_7 \rangle$	1
	• Distance of closest class centroid to image centre, $\langle f_8 \rangle$	1
	• Distance of furthest class centroid to image centre, $\langle f_9 \rangle$	1
	• Shortest distance between centroids, $\langle f_{10} \rangle$	1
	• Largest distance between centroids, $\langle f_{11} \rangle$	1
	• Centroid of class of maximum area, $\langle f_{12}, f_{13} \rangle$	2
	• Centroid of class of minimum area, $\langle f_{14}, f_{15} \rangle$	2
	• Number of centroids in each of 9 equal blocks, $\langle f_{16}, f_{17}, ..., f_{24}, \rangle$	9
	• Eccentricity of minimum area, $\langle f_{25} \rangle$	1
	• Eccentricity of maximum area, $\langle f_{26} \rangle$	1
	• Orientation of minimum area, $\langle f_{27} \rangle$	1
	• Orientation of maximum area, $\langle f_{28} \rangle$	1
	• Mean of red channel pixels belong to the maximum area class, $\langle f_{29} \rangle$	1
	• Mean of green colour pixels belong to the maximum area class, $\langle f_{30} \rangle$	1
	• Mean of blue colour pixels belong to the maximum area class, $\langle f_{31} \rangle$	1
	• Standard deviation of red colour pixels belong to the maximum area class, $\langle f_{32} \rangle$	1
	• Standard deviation of green colour pixels belong to the maximum area class, $\langle f_{33} \rangle$	1
	• Standard deviation of blue colour pixels belong to the maximum area class, $\langle f_{34} \rangle$	1
	• Skewness of red colour pixels belong to the maximum area class, $\langle f_{35} \rangle$	1
	• Skewness of green colour pixels belong to the maximum area class, $\langle f_{36} \rangle$	1
	• Skewness of blue colour pixels belong to the maximum area class, $\langle f_{37} \rangle$	1
	• Horizontal edge length, $\langle f_{38} \rangle$	1
	• Vertical edge length, $\langle f_{39} \rangle$	1
	• Total edge length, $\langle f_{40} \rangle$	1
Global Colour	• HSV histogram, $\langle f_{41}, ..., f_{104} \rangle$	64
	• Median intensity, $\langle f_{105} \rangle$	1
	• Mean saturation, $\langle f_{106} \rangle$	1
	• Percentages of pixels that fall into predefined ranges of HSV colourspace, $\langle f_{107}, ..., f_{117} \rangle$	11
Global Texture	• RGB colour coherence vector, $\langle f_{118}, ..., f_{245} \rangle$	128
	• HSV colour autocorrelogram, $\langle f_{246}, ..., f_{501} \rangle$	256
	• Image contrast, $\langle f_{502} \rangle$	1
Total		502

Table 3.2: Description of feature set.

3.6 Feature Space Reduction

At this point, for each image we have a lot of dimensions. There are 502 in our case. With higher dimension, the time required to run a search query rises exponentially. If a linear scan is used, yielding results may not run within the necessary time (usually, a few ms). Methods that create new features based on transformations or combinations of the original feature set are also called feature extraction algorithms.

Feature extraction methods determine an appropriate subspace of dimensionality

m in the original feature space of dimensionality d ($m \leq d$). Linear transforms, such as principal component analysis, factor analysis, linear discriminant analysis and projection pursuit have been widely used in image processing algorithms for feature extraction and dimensionality reduction, see Jain et al. (2000) and Wang et al. (2000).

The best known linear feature extractor is the principal component analysis (PCA) or Karhunen-Loève expansion, that computes the m largest eigenvectors of the $d \times d$ covariance matrix of the n d-dimensional patterns. The linear transformation is defined as

$$Y = XH, \tag{3.14}$$

where X is the given $n \times d$ pattern matrix, Y is the derived $n \times m$ pattern matrix, and H is the $d \times m$ matrix of linear transformation whose columns are the eigenvectors. With PCA, the transform is designed to decorrelate the features, and only those features with eigenvalues larger than a threshold will be retained.

Looking at the Table 3.2, three physical feature subsets have already created (features based on segmentation F_{SG}, features based on global colour F_{GC} and features based on global texture F_{TX}). Given an almost endless list of image features that one can come up with, a natural question to ask is whether they provide independent information about the image content, and if not, how to derive a reduced set of features that can best serve the purpose. One way to measure the correlation among features within the same feature subset and across different subset is by computing the covariance matrix

$$\mathbf{C} = \frac{1}{N} \sum_{\mathrm{F} \in \mathcal{F}} (\mathrm{F} - m)(\mathrm{F} - m)^T \text{ with } m = \frac{1}{N} \sum_{\mathrm{F} \in \mathcal{F}} \mathrm{F}, \tag{3.15}$$

where $\mathrm{F} = (f_1, f_2, ..., f_K)^T$ is a K-dimensional feature vector, $\mathcal{F}$ is the set containing all feature vectors derived from training sequences, and N is the total number of feature vectors in $\mathcal{F}$. The normalized correlation between features i and j is defined by

$$R(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j)}}, \tag{3.16}$$

where $C(i,j)$ is the $(i,j)^{th}$ element in $\mathbf{C}$. Figure 3.15 shows the correlation matrix between features from different subsets, after some preprocessing took place to remove feature vectors where min(F) = max(F). The first 40 features are segmentation features denoted as F_{SG}, the next 77 are global colour features F_{GC} and the last 344 are texture features F_{TX}. High correlation between certain features suggests that the feature dimension can be reduced applying PCA. We can get a quick impression of the F_{SG} data by plotting a box plot. We can see in Figure 3.16 that there is substantially more variability in f_{38}, f_{39}, f_{40} than the other features. As a result from PCA we keep the first principal components that explain over 90% of the total variability. The size of the new data will be as in Table 3.3. These results will be used in Chapter 6. An output of the PCA analysis can be found in Appendix A.3.

Feature subsets	**Before PCA**	**After PCA**
F_{SG}	40	19
F_{GC}	77	45
F_{TX}	385	59
TOTAL	502	123

Table 3.3: Dimensionality reduction using PCA.

3.7 Discussion

Image feature (content) description is the basis of CBIR. The most common visual characteristics represented are low-level features that represent colour, texture and spatial properties. Automated feature computation can be computationally demanding but can typically take place off-line prior to the retrieval process. Due to the subjective nature of visual information, no optimal feature descriptor exists for all tasks, although many representations have been proposed.

The colour histogram is the most traditional way of describing low-level colour properties of images; (Swain and Ballard, 1990). We call this low-level because it is extracted directly from digital representations of images in the database and has little or nothing to do with human perception. It is obtained by discretizing image colours and counting how many pixels belong to each colour. A recent proposal for how colour histograms can be effectively compressed with benefit in the speed of a CBIR system is presented in Berens et al. (2000). An approach which identifies the regions within images that contain colours from predetermined colour sets has been presented in Smith and Chang (1995). A compact colour descriptor and an efficient indexing method for this descriptor are presented in Deng et al. (2001), where colours in a given region are clustered into a small number of representative colours. The feature descriptor consists of the representative colours and their percentages in the region. The representative colours can be indexed in the three-dimensional colour space thus avoiding the high-dimensional indexing problems associated with the traditional colour histogram. A scheme of image retrieval from a database using queries prompted by only the colour and the shape of the objects present in different scenes has been proposed by Fuertes et al. (2001). They focus attention on the modules that allow feature extraction of the component objects from the scenes and the matching of the objects among the different images. Two new colour indexing techniques have been proposed by Stricker and Orengo (1995). The first one is that in the index they store the cumulative colour histograms as the second technique instead of storing the complete colour distributions, the index contains only their dominant features which are the first three moments of each colour channel of an image.

Textural features, based on variances of gray level spatial dependencies and line-angle-ratio statistics, for image database retrieval are presented in Aksoy and Haralick (June 1998), where the texture has been specified by the statistical distribution of the spatial relationships of gray level properties. Coarse textures are ones for which the distribution changes slightly with distance, whereas for fine textures the distribution changes rapidly with distance. This information can be summarized in gray level co-occurence matrices that are matrices of relative frequencies $P(i, j; d, \theta)$ with which two neighboring pixels separated by distance d at orientation θ occur in the image, one with gray level i and the other with gray level j. In order to use the information contained in the gray level co-occurence matrices 14 statistical measures (entropy, inertia, energy) have been defined in Haralick et al. (1973). Therefore, an image can be roughly represented by the lines extracted from it. Before feature extraction, each image is processed offline by an edge detector, edge linker, line selection operator and line grouping operator to detect line pairs. The features for each pair consist of the angle between two lines and the ratio of mean gray level inside the region spanned by that angle to the mean gray level outside that region.

Spatial entities are points, lines, regions and objects. They are hard to compute because of the segmentation problem difficulty. Since shapes bear a semantic meaning, automatic segmentation algorithms are not able to extract them in a reliable way from images, unless some knowledge of the application domain is available. Therefore, when the database is populated, it is usually requested that users identify shape contours either manually or with the help of special tools. Spatial relationships between entities often capture the most relevant and regular part of information in an image.

The second main category includes the *semantic* or *high-level* features. These features are abstract representations of images at various levels of detail. High-level semantic concepts play a large role in the way we perceive images and measure their similarity. Unfortunately, these concepts are not directly related to low-level image attributes but these attributes frequently capture information about the semantic meaning. For instance, autocorrelograms, colour coherence vectors, features based on segmentation, have proven to be such image features with some capability denoting semantics. Although, as we have seen in this Chapter, there are algorithms to describe colour, spatial and texture features, but these algorithms do not adequately model image semantics and therefore have many limitations when dealing with broad content image databases. Yet, due to their computational efficiency the low-level features are widely used for image retrieval, leaving the user interface of CBIR system with a task of bridging the gap between the low-level nature of these features and the high-level semantics people use to judge image similarity. Some high-level features may be synthesized from low-level features (segmentation features; sky on the top land on the bottom) whereas others can only be obtained through considerable human involvement (keyword annotation). High-level features denote the deeper domain semantics manifested in the images. At this level, progress has been more limited. Retrieval at this level involves matching derived or abstract attributes of the image such as the presence of specified types of building or animal in a scene. This kind of feature matching is qualitatively different, as it requires not just feature analysis, but a complex inferential process, making reference to some stored knowledge base of object paradigms. No amount of colour, texture or shape analysis can tell us on its own whether a particular image represents a dog, since dogs come in many shapes, sizes and colours. The ease with which humans can recognise an image as a dog conceals a complex and as yet incompletely understood process. For this reason, there are many who consider that the only way to provide this information in a CBIR system is to employ human indexers to add suitable keywords. Identification of the objects or individuals in a scene is formidable enough task; attempting to understand the significance of their presence or location in the scene well-nigh impossible without considerable intellectual effort. It might be possible to teach an image understanding system that a combination of small human beings around a table bearing a cake topped with candles can be interpreted as a child's birthday party. But identifying the connotations of a picture of, say, the "happy atmosphere" in the child's party, it is difficult to see what features can describe this and how retrieval at this level could be automated. Even the former task requires access to a knowledge base of formidable complexity. Prospects for viable image retrieval at this level seem small though this has not deterred us from attempting to search for "violent", "sad", "romantic" images, see Chapter 7. Kato (1992) attempted to index a collection of paintings by generating subjective desriptors such as "warm" and "romantic" directly from their colour histograms.

In summary, significant progress has been made in delivering usable image retrieval based on low-level features, though even here there is still considerable scope for improvement in both search efficiency and effectiveness. By contrast, progress at image retrieval based on high-level features has been extremely slow, though some potentially promising lines of investigation have been identified. Evidence both from informal discussions with picture librarians and from studies such as Enser (1993) makes it quite clear that while retrieval using low-level features might be nice to have, most users' real needs can be met only by providing higher levels of retrieval. There is thus a significant gap between users' needs and the capabilities of even the most advanced of today's experimental systems.

Most of the features were calculated based on PicHunter system in Cox et al. (2000), Cox et al. (1996). We have extended those features in the way that we accommodated features based on classification. Ultimately, some shape information or object-based scene description could be employed in CBIR systems. There is no feature set can hope to distinguish itself well for all possible queries. Research in new features is on-going. However the emphasis in this work is the feedback algorithm, rather than feature selection.

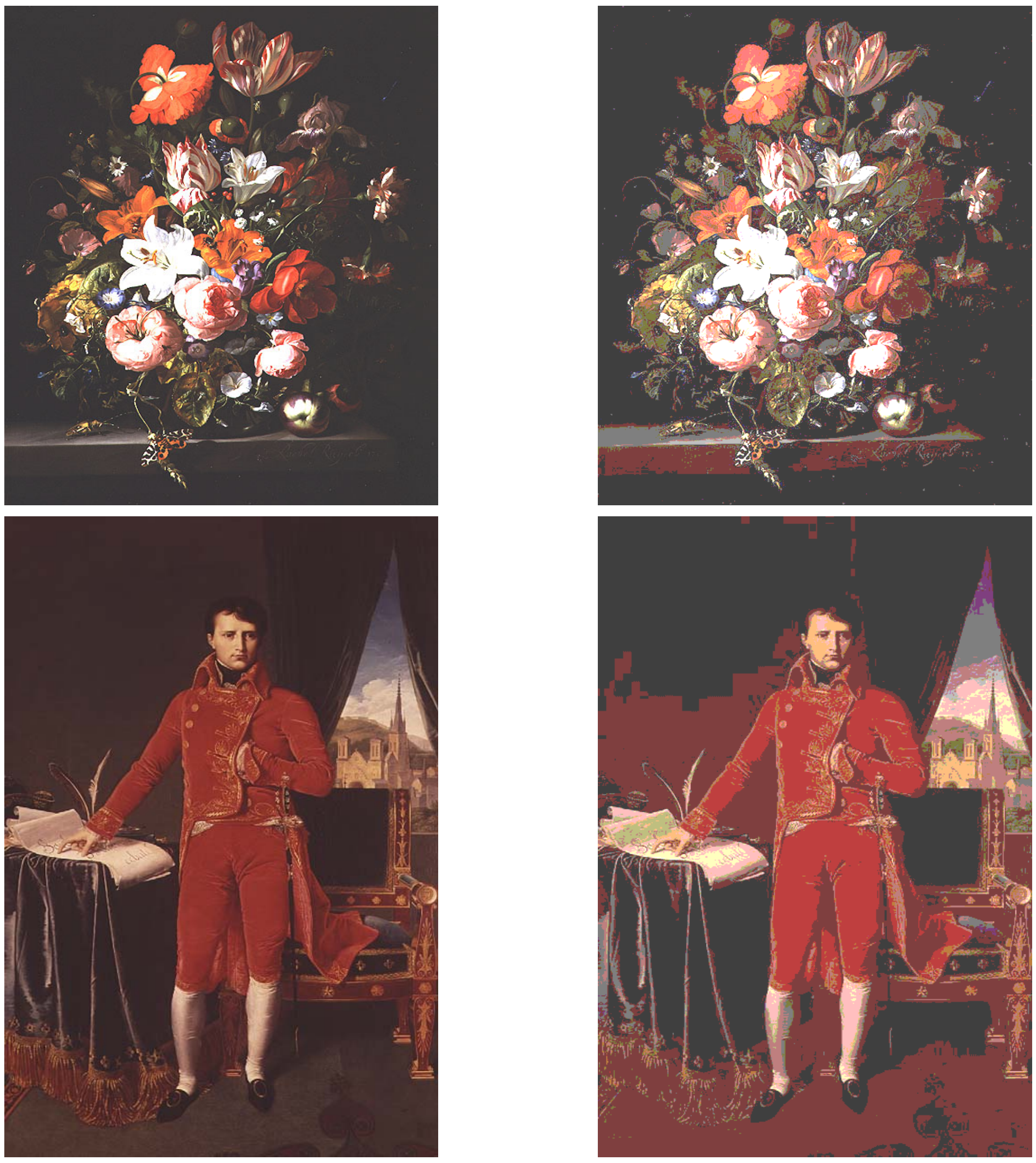


Figure 3.12: RGB Truecolour images (left) and the Quantized RGB images having only 64 colours (right).

7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
7														7
7		5	5	5	5	5	5	5	5	5	5	5		7
7		5										5		7
7		5		3	3	3	3	3	3	3		5		7
7		5		3						3		5		7
7		5		3		1	1	1		3		5		7
7		5		3		1	**p**	1		3		5		7
7		5		3		1	1	1		3		5		7
7		5		3						3		5		7
7		5		3	3	3	3	3	3	3		5		7
7		5										5		7
7		5	5	5	5	5	5	5	5	5	5	5		7
7														7
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7

Figure 3.13: Distances in lattice used for autocorrelogram computation.

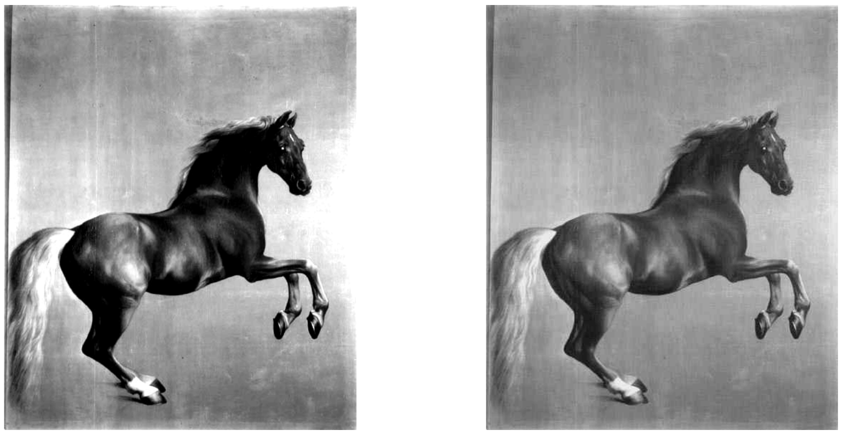


Figure 3.14: A grayscale image with high contrast (left) and low contrast (right).

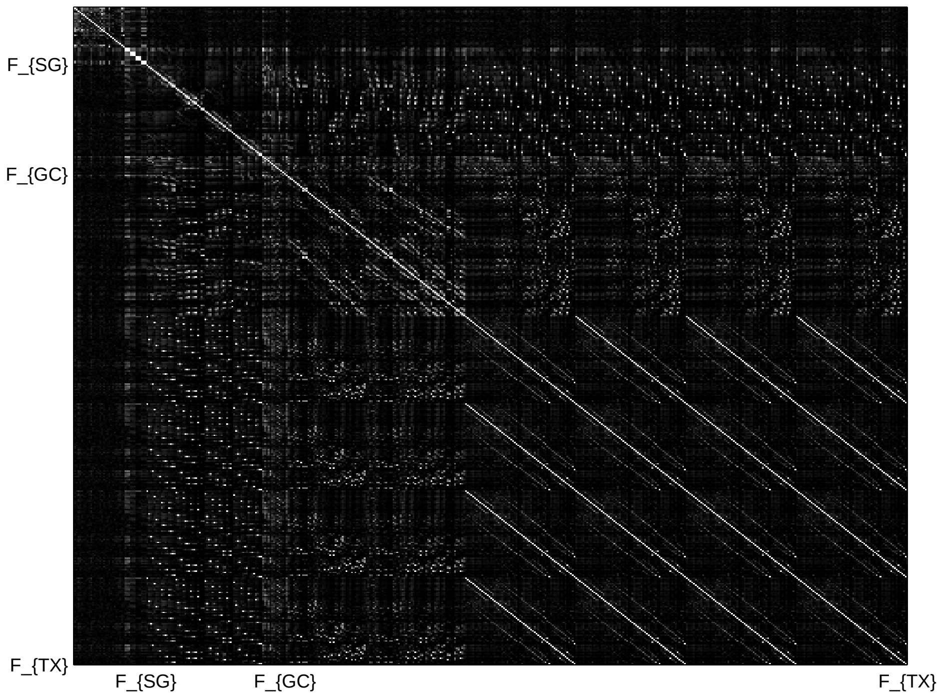


Figure 3.15: The normalised correlation matrix.

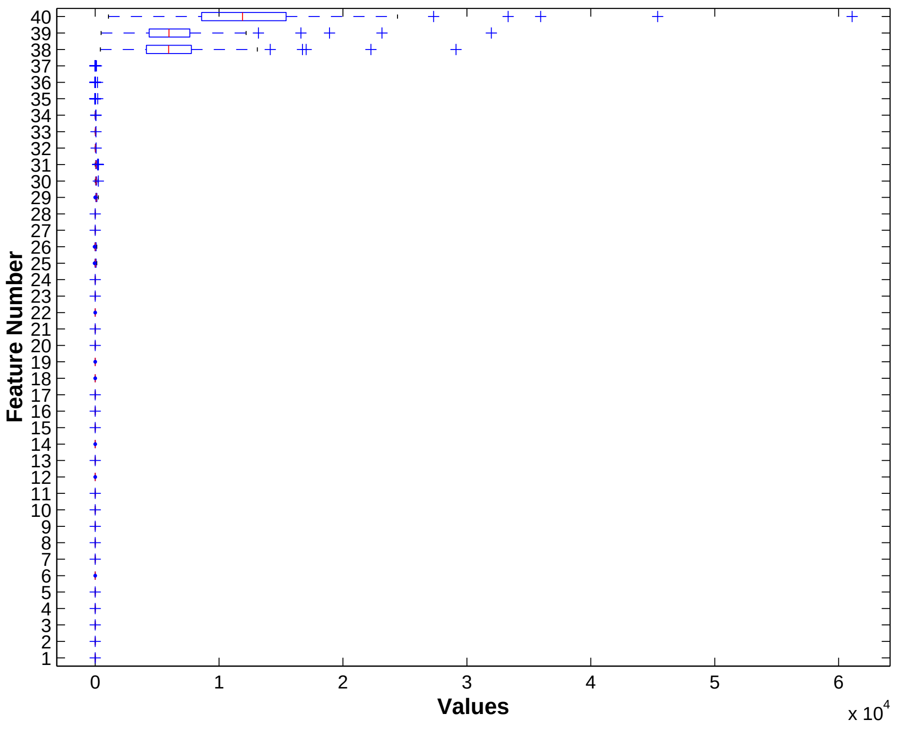


Figure 3.16: A box plot for the segmentation features.

Chapter 4

Content Based Image Retrieval

4.1 Introduction

This chapter describes the general mechanism of CBIR as well as the general issues arising from CBIR. It reviews some of the most well known CBIR systems and concludes with a description of the Bayesian Image Retrieval System Pichunter (Cox et al., 2000), on which this work is based. An implementation of Pichunter to our two image databases is also presented.

Mankind has always preferred concrete visual means (images, painting) to more abstract counterparts (written text) to express ideas and present information. Recent technological advances in handling digital data have further strengthened our dependence on visual modes of communication. This can, for instance, be witnessed in the explosively growing amount of digital image data, especially with the proliferation of the world wide web. Though it is fairly easy to create a large repository of visual data, the information stored there is virtually useless if it is unorganized. It is not unreasonable to draw an analogy between looking for a particular image on the web and searching for a book from a huge library without the aid of catalogues. The importance of the ability to search and retrieve images from an image collection cannot be overemphasized. Scientific organizations like NASA, broadcasting companies like BBC, CNN, photo agencies like MAGNUM that possess large archives of photos and videos have to invest considerable efforts to manage this data properly so that search and retrieval become feasible.

The next obvious question is: how difficult is the problem of searching for images? Could one use the well-studied methods in text retrieval for images? The answer is unfortunately no. It is much harder to handle image data than text. A text document is "one-dimensional" (an array of words), whereas a digital image is "two-dimensional" (and video is "three-dimensional" with an additional time component). In addition, the size of image data is much larger than text. Finally, words are semantic "objects", while the image data need to be processed and interpreted to extract meaning, which is a difficult and unsolved problem in computer vision and image understanding. Image data is likely to be explored, navigated, and retrieved by perceptual similarity, which is hard to capture.

One approach for indexing and retrieving image data is using manual text annotations. The annotations can then be used by some text information retrieval systems to search images indirectly. Typical searches for these first-generation visual information retrieval systems would be, for example, for "all images of paintings of the Florentine school of the XV century" or "all images of landscapes by Cezanne" (Bimbo, 1999). There are several inherent problems with this seemingly attractive approach. First, since image data contains very rich information, it is very difficult to capture the content of an image using only a few keywords, not to mention the tedious work involved in such an annotation process itself. Second, the manual annotation process is quite subjective, ambiguous, and incomplete. If a query refers to image content that was not initially annotated, or if the user uses different words to describe the same image content, the text retrieval system will fail.

Fleck et al. (1996) observe that most CBIR systems adopt the following two-step approach to search image databases:

- *Indexing* for each image in a database, a feature vector of properties of the image is computed and stored in a database;
- *Searching* given a query image, its feature vector is computed, compared to the feature vectors in the database, and images most similar to the query image are returned to the user.

Such a system is called Query-By-Example (QBE). In a typical QBE system, a user poses a query by providing an existing image (or creates one by drawing), and the system retrieves other "similar" images from the image database.

Besides the basic content-based image retrieval tasks, several related problems also need to be addressed. If the user gets back irrelevant results, he or she can provide feedback information that describes the relevance of the retrieved results. The CBIR system should then process this feedback efficiently and return better results according to the user's criteria. While most CBIR systems retrieve images based on an overall image appearance comparison, users are also interested in object searching; see Forsyth et al. (1996) and Ravela and Manmatha (1997). In the case of object searching, the user specifies an "interesting" subregion (usually an interesting object) of an image as a query. The system should then retrieve images containing this subregion or object from the image database.

Most CBIR systems generate low-level image features such as colour, texture and shape for image indexing and retrieval. This is partly because low-level features (e.g., colour histograms, texture patterns) can be computed automatically. The semantics of images, with which users prefer most of their interaction, however, are seldom captured by low-level features. On the other hand, there is no effective method yet to automatically generate good semantic features of general images. No contemporary computer vision or image understanding technique can automatically annotate the contents of generic real-world scenes. One common compromise is to obtain some semantic information through manual annotations, which is neither accurate nor efficient as mentioned before. Therefore one currently relies on low level features, in the hope that these can at least capture some of the semantic content of the image that the user uses in the search process.

Given this, the importance of feedback becomes clear. Given the lack of good automatic semantic feature extraction tools, we must rely on the user to give feedback to aid the search process. It is unrealistic to expect a computer to relate low level features to semantic content without this help.

4.2 Functions of a Typical CBIR System

A typical CBIR system deals not only with various sources of information in different formats (for example, text, image, video) but also users' requirements. Basically it analyses both the contents of the source of information as well as the user queries, and then matches these to retrieve those items that are relevant. The major functions of such a system are the following:

1. Analyze the contents of the source information, and represent the contents of the analyzed sources in a way that will be suitable for matching user queries.

The space of source information is transformed into feature space for the sake of fast matching in a later step. This step is normally very time consuming since it has to process sequentially all the source information (images) in the database. However, it has to be done only once and can be done off-line.

2. Analyse user queries and represent them in a form that will be suitable for matching with the source database. Part of this step is similar to the previous step, but applied only to the query image.

3. Define a strategy to match the search queries with the information in the stored database. Retrieve the information that is relevant in an efficient way. This step is done online and is required to be very fast. Modern indexing techniques can be used to reorganize the feature space to speed up the matching processing, when the number of images in the database is very large. If the database is of a small/medium size (a few thousand images like the BAL database), the whole index can fit in main memory. A presentation or implementation of indexing techniques is out of the scope of this thesis; good references are Bimbo (1999) and Santini (2001).

4. Make necessary adjustments in the system (usually by tuning parameters in the matching engine) based on feedback from the users and/or the retrieved images.

It is evident from the above discussion that, on the one side of a CBIR system, there are sources of visual information in different formats, and on the other there are the user queries. These two sides are linked through a series of tasks as illustrated in Figure 4.1.

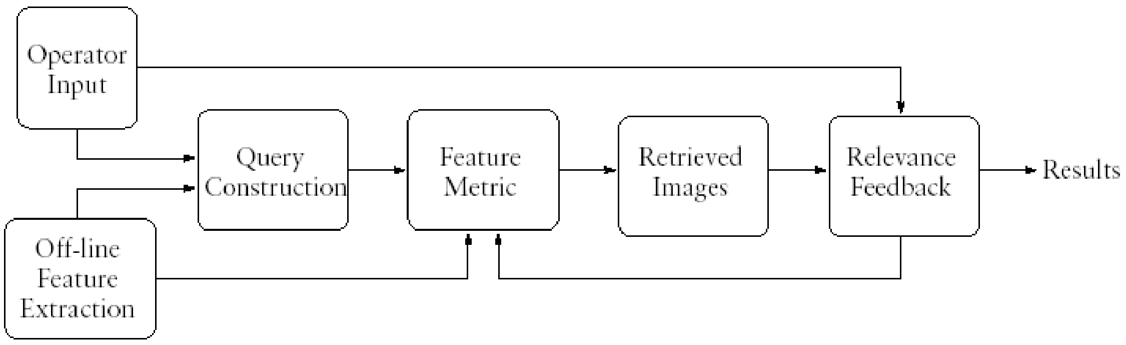


Figure 4.1: The format of a typical CBIR system.

4.3 Types of Content Based Search

It is worthwhile to better define what types of content based search exist. Cox et al. (1996) identify three classes of search:

- **Target search** Users try to find specific target images. For example, an art-history student might need to find a specific painting, or a graphic artist might look for a specific, known stock photo.

- **Category search** Users seek one or more images from general categories such as: "sunsets", "birds", "hats", "Volkswagon Beetles" or "pictures of the Eiffel Tower". This is more complex than target search because it places more emphasis on the semantic content of images like "happy sunset", "melancholic sunset", "kinds of birds", "a Panama hat" and often requires subjective judgments.

- **Open-ended browsing** Users have, at best, vague ideas of what they are looking for, and may change their minds repeatedly during their searches. Many graphic design problems have to do with finding the right image for the right application in cases in which the concept of "right image" is extremely elusive. When a designer is creating, say, a brochure for a new product, he typically starts with a more or less imprecise idea of the thematic content of the final brochure, and then searches catalogues from some stock images provider to look for inspiration. The consummate browser may have no specific goal in mind at all, simply wandering through a database for fun.

Each search class is important, but it is hard to define what *correct* behaviour means for the second and third classes.

4.4 Pre-attentive and Attentive Similarity

We usually distinguish between *pre-attentive* and *attentive* human similarity. Attentive similarity has to do with interpretation. It usually involves previous knowledge and a form of reasoning. Pre-attentive similarity instead is simply based on the perceived similarity between stimuli, with no form of interpretation.

Figure 4.2 shows an example of such a distinction. The two faces are in an unfamiliar position but, at first glance, seem to be the same. Turning the page upside down, so that the faces are in a more familiar position, the face images are interpreted and the differences in eye and mouth contour can be realised. In the first case, pre-attentive similarity is involved. Features considered are those prominent in the image, like colour, texture, presence and location of salient elements of a face such as eyes, mouth, nose; whereas, the precise identification of these elements and their recognition through feature matching are tasks performed when attentive similarity is involved.

Attentive similarity is used in domain-specific retrieval applications such as facial, mechanical parts or medical databases. These use object matching and ad-hoc similarity criteria, particular to the application domain. Pre-attentive similarity, instead, is requested in retrieval applications where colour, texture, shape and spatial relationship perception is more important. This is the application area in this thesis. These need similarity models that closely conform to human similarity perception of sensorial stimuli.

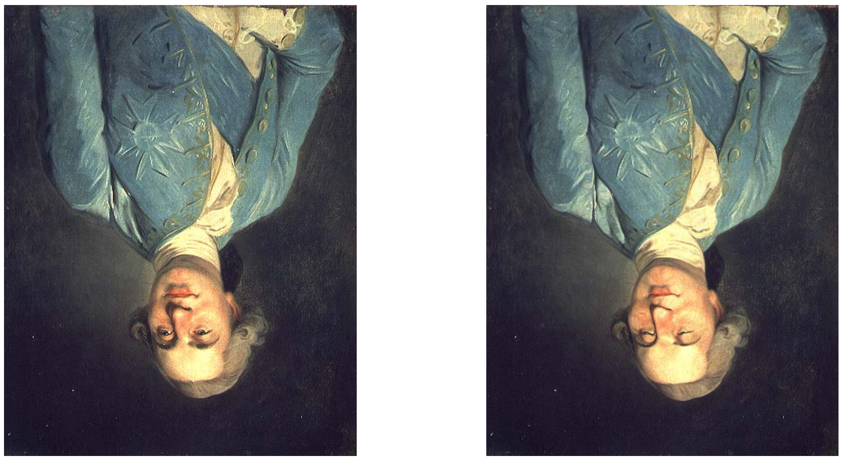


Figure 4.2: Right side up: pre-attentive similarity assessment of two images of the same face. Upside down: attentive similarity assessment of the same two images.

4.5 Defining a Distance Measure on Images

Of crucial importance in user modeling is the design of similarity metrics used to compare current query and image database feature vectors.

An important component of the model for relevance feedback is a distance between images. A very well-known theory postulates that human similarity perception is based on the measurement of an appropriate distance in a metric psychological space. A psychological space is a region in which we may place and classify elements of our experience. In fact, human similarity perception is based on the measurement of an appropriate distance in a metric psychological space, whose form is doubtless quite different from the metric spaces typically used for vector comparison. Hence, to be truly effective, feature representation and feature matching models should somehow replicate the way in which humans assess similarity between different images.

Letting $f(X) \in \mathbb{R}^p$ be a vector of p features of the image X, then we can define the distance between two images X and Y to be the distance in $\mathbb{R}^p$ between their feature vectors:

$$d(X, Y) = \|f(X) - f(Y)\|. \tag{4.1}$$

The Minkowski metric is widely used for measuring similarity between images. Suppose two images X and Y are represented by two p dimensional vectors $\langle x_1, x_2, ..., x_p \rangle$ and $\langle y_1, y_2, ..., y_p \rangle$, respectively. The Minkowski metric $d(X, Y)$ is defined as

$$d(X, Y) = \left(\sum_{i=1}^{p} |x_i - yi|^r \right)^{\frac{1}{r}}, \tag{4.2}$$

where r is the Minkowski factor for the norm. Particularly, when r is set as 2, it is the well known Euclidean distance; where r is 1, it is the Manhattan distance (otherwise known as the L_1 or City Block distance). An image feature vector located a smaller distance from a query image feature vector is deemed more similar to the query image. For further discussion on metrics in feature space, see Wilson et al. (1997), Li et al. (2002), Santini and Jain (1999) and Bimbo (1999). An alternative approach is to define a distance measure directly on the image matrix. Baddeley (1992) proposed a distance measure for binary images, while Rue and Hurn (1997) extended to categorical images (such as pixel labellings in a segmentation). We have not seen any work in image retrieval that attempts to compute distances in this way.

The intention is that images that are close semantically are also close in feature space. Of course this can never be true for all circumstances and all interpretations of a image, a fact that is known as the semantic gap.

4.6 The Semantic Gap

As we have already described, a database engine analyses an image and extracts a number of features to describe it. Most CBIR systems use simple perceptual quantities or low level image features because of the very diverse nature of possible queries and the frustrating unreliability of object recognition systems. Typical features are: colour histograms, local colour histograms, edge elements and their direction (e.g horizontal/vertical) and texture descriptors. These descriptors are collected into a feature vector, and the similarity between two images is determined to be a decreasing function of the distance between their respective feature vectors.

A similarity query using features like these will give results like those in Figure 4.3. If we look at this answer we may at first be a little baffled. We were looking for something that was a framed picture, and the result contained a couple of arched corridors, some portraits and an active volcano. The similarity criterion is specified to be the Euclidean distance between mixture of normalised colour, texture and segmentation features. With this similarity in mind, consider the man in the top-right corner. Note that the curve of his forehead has almost the kind of shape and size of the arch on top of the framed picture in the query. Also, there are other superficial similarities, in both the query and the image of the man's face that the arch encloses contains a section of the image with many edges, while outside the arch there are not as many edges. This similarity in structure was obviously strong enough to give the image a high score, and make it appear in the third position.

Similar considerations apply to the image of the man appearing in the fourth position. In this case the colour, position and size of the vest he is wearing make the image relatively similar to the query. Most of the images returned by the CBIR engine can

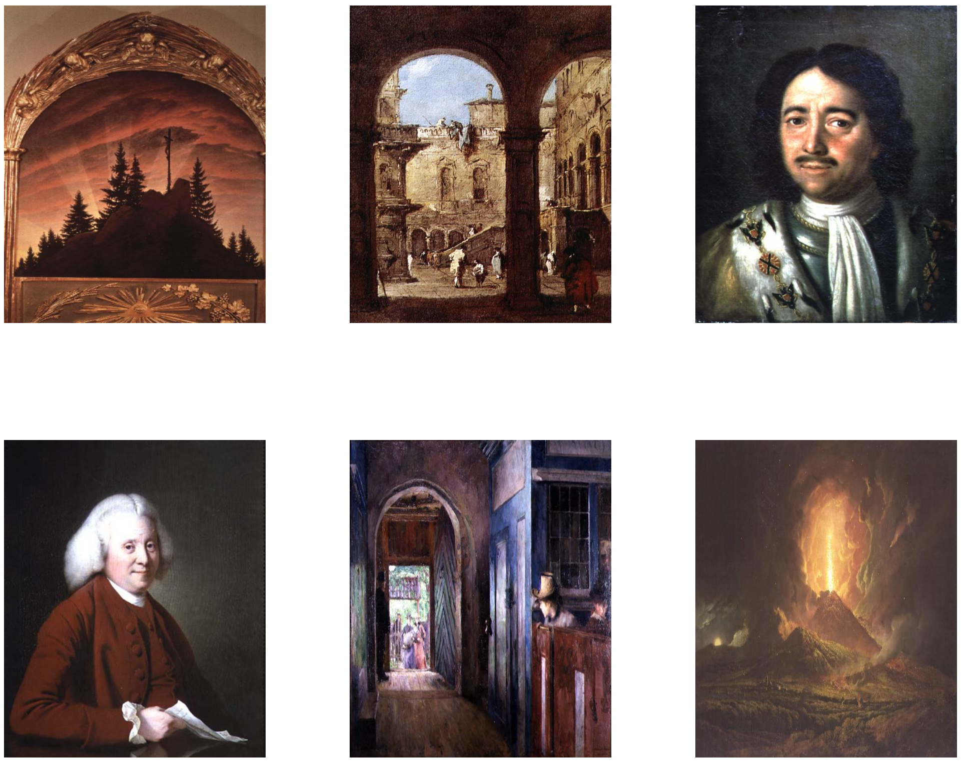


Figure 4.3: A query (the image in the top-left corner of the figure) and the five images closest to it in the BAL image database.

be explained this way. Most of the retrieved images have similar structure with the query image. Some images are not so easy to explain. For instance, it is not easy to justify the presence of the "active volcano" image (bottom right corner). After a closer investigation at the raw feature numbers we found that the "active volcano" image was close on the textural and global colour features like f_{77} (component of the HSV histogram to do with green and blue colour), f_{95} (component of the HSV histogram to do with purple and red colour), f_{248} (component of the HSV colour autocorrelogram), f_{312} (component of the HSV colour autocorrelogram), f_{376} (component of the HSV colour autocorrelogram), but the distance on some other features like f_{11} (largest distance between centroids), f_{20} (number of centroids in the 5^{th} block), f_{26} (eccentricity of maximum area), f_{253} (component of the HSV colour autocorrelogram), f_{317} (component of the HSV colour autocorrelogram) was large.

Santini (2001) expresses this finding informally as follows. The system does not know that we are looking for a framed picture. The similarity criteria form a framework into which the system works or in other terms, they give the database a logic to which to conform. The database, in general comes up with an answer that makes sense within this framework. One does not expect it to give the exact answer 100% of the time; see also Santini and Jain (1998) and Santini and Jain (1997).

Therefore, the problem is not that the database is wrong according to the semantic induced by the similarity criterion. The problem is that the similarity criterion does not successfully capture the semantics intended by the user. The problem lies in the different semantic spaces in which the user and the database operate. The database operates in an environment whose objects are colours, edge fragments, texture patches, and so on. Colours can be bright or dark, edges have direction, and textures have a number of complex characteristics. The user operates in an environment of depressing framed pictures, particular type of corridors, serious faces, active volcanos and so on. It seem obvious that the semantic of the latter should somehow be based on that of the former, but CBIR systems, and indeed any other image processing method that attempts to model user semantics, are not very good at doing this.

4.7 Relevance Feedback

One technique, which helps towards reducing the semantic gap between the user's perception of image similarity and the feature-based representation of images, is relevance feedback. Relevance feedback is an interactive process in which the user judges the quality of the retrieval performed by the system by marking among the images retrieved by the system, the one that is perceived as truly relevant. This information is then used to refine the original query, resubmitted for a sharper selection. Most existing systems use relevance feedback strategies from text retrieval techniques, where the relative frequency of terms is used as indicator of term discrimination power. User feedback increases the contribution of terms appearing in documents marked as relevant.

Celentano and Chiereghin (1999) point out that image similarity is evaluated on visual properties, which should not be biased by the content of the whole image database. Moreover, the user can check at a glance the quality of the retrieved set, while reading retrieved texts to investigate their relevance requires much more time.

Relevance feedback techniques for image retrieval are very different from those for text because both image indices and image similarity functions are more complex. Meghini et al. (2001) argue that images have a much richer perceptual content than text and that the experimental evidence gathered so far seems to indicate that a representation of images that is, at the same time, as simple and as effective for retrieval as that of text is unlikely to exist. For these reasons, image indexes, in most CBIR systems, are collections of features (e.g. colour, texture, shape) where each feature is possibly multiply represented (e.g histogram, colour coherence vector), each representation being a multidimensional object, such as a vector, of its own. As a consequence, image similarity functions are typically linear combinations of distance measures relative to single features. Under these circumstances, relevance feedback can be used to calculate what emphasis should be given to each feature; see for example Rui et al. (1998).

Other usages of user judgments are possible, such as altering the query index

(Ciocca and Schettini, 1999). However, given the composite nature of such indexes, emphasizing and deemphasizing their components according to user preferences would seem more appropriate than numerically manipulating them.

Finally, other relevance feedback mechanisms are adopted to refine image-based queries by asking users to mark the set of images retrieved in a neighbourhood of the query as being relevant or not. Bayesian decision theory can be used to estimate the boundary between relevant and non-relevant images (Giacinto and Roli, 2004). Then a new query is computed whose neighbourhood is likely to fall in a region of the feature space containing relevant images.

4.8 CBIR systems

In recent years, CBIR has become a highly active research area. Many image retrieval systems, both commercial and for research, have been built. In the following discussion, we briefly describe some of the well-known CBIR systems that have been developed. A very nice survey about content based image retrieval systems can be found in Veltkamp and Tanase (2000).

4.8.1 QBIC

IBM's QBIC system (Faloutsos et al., 1994) is probably the best-known of all image content retrieval systems. It is available commercially either in stand alone form, or as part of other IBM products such as the DB2 Digital Library. It offers retrieval by any combination of colour, texture or shape as well as by text keyword. Image queries can be formulated by selection from a palette, specifying an example query image, or sketching a desired shape on the screen. The system extracts and stores colour, shape and texture features from each image added to the database. At search time, the system matches appropriate features from query and stored images, calculates a similarity score between the query and each stored image examined based on Euclidean distance, and displays the most similar images on the screen as thumbnails. Any retrieved image can be used as a seed for a subsequent query by example (relevance feedback). The latest version of the system incorporates more efficient indexing techniques, an improved user interface and the ability to search grey-level images. An online demonstration, together with information on how to download an evaluation copy of the software, is available on the World-Wide Web at `http://wwwqbic.almaden.ibm.com/`.

4.8.2 Virage

Virage is a content-based image search engine developed at Virage Inc. Similar to QBIC, Virage (Bach et al., 1996) supports visual queries based on colour, composition (colour layout), texture, and structure (object boundary information). A basic concept is that of a *primitive*, which denotes a feature's computation type and a matching distance. Virage goes one step further than QBIC in that it supports arbitrary combinations of the above three queries. A function for computing the similarity between two sets of feature data previously extracted must also be supplied by the developer. Its relevance feedback allow the users to adjust the weights associated with the features according to their own emphasis. The visual features (primitive) are classified as general (such as color, shape, or texture) and domain specific (face recognition, cancer cell detection, etc.). Various useful type of features can be added to the open structure, depending on the domain requirements. To go beyond the query by example mode, Gupta and Jain (1997) proposed a eight-component visual information framework. This framework includes an image-processing tool that:

- interactively segments the image or modifies the properties of a local region in the image;
- a feature-space manipulation tool which would allow the user to explore the feature space and to specify a neighbourhood query like this "if each image is viewed as a point in the n-dimensional feature vector space, find the nearest x images within distance d of this image";
- an object specification tool especially for domain-specific systems in which the object of interest occupies only part of the image;
- a measurement specification tool that is used in any domain in which the size of the objects or regions in an image is an important concern;
- a classification tool which would allow the user to perform a grouping operation on visual objects by specifying a grouping criterion on one or more visual features of interest; this grouping allows queries like "based only on elongation and texture, what are the major groups of tumor objects in this collection of magnetic resonance images? display two images from each group";
- a spatial arrangement tool which would allow the user to specify location-sensitive queries and move query objects denoted by the object specification tool to position them in the place of interest, "find all images containing all the same objects as this one but having them arranged differently";
- an annotation tool where the user can provide a keyword or a hidden annotation as in PicHunter (Cox et al., 2000);
- a data definition tool would enable applications in which the user has a prior set of models to characterise properties of the image; for a database of, say, chest

X-rays or face shots, the tool would help define a database schema (to define the contents), so the user can specify a query like "find other face shots with similar facial features, but bigger eyes, wider lips, and a scar on the left side of the eyebrow". The other task of a data definition tool is to support an ontology and examples of words for cases in which visual descriptions are too complex to create.

An example would be to create a set of image examples for the word "human" so these examples may be used to start a query on humans. The system is available as an add-on to existing database management systems such as Oracle or Informix. The AltaVista Photofinder and Illustra's Visual Intelligence system are two applications of Virage technology.

4.8.3 Excalibur Visual RetrievalWare

RetrievalWare is a content-based image retrieval engine developed by Excalibur Technologies Corp. It allows queries by example based on HSV colour histograms, shape, texture, brightness, colour layout, as query features. It also supports the combinations of these features and allows the users to adjust the weights associated with each feature. The software has been used in the Image Surfer system, which was used by the Yahoo! and Infoseek WWW search engines.

4.8.4 VisualSeek and WebSeek

VisualSEEk (Smith and Chang, 1996b) is a visual feature search engine and WebSEEk (Smith and Chang, 1997) is a World Wide Web oriented text/image search engine, both of which have been developed at Columbia University. The main research features are a spatial relationship query for image regions and visual feature extraction. The visual features used in these systems are colour sets and texture features. VisualSEEk supports queries based on both visual features and their spatial relationships. This enables a user to submit, for example, a sunset query as a red-orange colour region on top and blue or green region at the bottom as its "sketch". After the system returns the thumbnail images of the best matches, the user is allowed to search by example using the returned images.

To find the matches of a query image with a single region, queries on colour set, region absolute location, area and spatial extent are first done independently. The colour set similarity is computed by a Mahalanobis distance $d(c_q, c_t) = (c_q - c_t)^t A^{-1}(c_q - c_t)$, where c_q, c_t are two colour sets for the query and the target respectively, and A is the variance-covariance matrix. If the user has defined spatial boundaries for the query region, then its distance to a target region is 0 if the target region centroid falls inside these boundaries, and is given by the Euclidean distance between the centroids otherwise. The distance in area between two regions is the absolute value of the difference, while the distance between the minimum bounding rectangles, (w_q, h_q) and (w_t, h_t), of two regions is the Euclidean distance. The best matching images are taken by minimizing a total distance given by the weighted sum of the four distances mentioned. The results of a query are displayed in decreasing similarity order. Under each retrieved image, the distance to the query image is indicated.

On the other hand, WebSEEk is a web-oriented search engine. It makes text-based and colour-based queries through a catalogue of images and videos collected from the Web. The user initiates a query by choosing a subject from the available catalogue or entering a topic. The results of the query may be used for a colour query in the whole catalogue or for sorting the result list by decreasing colour similarity to the selected item. Also, the user may manually modify an image colour histogram before reiterating the search. The distance is based only on the colour histogram and is the same as the one is used in Visualseek. Images retrieved are displayed page by page in a descending similarity order.

The user has the possibility of selecting positive and negative examples from the result of a query in order to reformulate the query, as its relevance feedback. A demo can be found at `http://www.ctr.columbia.edu/~ jrsmith/`.

4.8.5 Photobook

Photobook (Pentland et al., 1994) is a set of interactive tools for browsing and searching images and has been developed at the MIT Media Lab.

Photobook uses colour, texture, and shape features though it will work with others. Features are compared using one out of a library of matching algorithms that Photobook provides. These include Euclidean, Mahalanobis, divergence, vector space angle, histogram, Fourier peak, and wavelet tree distances, as well as any linear combination of these. Later versions allow user-defined matching algorithms.

A unique feature of Photobook, at least when it was first proposed, is that it is able to select and combine the matching algorithms based on examples from the user. It calls this system FourEyes. This makes Photobook different from tools like QBIC and Virage, which all support search on various features but offer little assistance in actually choosing one for a given task.

Photobook consists of three subbooks from which shape, texture, and face features are extracted, respectively. Users can then query on the basis of the corresponding features in each of the three subbooks. In its more recent version, Photobook allows relevance feedback that allows feature selection.

4.8.6 Netra

Netra is a prototype image retrieval system developed in the University of California Santa Barbara Alexandria Digital Library (ADL) project (Ma and Manjunath, 1999) by the Department of Electrical and Computer Engineering. Netra uses colour, texture, shape, and spatial location information in the segmented image regions to search and retrieve similar regions from the database.

There are 2500 images from the Corel photo collection, organised in 25 categories, with 100 images in each category. You can select any one of them as the query image. All images in the database have been segmented into homogeneous regions. You can click on one of the regions and select one of the four image attributes: colour, spatial location, texture and shape. Instead of using an image example, you can also directly specify the colour and spatial location. The spatial location querying tool utilizes two bounding boxes to define the area of interest. The inner box is used to define the preferred area, and the box outside is used to constrain the objects to be within this area. Thus, if the object has any of its parts outside this outer box, they will not be considered.

Euclidean distance is used for region colour and shape similarity, and $L1$ distance is used for texture similarity. The first feature the user specifies is used to retrieve about 100 candidates. Then this feature and the possible other features together are used to order the retrieval result. The matched images are ordered according to the similarity criterion.

4.9 PicHunter

We conclude with a Bayesian image retrieval system, PicHunter. The reason we present it in a different section is that it forms the basis of our work.

The reason that we adopted this system is that Bayesian methods have been successfully adopted in many image analysis and computer vision problems, as we have already presented in Chapter 2. However, its use for content-based retrieval from image databases is just being realized (Vailaya et al., 2001). This gives us a space for interesting extensions to existing methods as well as for new proposals.

PicHunter is a prototype CBIR system that was developed 1997 in its simple form (simple low level features, most probable display scheme) and was updated in 2000 (more sophisticated features and informative display scheme). PicHunter makes four primary contributions to research on content-based image retrieval. First, it represents a simple instance of a general Bayesian framework which is described for using relevance feedback to direct a search. With an explicit model of what users would do, given what target image they want, PicHunter uses Bayes's rule to predict what is the target they want, given their actions. This is done via a probability distribution over possible image targets. Second, an entropy-minimizing display algorithm is described that attempts to maximise the information obtained from a user at each iteration of the search. Third, PicHunter makes use of hidden annotation where the users are not required to learn a vocabulary of linguistic terms before using the system, or even use a particular language. Finally, PicHunter introduces two experimental paradigms to quantitatively evaluate the performance of the system.

4.9.1 Relevance feedback in PicHunter

PicHunter implements a probabilistic relevance feedback mechanism, which tries to predict the target image the user wants based on his actions. A vector is used for retaining each image's probability of being the target. This vector is updated at each iteration of the relevance feedback, based on the history of the session (images displayed by the system and user's actions in previous iterations). The updating formula is based on Bayes' rule.

During each iteration $t = 1, 2, ...$ of a PicHunter session, the program displays a set D_t of N_D images from its database, and the user takes an action A_t in response, which the program observes. For convenience the history of the session through iteration t is denoted H_t and consists of $\{D_1, A_1, D_2, A_2, ..., D_t, A_t\}$.

The database images are denoted $T_1, ..., T_n$ and PicHunter assumes that the target is in the database; this is the case in all of our work. After iteration t the posterior probability that database image T_i is the user's target T, given the session history, is then written

$$P(T = T_i|H_t) = P(T = T_i|D_t, A_t, H_{t-1}) = \frac{P(A_t|T = T_i, D_t, H_{t-1})P(T = T_i)}{\sum_{j=1}^{n} P(A_t|T = T_j, D_t, H_{t-1})P(T = T_j)}. \quad (4.3)$$

The system's prior estimate on starting the session is denoted $P(T = T_i)$. The canonical choice of this prior assigns probability $1/n$ to each image in the database, but one might use other starting functions that use the results of earlier sessions, for example the posterior distribution from a previous similar query.

In computing the probability of a user to take a certain action A_t given the history so far and the fact that the target is indeed T_i, namely $P(A_t|T = T_i, D_t, H_{t-1})$, one approach is to estimate the probability of the user to pick an image X_a from the displayed set of images $D_t = \{X_1, X_2, ..., X_{N_D}\}$ by

$$P(A_t|T = T_i, D_t, H_{t-1}) = \frac{\exp(-d(X_a, T)/\sigma)}{\sum_{i=1}^{N_D} \exp(-d(X_i, T)/\sigma}. \tag{4.4}$$

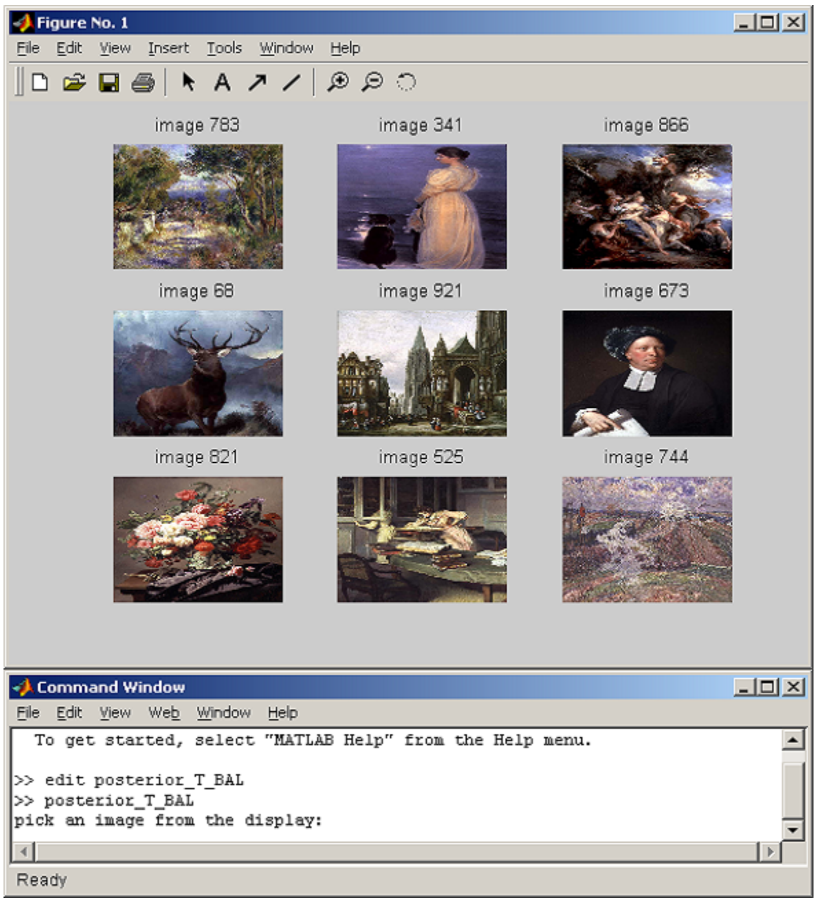
where d is a distance in normalised feature space. Note that this assumes that, conditional on T, the user's choice is independent of previous choices H_t. In the current PicHunter the distance which is used for similarity matching between the query image and every image in the database is $L1$ distance, and σ is a parameter of the model chosen to be the maximize likelihood estimator. The parameter σ can be interpreted as the degree of precision in the distance measurements.

After iteration t the program must select the next set D_{t+1} of images to display. One strategy for doing so selects the N_D most likely images from the posterior $P(T = T_i|H_t)$. This is a target search oriented strategy, where the users are required to find a specific target image in the database. For *category search* and *open-ended search* (browsing) other possibilities are explored later in this thesis.

4.9.2 User interface

All the systems, that we have developed in this thesis, have been implemented using MATLAB software. We keep the user interface as simple as possible. The entire user interface of our version of the PicHunter system is shown in Figure 4.4. At any given time during a search, nine images are displayed on the screen when using the system with the BAL database and six images are displayed when using the system with the IGN database. We display less images when we use the IGN database because of its small size (146 images). The user can select one image by typing its number in the command window. After typing an image's number in the command window the user calls up the next set of nine images by hitting the "enter" button on the keyboard.

The user continues selecting images by typing their number in the command window and pressing enter until the target appears. At the Figure window there is a number which is denote the current iteration of the search. At the top of every image is its number.



```
  To get started, select "MATLAB Help" from the Help menu.

>> edit posterior_T_BAL
>> posterior_T_BAL
pick an image from the display:
```

Figure 4.4: User interface of our system.

4.9.3 Effect of Altering σ

Different values of σ cause the probability $P(T = T_i|H_t)$ to be assigned differently in every image in the database. We consider the following three cases for three values of σ.

Case 1, σ=0.1

In Figure 4.5 we can see four iterations of search, where σ has been set manually at 0.1. We would like to find an image with a dominant urban area as our target image. In the first iteration (top left) six random images appear where the probability distribution over the database is uniform; image 57 is selected. This is followed by three iterations in which images with urban area start to be displayed. At the second iteration (top right) image 54 has been selected and at the third iteration (bottom left) image 65 has been selected. We conclude that the highest probability after four iterations is assigned to image 65 and this is $P(T = T_{65}|H_4) = 0.85586$.

Case 2, σ=1

Working on the same way we present the outcome for the identical retrieval search as in case 1 but now σ has been set 1. Picking exactly the same sequence of images as in case 1, we can see that the retrieval results are pretty much the same but there are some rearrangments of the retrieved images. The highest probability has been assigned for image 65 but now is less ($P(T = T_{65}|H_4) = 0.14094$) than in case 1 (Figure 4.6).

Case 3, σ=10

Finally, we present the results (Figure 4.7) of the same identical search for a bigger value of σ. The highest probability after four iterations has been assigned to the image 65. But this probability has been dramatically small ($P(T = T_{65}|H_4) = 0.011056$) comparing the previous cases. We should mention that in all these three examples the retrieval results are more or less the same related to the retrieved image themes, except the ordering of some of the retrieved images. The ordering of the images depends on σ. It is a real property of the algorithm; as sigma changes, the ordering of the images

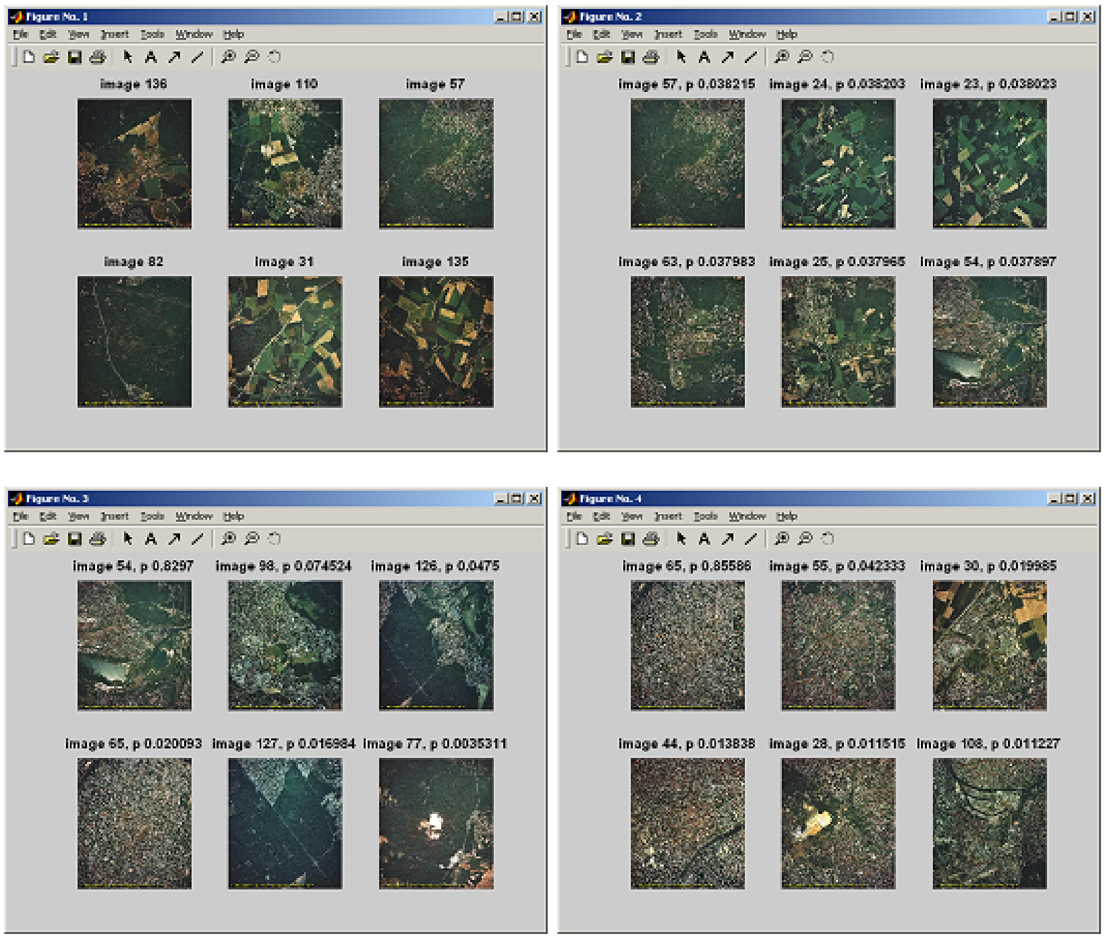


Figure 4.5: Four iterations of a search with σ=0.1.

in the posterior changes.

We have set a value of σ=0.5 for our experiments in this chapter because it gives reasonable probability values and there is a smooth change to the posterior probability $P(T = T_i|H_t)$ as the user searches the system. However this value is application dependent, and with another database would most likely be different. Later we discuss how its value may be inferred from the search process; the purpose of these experiments is to illustrate how the basic PicHunter system performs.

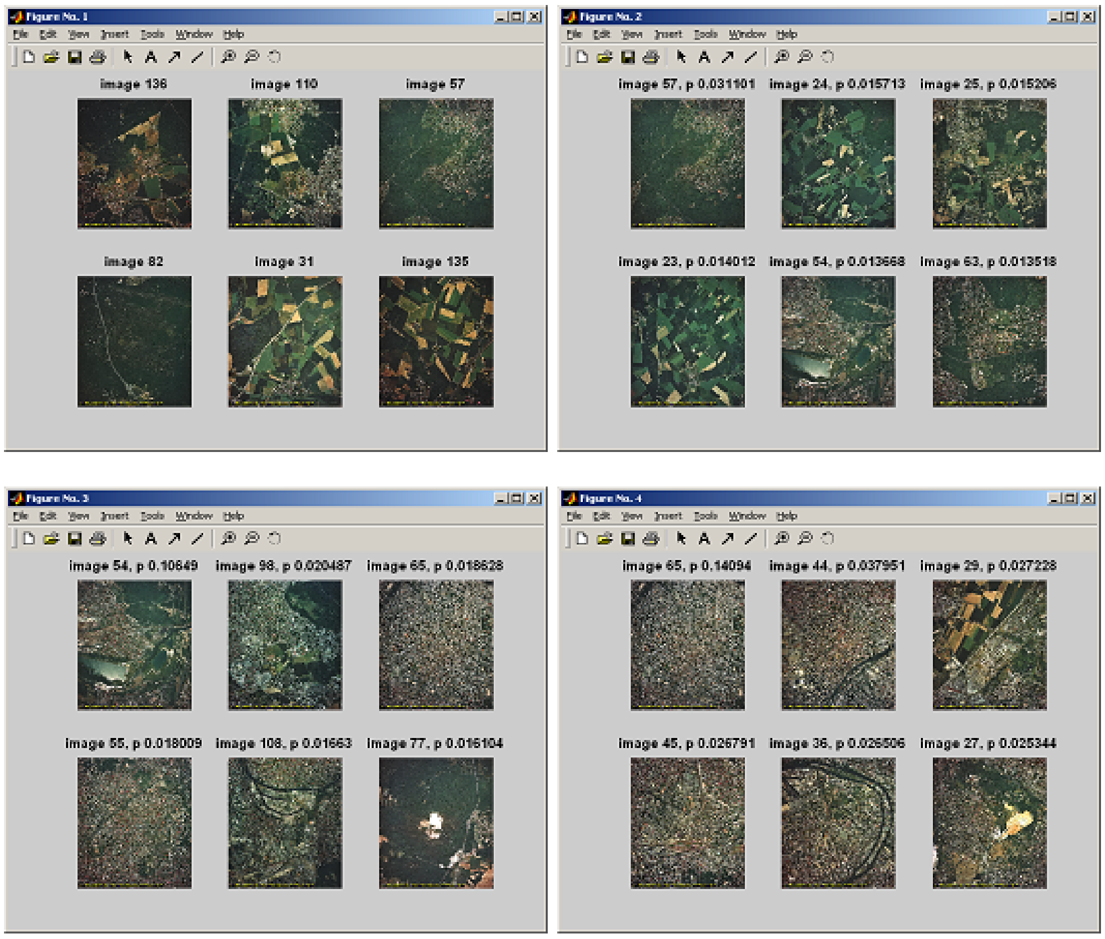


Figure 4.6: Four iterations of a search with σ=1.

4.9.4 Application to the IGN database

In this section we present a demonstration of the simple form (without hidden annotation) of the current PicHunter as appears in Cox et al. (2000). We will consider two characteristic examples searching this particular database. In the first example the user wants to find images with dominant urban area from a database of aerial images. Figure 4.8 illustrates this example. At the first iteration (top left MATLAB figure window) six random images are displayed. Given that the user is looking for an image with a lot of urban area, the search starts by selecting image 34. At the second

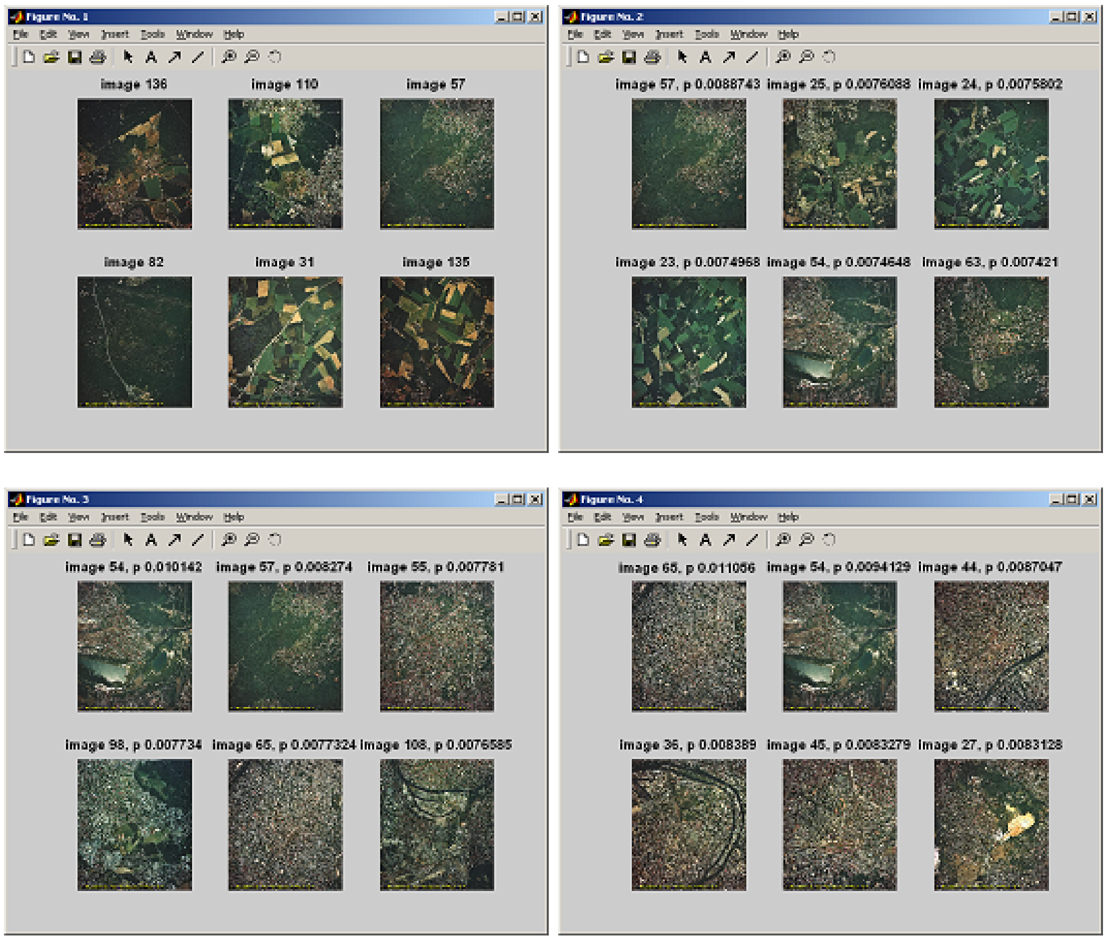


Figure 4.7: Four iterations of a search with σ=10.

iteration (top right MATLAB figure window), images with dominant urban area start to appear. Image 45 was selected at the second iteration and image 65 was selected at the third iteration.

In the second example (Figure 4.9), the user wants to find an image full of fields. Images 67, 50, 70 were selected at the first, second and third iteration respectively.

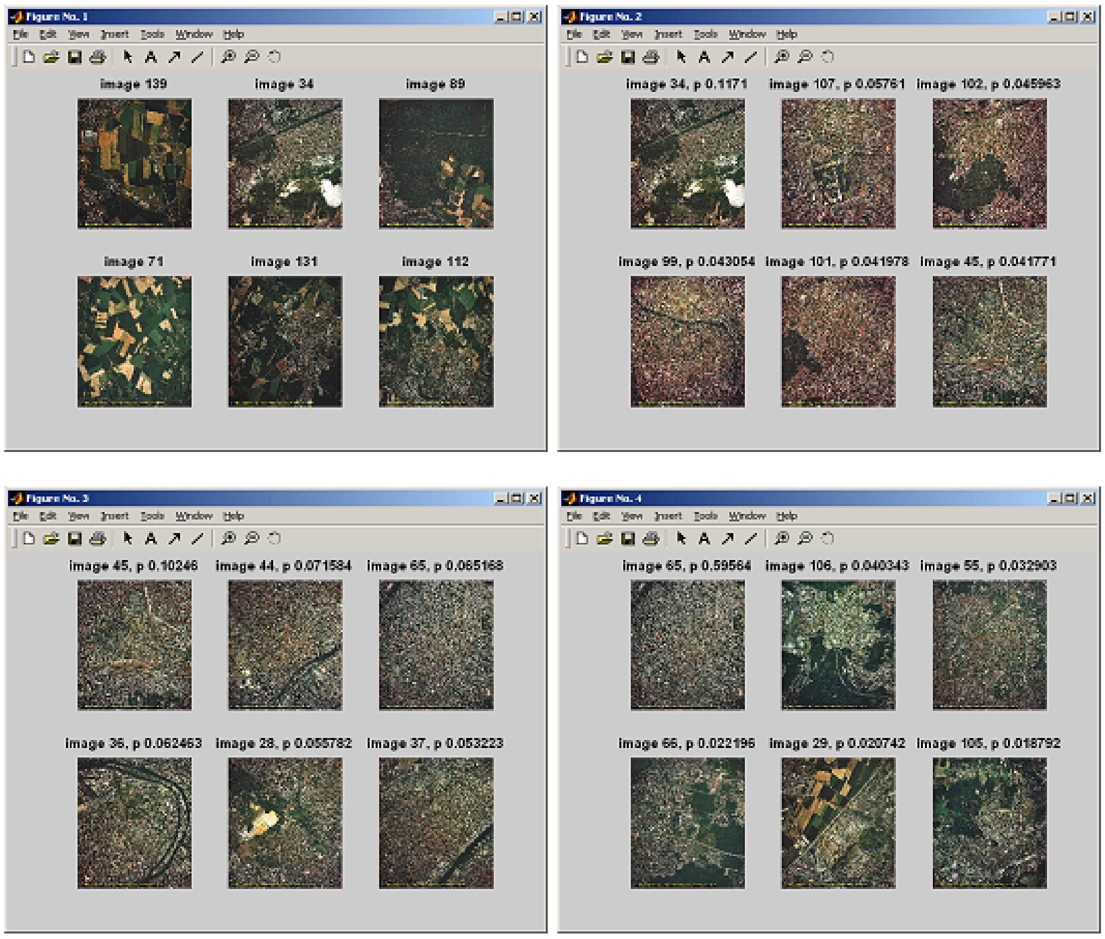


Figure 4.8: Searching for an urban satellite image (σ=0.5).

4.9.5 Application to the BAL database

In this subsection, we show a retrieval example using the BAL image database. Because of the bigger size of this particular database, the display set now is N_D=9 images per iteration. Figure 4.10 illustrates an example where a user looks for a portrait image. Images 574, 1029, 1066 were selected for the first, second and third iteration respectively. As we can see the system tries to retrieve images similar to the user's action. But we also should note that not all of them are successful. As we have already described in Section 4.6, the semantic gap implies that there will almost always be

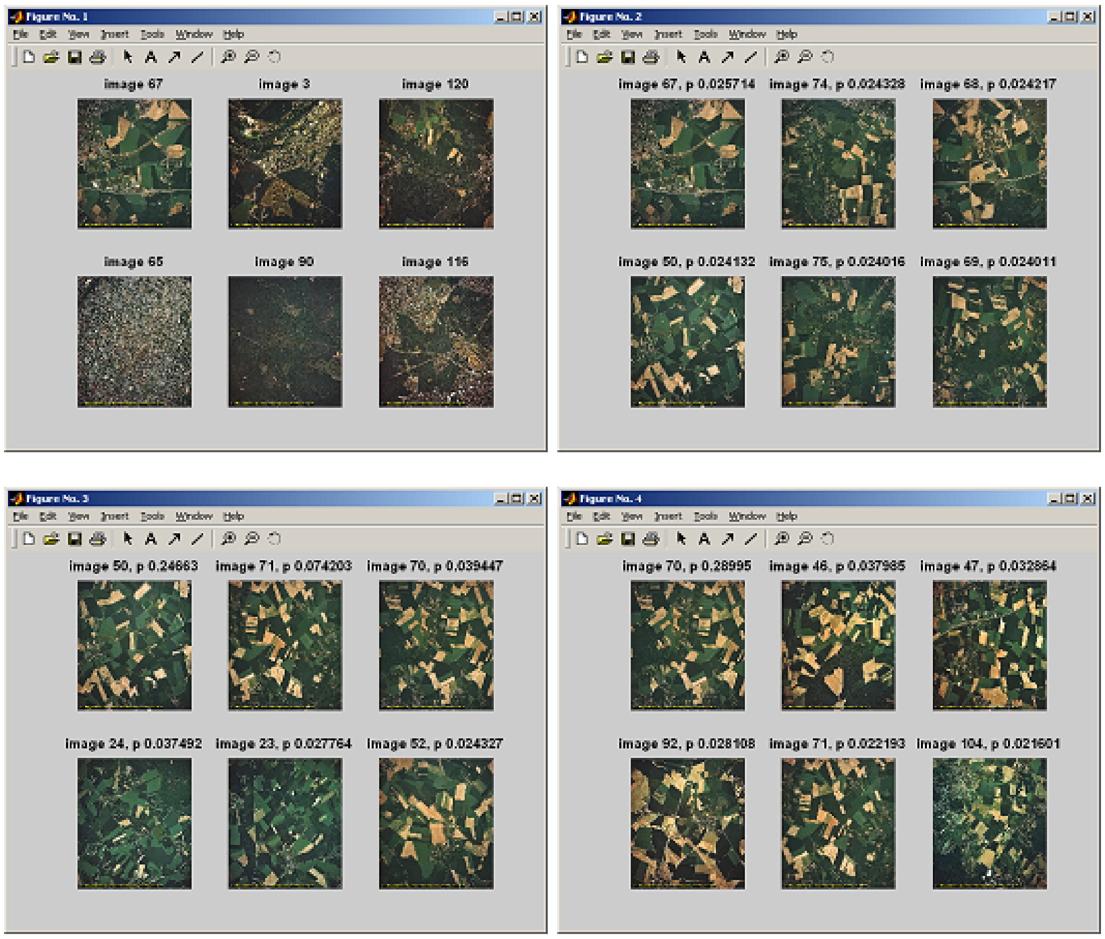


Figure 4.9: Searching for an agriculture image (σ=0.5).

mismatches between the user's query and retrieved images. This problem is more acute in the BAL database because of its considerably more complex semantics.

Another example is illustrated in Figure 4.11 where a user searches for a landscape image. The search was conducted picking images 850, 726, 876 at the first, second and third iteration respectively.

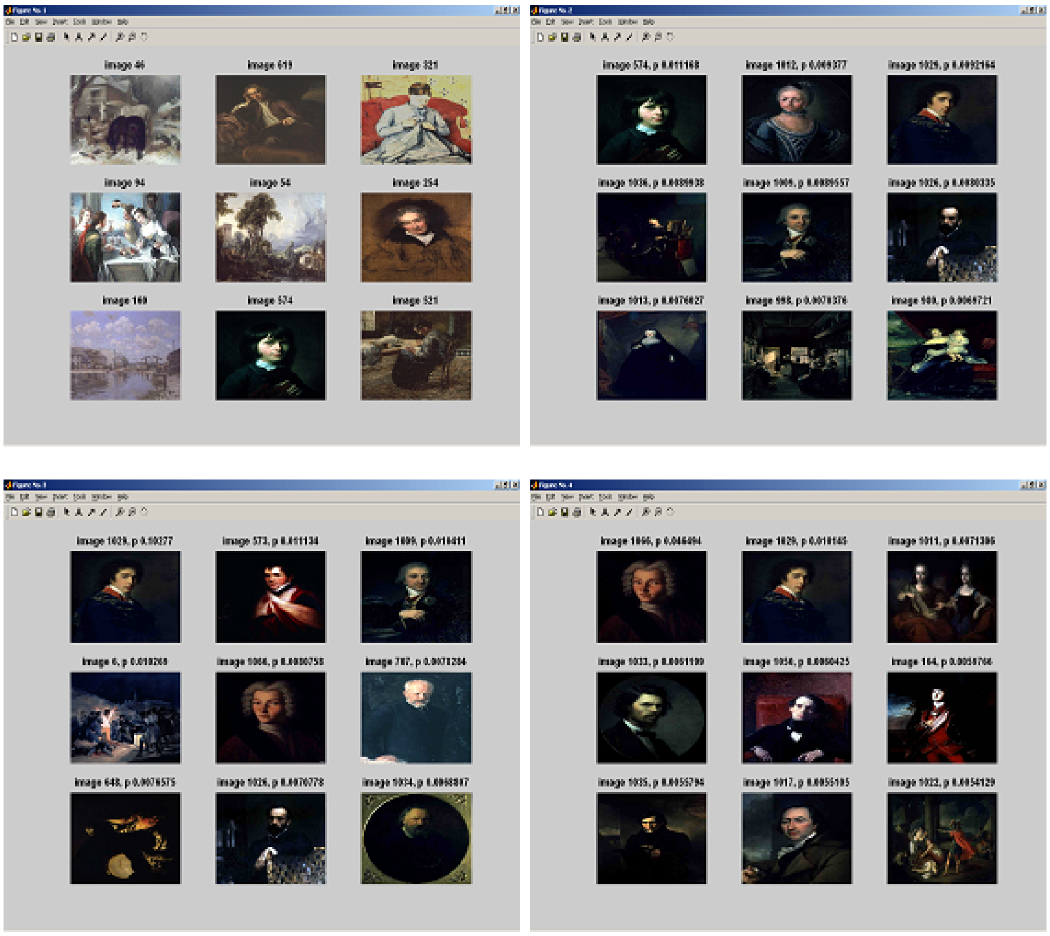


Figure 4.10: Searching for a portrait image (σ=0.5).

4.10 Discussion

PicHunter's new approach is its formulation within the Bayesian framework, which tries to predict the user's desired target image through a posterior probability distribution on the database that represents the probability of it being the target given the relevance feedback history. This distribution is updated based on the user action after each iterative display. Like all CBIR systems, within the model is a rudimentary knowledge of humans' judgments of image similarity, based on empirically derived pictorial features. The image similarity is achieved through a Euclidean distance metric

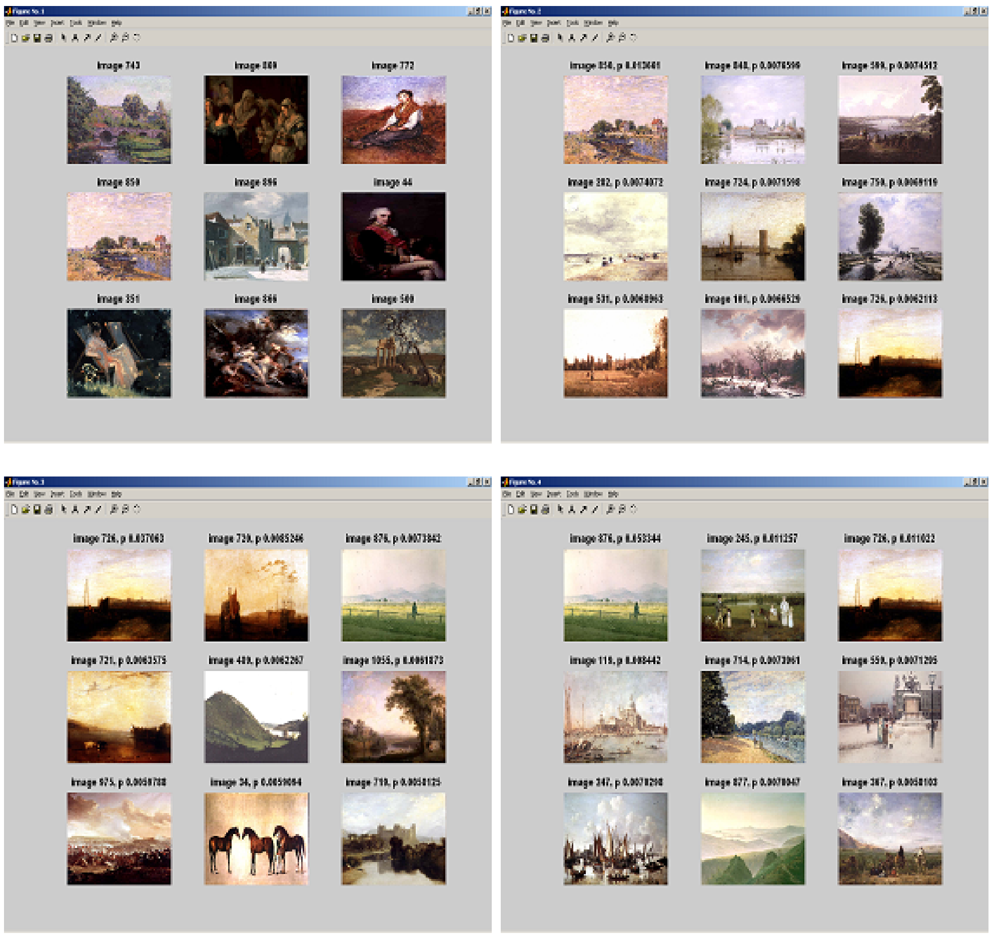


Figure 4.11: Searching for a landscape image (σ=0.5).

function. The third major component, the "display-updating scheme" is concerned with how to select the images for the next iteration's display. The most-probable display updating scheme was used here. This is an obviously reasonable strategy, for the next display choose the N_D images that possess the highest probabilities of being the target; possible ties are broken with random selections. Typically, this updating scheme produces displays whose images belong to a common theme, such as portraits or landscapes, even with the purely pictorial feature user model, somehow exhibiting an ability to extract semantic content. However, the current state of computer vision does not allow semantic information to be easily and automatically extracted from the images but colour has proven to be an image feature with some capability of retrieving images from common semantic categories.

However intriguing this semantic behaviour of PicHunter may be, we should refer to a problem of the system that we experienced. If the user's image target is a picture of a "vase with flowers", we might quickly find a number of similar images, but before we reach the particular "vase with flowers" image that we are looking for, we often find ourselves suddenly presented with a collection of images of blue-green landscapes or of human beings standing in outdoor scenes. These incorrect images will be shown for a few iterations before the system returns to the class of "vase with flowers" images. There are three factors at work here: the particular images are few in the database, so there are no other similar images to be retrieved; the probability distribution which has developed over the multidimensional feature space is multimodal and finally the semantic gap. The multimodality of the distribution means that there are many landscape images and many "vase with flowers" images that are highly probable. The most probable 9 images might easily all come from one group ("vase with flowers" images), and the next 9 from the other (landscapes with human beings), so the system will hop between these two modes. We will propose a user model that will try to reduce the number of modes that develop, thus reducing this disconcerting behaviour. An algorithm for choosing the set of images to display might reduce the damage done by multiple modes. Such an investigation is presented in Chapter 6.

Other extensions could accommodate a more complex display. Such a display could show on the iterative session window which features caused the images to be displayed. Users can benefit by knowing what are Pichunter's current "beliefs" about what features the user is using to choose images, as this will give them an idea of how their choices affect the system. A more complex display could indicate which set of features caused the retrieved images to be displayed.

However, when a user operates Pichunter to search for images that are similar to a prototype image, say a landscape scene, the system quickly produces displays with similar images; in a lax sense, under these conditions, this type of search can be considered as a category search. Pichunter can become a “semantical category-search” engine (for example, looking for sad, romantic or violent images) if the Bayesian scheme is modified to treat sets of images rather than individual images. The challenge for the system would be to discern the commonality of the features that specify a certain category that the user has in mind. The main characteristic of “open-ended browsing” is that users change their goals during the search either gradually or quite abruptly, as a result of having encountered something interesting that they had not even considered at the beginning of the search. Accommodating these changes necessitates a modification of the display updating scheme.

Chapter 5

Extensions of PicHunter system

5.1 Introduction

In this Chapter we propose some extensions based on the current PicHunter model as it appeared in Cox et al. (2000). Those proposals have to deal with inserting some new parameters into the current model. The computation of those new models is not trivial given the strong restrictions that we have on computation time; this is an on-line computation. Methods such as direct Monte Carlo and direct evaluation of a posterior on a grid were tried, with the latter approach found to be better.

5.2 Bayesian Inference on (T, σ)

Previously, in Chapter 4 we computed the probability that database image T_i is the user's desired image, given the observed history and with σ fixed. One option is to define σ deterministically. But choosing manually values for σ we cannot guarantee which is the best value for σ. Given that the user must pick one image from the display

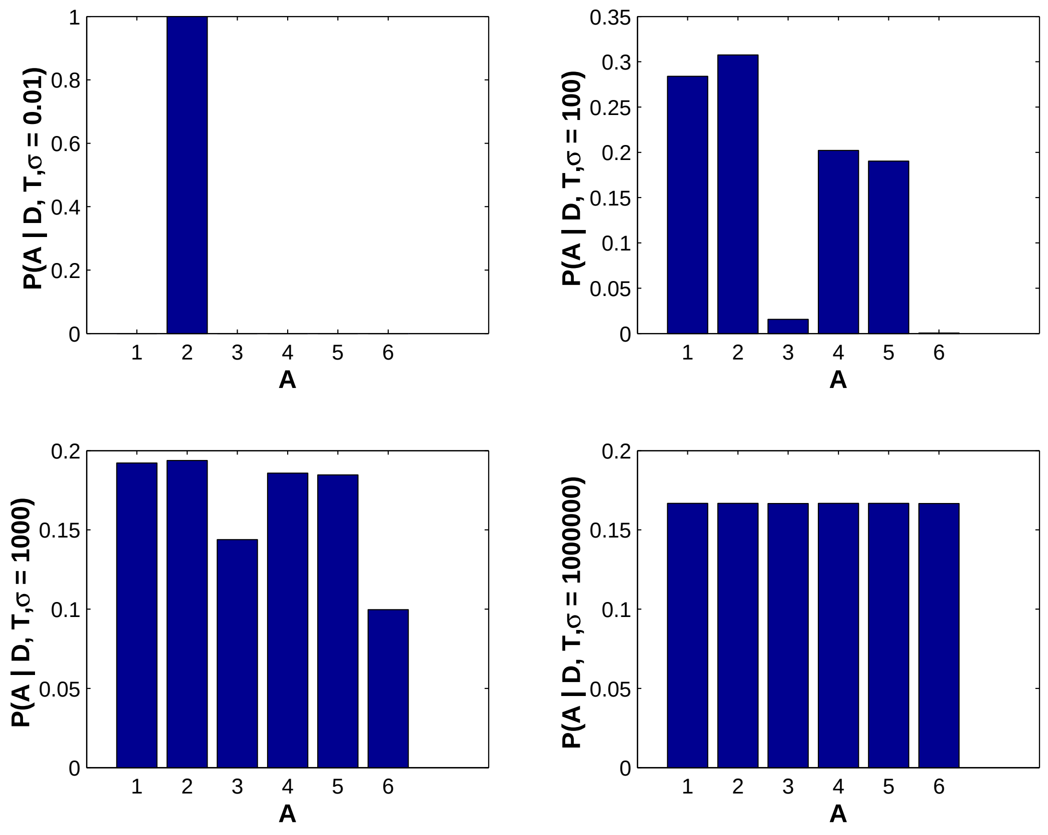


Figure 5.1: Different values of σ produce different likelihoods.

set $D = \{X_1, X_2, ..., X_{N_D}\}$ as his action $A = 1, ..., N_D$ and considering the likelihood

$$P(A = \alpha | X_1, ..., X_{N_D}, T) = \frac{\exp(-d(X_\alpha, T)/\sigma)}{\sum_{i=1}^{N_D} \exp(-d(X_i, T)/\sigma)}, \tag{5.1}$$

we can generalize σ's behaviour by the following simple example.

We set $D = \{X_1, X_2, X_3, X_4, X_5, X_6\}$ and their distances from the target image T to be $d(X_i, T) = \{10, 2, 300, 44, 50, 667\}$. In Figure 5.1 we can see four versions of the likelihood for particular values of σ where it becomes clear that, as $\sigma \to \infty$ then $P(A|D, T, \sigma) = 1/N_D$ which implies that the user is equally likely to pick any image in D. Secondly, as $\sigma \to 0$ then $P(A|D, T, \sigma) = 1$ for the image from the display set D that is closest to target image T and $P(A|D, T, \sigma) = 0$ otherwise. This implies that user will always pick an image which is closest to T. In reverse, when inferring

σ, we have that if our picked images $X_1, X_2, ..., X_{N_D}$ are close to the target image T in feature space, then the posterior of σ will be concentrated on small values. If our picked images are far from the target image in feature space, the posterior of σ will be concentrated on large values. The posterior distribution of σ tells us if the feature space that we use is describing the target image well (small σ) or badly (large σ). It is a measure of how well we are closing the semantic gap; how well the user's semantic notion of similar images is being modelled by distance in feature space. The posterior which we want to calculate now is:

$$\begin{aligned}
P(T = T_i, \sigma | H_t) &= P(T = T_i, \sigma | A_1, D_1, ..., A_t, D_t) \\
&= \frac{P(A_1, ..., A_t | D_1, ..., D_t, T = T_i, \sigma) P(T = T_i) P(\sigma)}{\int_\sigma \sum_{m=1}^{n} P(A_1, ..., A_t | D_1, ..., D_t, T = T_m, \sigma) P(T = T_m) P(\sigma)} \\
&\propto P(A_1, ..., A_t | D_1, ..., D_t, T = T_i, \sigma) P(T = T_i) P(\sigma) \\
&\propto P(A_1, ..., A_t | D_1, ..., D_t, T = T_i, \sigma) \\
&= \prod_{k=1}^{t} \frac{\exp(-d(X_{A_k}, T_i)/\sigma)}{\sum_{j=1}^{N_D} \exp(-d(X_{A_j}, T_i)/\sigma)},
\end{aligned} \tag{5.2}$$

assuming uniform priors on T and σ, the latter in a range (0,1). We picked this range after considerable testing, as a compromise between a sufficiently large range, and being able to compute the posterior at a sufficiently fine set of values. There are several possible approaches to computing Equation 5.2, such as: MCMC (not possible because we require fast computation times), direct evaluation of posterior on a grid (the grid will have to be quite coarse to keep speed satisfactory) and direct Monte Carlo (Importance Sampling).

The next step is to calculate the marginal probabilities of T and σ. The marginal of T,

$$P(T = T_i | H_t) = \int_0^1 P(T = T_i, \sigma | H_t) \; d\sigma$$

shows which images are more likely to be the target. The marginal of σ,

$$P(\sigma|H_t) = \sum_{i=1}^{N} P(T = T_i, \sigma|H_t)$$

indicates the level of confidence that the algorithm has in its ability to model the user's query.

Considering a simulated database of $N = 1000$ images in a two dimensional (uniform) feature space, we investigate the behaviour of σ. We compute exactly the posterior distribution of T and σ as in Equation 5.2, where $P(T)$ and $P(\sigma)$ are prior distributions that we assume are uniform: $P(T) = N^{-1}, i = 1, ..., N$ and $P(\sigma) = 1/1 - 0$, $0 \leq \sigma \leq 1$. We discretise the range of σ into 20 values 0.01, 0.06, ..., 1.

Case 1 - Where σ is getting smaller

The strategy here is to pick continuously the second most probable image from the display set D_t. As regards σ, the same inference process applies; if the selected image is close, relative to others in the display set, to the images with high posterior probability then that is evidence that σ is small. In Figures 5.2, 5.3 we see several iterations. In the top left corner we see the simulated uniform feature space of 1000 images. Green stars indicate displayed images, and the red star indicates the image that was selected as most relevant. In the first iteration the system displays randomly 6 images out of 1000. For this example the six random images are 20, 16, 14, 3, 2 and 1. All the ten iterations of this search are shown in Table 5.1. At the top right corner we see the retrieved images after the user's action. These two graphs are followed by another pair of graphs; the marginal posterior distribution of T (left) and the marginal posterior distribution of σ (right). As the search goes we still remain in the same cluster of images in feature space. After some iterations the posterior distribution of σ starts to be concentrated on small values.

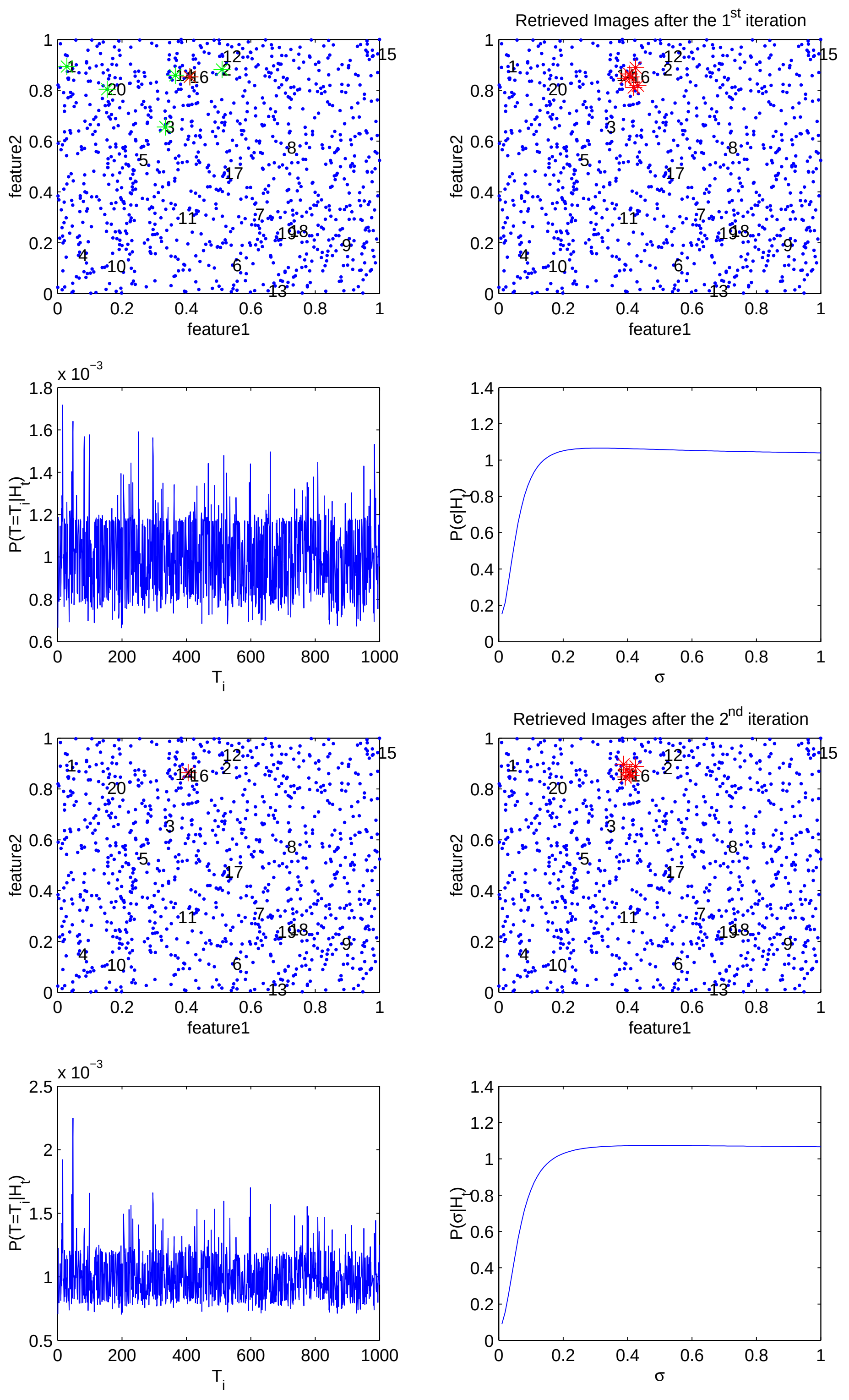


Figure 5.2: Calculation of the exact posterior for a simulated database of 1000 images for the 1st and 2nd iteration. The strategy here is to pick continuously the second most probable image form the display set D_t.

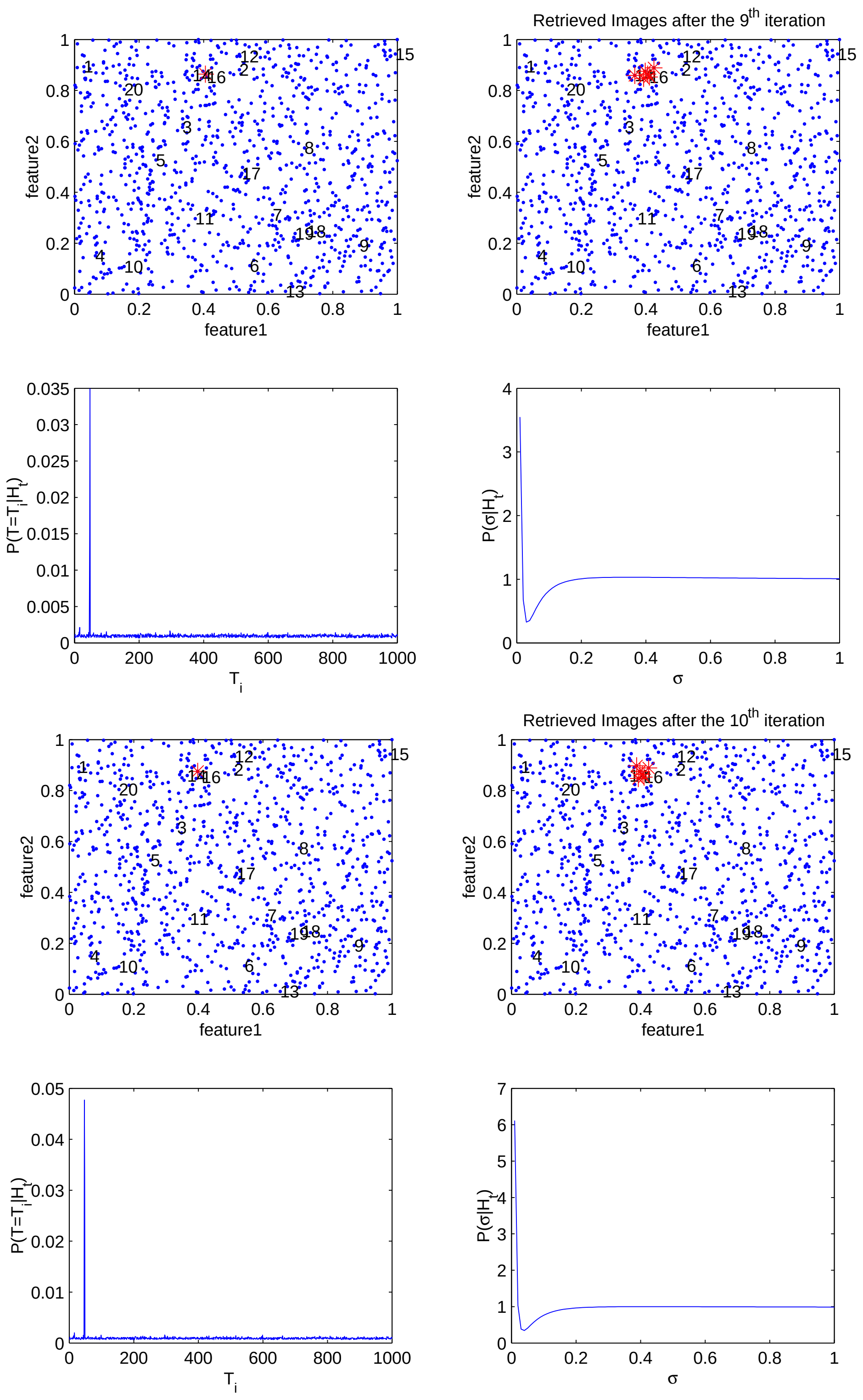


Figure 5.3: Calculation of the exact posterior for a simulated database of 1000 images for the 9^{th} and 10^{th} iteration. The strategy here is to pick continuously the second most probable image form the display set D_t.

ITERATION	QUERY IMAGE	DISPLAYED IMAGES	Mean of σ
Initial	—	20, 16, 14, 3, 2, 1	—
1	16	16, 48, 251, 99, 83, 296	0.52789
2	48	48, 47, 16, 599, 296, 99	0.53613
3	47	47, 48, 16, 44, 599, 99	0.53855
4	48	48, 47, 16, 296, 99, 44	0.5345
5	47	47, 48, 16, 296, 599, 44	0.53605
6	48	48, 47, 16, 296, 99, 599	0.53176
7	47	47, 48, 16, 296, 99, 44	0.53004
8	48	47, 48, 16, 296, 99, 44	0.51152
9	48	48, 47, 16, 296, 99, 14	0.49885
10	47	47, 48, 16, 296, 599, 99	0.4672
11	48	48, 47, 16, 296, 99, 44	0.4237

Table 5.1: Sequence of picked and displayed images for the simulated database for the case that σ is getting smaller.

Case 2 - Where σ is getting bigger

Exactly opposite behaviour of the σ will be obtained when the user picks continuously the 6^{th} lowest probable image form the display set D_t. As regards σ, the same inference process applies; if the selected image is far away, relative to others in the display set, to the images with high posterior probability then that is evidence that σ is large (the uncertainty about σ is high). In Table 5.2 is shown a sequence of picked and retrieved images. In Figures 5.4, 5.5 we see the first two and the last two iterations of a search. We should note that there is a small movement of the cluster in feature space which does not allow the model to learn about user preferences.

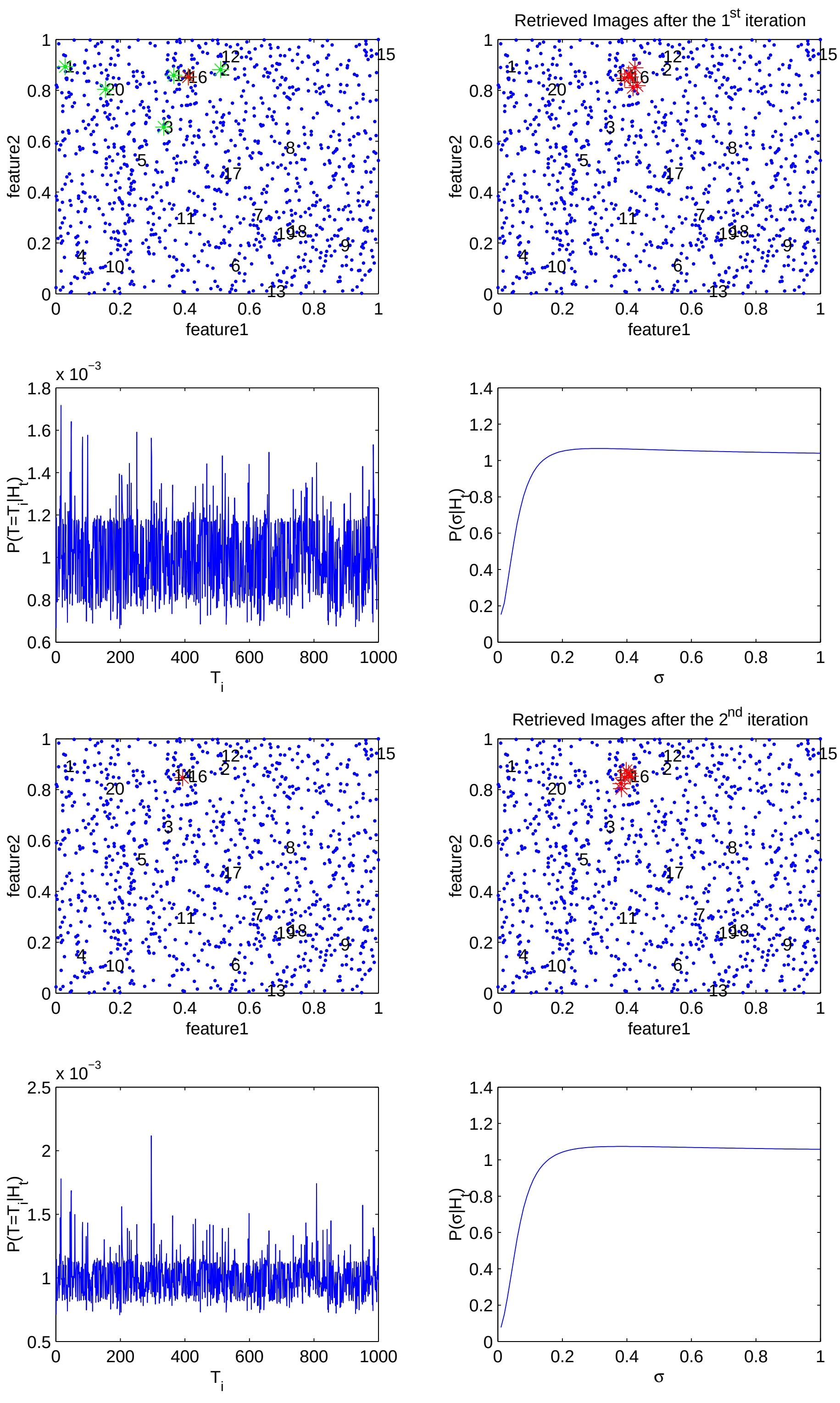


Figure 5.4: Calculation of the exact posterior for a simulated database of 1000 images for the 1st and 2nd iteration. The strategy here is to pick continuously the lowest (6th) most probable image form the display set D_t.

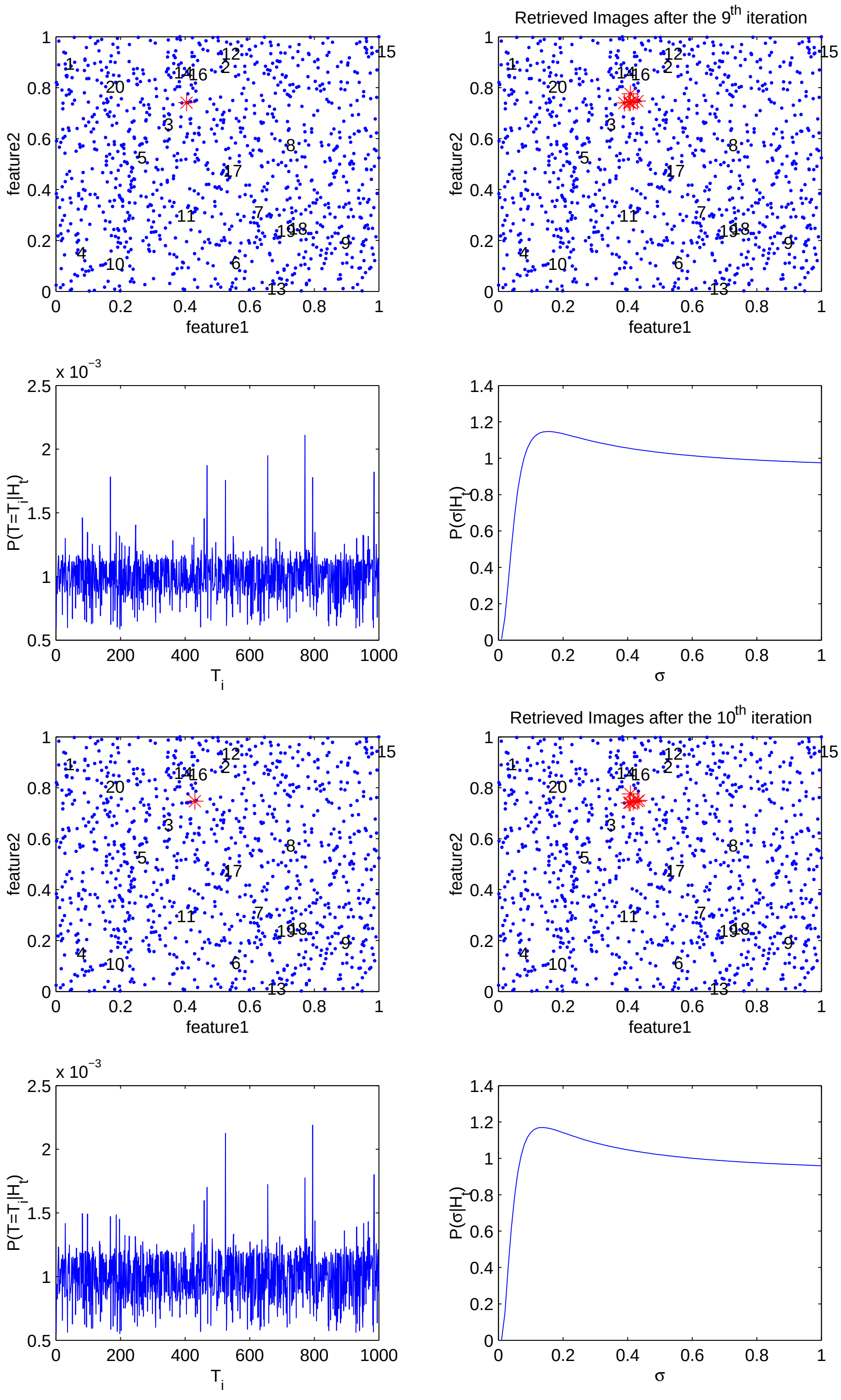


Figure 5.5: Calculation of the exact posterior for a simulated database of 1000 images for the 9th and 10th iteration. The strategy here is to pick continuously the lowest (6th) most probable image form the display set D_t.

ITERATION	QUERY IMAGE	DISPLAYED IMAGES	Mean of σ
Initial	—	20, 16, 14, 3, 2, 1	—
1	16	16, 48, 251, 99, 83, 296	0.52789
2	296	296,16,808,48,47,951	0.5341
3	951	951,204,362,808,841,468	0.5249
4	468	468,985,795,525,656,771	0.51001
5	771	771,656,169,468,985,795	0.51152
6	795	795,525,985,771,656,468	0.50598
7	468	468,82,984,951,795,525	0.51838
8	525	525,468,795,985,197,771	0.50685
9	771	771,656,468,985,169,795	0.51126
10	795	795,525,985,771,656,468	0.50575
11	468	468,795,525,82,984,985	0.51836

Table 5.2: Sequence of picked and displayed images for the simulated database for the case that σ is staying more uncertain.

5.2.1 Repeating the Simulated Example using Importance Sampling

For larger databases with more features, exact computation of the posterior may not be possible, and we may have to resort to other approaches. In this Subsection we evaluate the performance of importance sampling by the simulated example of Section 5.2. Thus we can compare the exact and Monte Carlo approaches. The basic steps of the importance sampling algorithm for simulation from a posterior distribution $P(\theta|x)$ are (Tanner, 1996):

a) Sample θ^1, θ^2, ..., θ^K from $P(\theta)$

b) Calculate

$$p^{(k)} = \frac{P(x|\theta = \theta^{(k)})}{\sum_{j=1}^{K} P(x|\theta = \theta^{(j)})} \quad , \quad 0 \leq p^{(k)} \leq 1, \quad \sum_k p^{(k)} = 1.$$

c) Randomly pick one $\theta^{(k)}$ using probabilities $p^{(1)}, p^{(2)}, ..., p^{(k)}$ using inverse transform for discrete distributions.

Then the chosen $\theta^{(k)}$ is a sample from $P(\theta|x)$. We should note that if K is not large enough, none of the K prior samples will lie in the region of high posterior probability, leading to a biased sample. Usually this can be detected because in this situation one of the $p^{(k)}$ is near 1 and all others are near 0. For this reason we make K as large as possible given the constraints on storage space and computation time.

In our case $\theta = (T, \sigma)$ and $x = H_t$. We repeat step c) M times to get a sample of M values from $P(T, \sigma \mid H_t)$; in this case we repeat $M = 200$ times, to keep the computation time practical. In our case the posterior distribution on σ tended to be quite uniform, and so 200 samples were enough to provide a good sample from T. Note that in most applications that are not as time-constrained as here, many more samples are generated. We denote the samples $T(m), \sigma(m)$, $m = 1, \ldots, M$.

Given these samples from the joint posterior we can draw histograms of them. We expect to see that:

- $P(\sigma|H_t) \approx$ histogram of values of σ from *importance sampling* .
- $P(T = T_i|H_t) \approx$ proportions of samples of T from importance samples that are T_i.

We have two options for computing the marginal posterior of T and σ. The first option is to use the histograms of $T(m)$ and $\sigma(m)$. However, from a sample of size 200 this histogram shows high variance; in particular, in our experience there is considerable uncertainty in the set of 6 most probable images.

The second option for calculating the marginal of T and σ is to use Monte Carlo integration. We have:

$$\begin{aligned} P(T = T_i|H_t) &= \int_0^1 P(T = T_i, \sigma|H_t)\ d\sigma \\ &= \int_0^1 P(T = T_i|\sigma, H_t) \cdot P(\sigma|H_t)\ d\sigma \\ &= \mathrm{E}[P(T = T_i|\sigma, H_t)] \\ &\approx \frac{1}{M}\sum_{m=1}^{M} P(T = T_i|\sigma(m), H_t); \end{aligned} \tag{5.3}$$

The expectation is with respect to the posterior distribution of σ. The $P(T = T_i|\sigma(m), H_t)$ is the full conditional distribution of T and we have:

$$P(T = T_i|\sigma(m), H_t) \propto P(T = T_i, \sigma(m)|H_t); \tag{5.4}$$

Thus the full conditional is proportional to the joint posterior, which is known. So we have:

$$\begin{aligned} P(T = T_i|H_t) &\approx \frac{1}{M}\sum_{m=1}^{M} P(T = T_i|\sigma(m), H_t) \\ &\propto \frac{1}{M}\sum_{m=1}^{M} P(T = T_i, \sigma(m)|H_t). \end{aligned} \tag{5.5}$$

We compute this last function above for $i = 1, ..., N$ for every image in the database and then normalise these probabilities such that they sum to one.

Running the program for 11 iterations, we can see the results in the Table 5.3 comparing with the results from the exact calculation of the posterior of T and σ from Table 5.1. The importance sampling cannot always approximate correctly the marginal posterior distribution of T and the marginal posterior distribution of σ. We need to greatly increase K and M to achieve better results. But, this is not practical in an on-line computation.

	TRUE POSTERIOR			IMPORTANCE SAMPLING		
ITER.	QUERY IMAGE	DISPLAYED IMAGES	Mean of σ	QUERY IMAGE	DISPLAYED IMAGES	Mean of σ
Initial	—	20, 16, 14, 3, 2, 1	—	—	20, 16, 14, 3, 2, 1	—
1	16	16, 48, 251, 99, 83, 296	0.52789	16	16, 48, 296, 251, 99, 47	0.54278
2	48	48, 47, 16, 599, 296, 99	0.53613	48	48, 16, 47, 296, 99, 599	0.502
3	47	47, 48, 16, 44, 599, 99	0.53855	47	48, 47, 16, 44, 599, 296	0.59051
4	48	48, 47, 16, 296, 99, 44	0.5345	48	48, 47, 16, 99, 296, 661	0.54472
5	47	47, 48, 16, 296, 599, 44	0.53605	47	48, 47, 16, 44, 599, 296	0.55103
6	48	48, 47, 16, 296, 99, 599	0.53176	48	48, 47, 16, 99, 296, 44	0.5245
7	47	47, 48, 16, 296, 99, 44	0.53004	47	47, 48, 16, 99, 44, 296	0.52067
8	48	47, 48, 16, 296, 99, 44	0.51152	48	48, 47, 16, 99, 296, 599	0.52024
9	48	48, 47, 16, 296, 99, 14	0.49885	48	48, 47, 16, 296, 99, 44	0.52818
10	47	47, 48, 16, 296, 599, 99	0.4672	47	48, 47, 16, 296, 99, 44	0.53232
11	48	48, 47, 16, 296, 99, 44	0.4237	48	48, 47, 16, 296, 99, 14	0.54903

Table 5.3: Sequence of displayed images and posterior mean of σ for the simulated database, according to exact calculations and the approximation using importance sampling.

5.3 Bayesian Inference on $(T, \sigma, w_{gc}, w_{tx}, w_{sg})$

The second idea we describe in this Chapter is learning about the features in two ways. Currently, we compute the Euclidean distance as:

$$d(X_A, T_i) = \left[\sum_{k=1}^{F} (f_k(X_A) - f_k(T_i))^2 \right]^{1/2}. \tag{5.6}$$

We can redefine the distance as a weighted linear combination of distances over feature subsets, in this case global colour, texture and segmentation subsets, as

$$d(X_A, T_i) = w_{gc} d_{gc}(X_A, T_i) + w_{tx} d_{tx}(X_A, T_i) + w_{sg} d_{sg}(X_A, T_i), \tag{5.7}$$

where $w_{gc}, w_{tx}, w_{sg} \geq 0$, $w_{gc} + w_{tx} + w_{sg} = 1$ and each distance is normalised by $\sqrt{\text{dim}}$ of each feature subset so that its maximum possible value is 1. One way to set the weights is deterministically according to the query. For instance, if the query is an image with green grass and blue sky then more weight can be put to global colour features than the texture features, or if the query is a type of fabric (in interior design) more weight can be assign to textural features.

Because we cannot guarantee which is the best value for the weights, we try to infer them. The unknowns in our system are now: T, σ, w_{gc}, w_{tx} and w_{sg}. Given H_t, our knowledge about these unknowns is given by the posterior distribution:

$$P(T, \sigma, w_{gc}, w_{tx}, w_{sg}|H_t) \propto \left(\prod_{k=1}^{t} P(A_k|D_k, T, \sigma, w_{gc}, w_{tx}, w_{sg}) \right) P(T)P(\sigma)P(w_{gc}, w_{tx}, w_{sg}), \tag{5.8}$$

where

$$P(A_k|D_k, T, \sigma, w_{gc}, w_{tx}, w_{sg}) = \frac{\exp\left(\frac{-\left(w_{gc} \frac{d_{gc}(X_{A_k}, T_i)}{\sqrt{\dim(F_{gc})}} + w_{tx} \frac{d_{tx}(X_{A_k}, T_i)}{\sqrt{\dim(F_{tx})}} + (1 - w_{gc} - w_{tx}) \frac{d_{seg}(X_{A_k}, T_i)}{\sqrt{\dim(F_{seg})}} \right)}{\sigma} \right)}{\sum_{j=1}^{N_D} \exp\left(\frac{-\left(w_{gc} \frac{d_{gc}(X_j, T_i)}{\sqrt{\dim(F_{gc})}} + w_{tx} \frac{d_{tx}(X_j, T_i)}{\sqrt{\dim(F_{tx})}} + (1 - w_{gc} - w_{tx}) \frac{d_{seg}(X_j, T_i)}{\sqrt{\dim(F_{seg})}} \right)}{\sigma} \right)} \tag{5.9}$$

where $P(T)$, $P(\sigma)$, $P(w_{gc}, w_{tx}, w_{sg})$ are prior distributions that we assume are uniform: $P(T = T_i) = N^{-1}$, $i = 1, ..., N$, $P(\sigma) = 1$, $0 \leq \sigma \leq 1$; we discretise the range of σ into 20 values 0.01, 0.06, ..., 1 and $P(w_{gc}, w_{tx}, w_{sg})$ is also continuous uniform on $w_{gc}, w_{tx}, w_{sg} \geq 0$ and $w_{gc} + w_{tx} + w_{sg} = 1$; the constrained uniform on the weights is a particular instance of the Dirichlet distribution, this is a common prior for circumstances like this. We discretise the range of w_{gc}, w_{tx} into 11 values 0, 0.1, 0.2, ..., 1 that satisfy this restriction. We are interested in the marginal posterior distribution of

T:

$$P(T = T_i|H_t)$$
$$= \int_0^1 \int_0^{1-w_{gc}} \int_0^1 P(T, \sigma, w_{gc}, w_{tx}, 1 - w_{gc} - w_{tx}|H_t)\ d\sigma\ dw_{tx}\ dw_{gc}, \quad (5.10)$$

which is used to determine images to be displayed. The marginal posterior distribution of σ is given by:

$$P(\sigma|H_t)$$
$$= \int_0^1 \int_0^{1-w_{TX}} \sum_{i=1}^{N} P(T, \sigma, w_{gc}, w_{tx}, 1 - w_{gc} - w_{tx}|H_t)\ dw_{gc}\ dw_{tx} \quad (5.11)$$

The marginal posterior distributions for the weights are:

$$P(w_{gc}|H_t) = \int_0^{1-w_{gc}} \int_0^1 \sum_{i=1}^{N} P(T, \sigma, w_{gc}, w_{tx}, 1 - w_{gc} - w_{tx}|H_t)\ d\sigma\ dw_{tx}, \quad (5.12)$$

$$P(w_{tx}|H_t) = \int_0^{1-w_{tx}} \int_0^1 \sum_{i=1}^{N} P(T, \sigma, w_{gc}, w_{tx}, 1 - w_{gc} - w_{tx}|H_t)\ d\sigma\ dw_{gc}, \quad (5.13)$$

$$P(w_{sg}|H_t) = \int_0^{1-w_{sg}} \int_0^1 \sum_{i=1}^{N} P(T, \sigma, w_{gc}, 1 - w_{gc} - w_{sg}, w_{sg}|H_t)\ d\sigma\ dw_{gc}. \quad (5.14)$$

Given that importance sampling is not accurate enough, we compute the posterior $P(T, \sigma, w_{gc}, w_{tx}, w_{sg}|H_t)$ over a coarse grid of values for $\sigma, w_{gc}, w_{tx}, w_{sg}$. In Figure 5.6, we can see the grid for the weights over which the posterior is calculated. We discretise the range of w_{gc} and w_{tx} into 11 values: $w_{gc} = 0, 0.1, ...1$, $w_{tx} = 0, 0.1, ..., 1$ such that $w_{gc} + w_{tx} \leq 1$, see Figure 5.6. The σ has also been discretised into 20 equally spaced values, between 0.01 and 1. So, we can compute the posterior exactly at all combinations of $(T, \sigma, w_{gc}, w_{tx}, w_{sg})$ on the discrete grid.

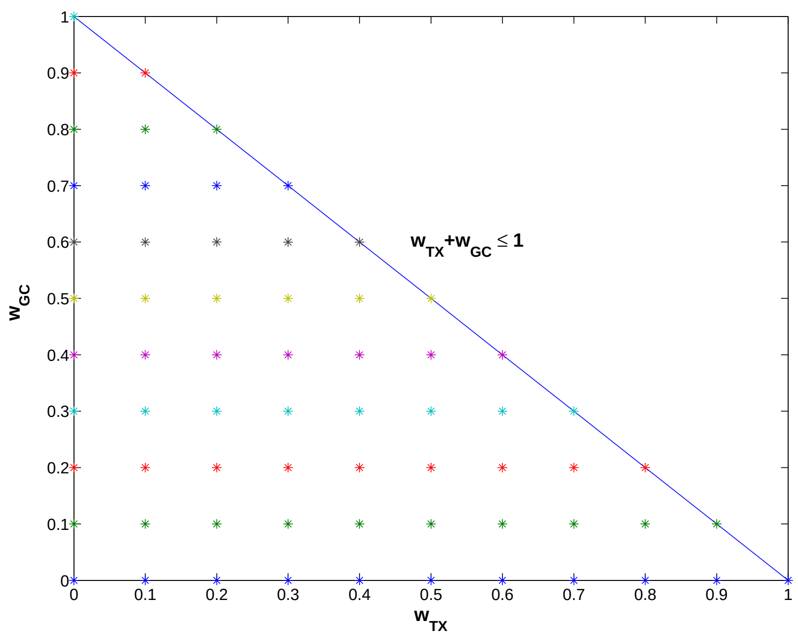


Figure 5.6: Dicretisation of w_{gc}, w_{tx}.

However, some problems emerged using this model. The first is that the computational cost is still high. Computation time per iteration is still over 1 minute for the BAL database of 1066 images, which is not acceptable for a CBIR system, either in development or commercially. So there is a need for other faster models that still allow us to select feature subsets. The second problem is that using this model there are no big changes in weights; the data do not tell us much about them. So there is also a need to build a model that finds the data more informative. Figure 5.7, 5.8 show some iterations of this model applied to the BAL database where we can see that there is no big change in the posterior mean of the weights after several iterations. This was our experience over many different runs of the system. At the first iteration six images from the BAL database are being displayed randomly. Images 475, 868, 383, 316, 68, 358, 341, 383, 868, 383, 868 are picked. In every figure window, the marginal posterior of σ (bottom-left corner of each figure window) and the mean value of each weight (bottom centre of each figure window) are also displayed.

5.4 Bayesian Inference on (T, σ, F)

Because the model of Section 5.3 is quite computationally expensive (the above Equations 5.8 to 5.14 must be computed on-line) another faster model is proposed here. In this case we have 3 sets of features global colour, texture and segmentation features, so $F \in GC, TX, SG$. The idea is that users may search according to different feature types, which we should learn about through inferring F. The unknowns are now T, σ and F. Given H_t, our knowledge about these unknowns is given by the posterior distribution:

$$P(T, \sigma, F|H_t) \propto \left(\prod_{k=1}^{t} P(A_k|D_k, T, \sigma, F)\right) P(T)P(\sigma)P(F), \tag{5.15}$$

where $P(T)$, $P(\sigma)$ and $P(F)$ are prior distributions that we assume are uniform: $P(T = T_i) = n^{-1}, i = 1, ..., n$, $P(\sigma) = 1, 0 \leq \sigma \leq 1$ and $P(F) = 1/3$, $F \in \{GC, TX, SG\}$.

This learning algorithm is based around the model for the probability of which image a user picks from D_k given the target image T:

$$P(A_k|D_k, T, \sigma, F) = \frac{\exp(-d_F(A_k, T)/\sigma)}{\sum_{T_j \in D_k} \exp(-d_F(T_j, T)/\sigma)}, \; A_k \in D_k, \tag{5.16}$$

where σ is a precision parameter and d_F is a normalised distance measure in the set of image features F.

The unknowns are T, σ and F. Given H_t, our knowledge about these unknowns is given by the posterior distribution:

$$P(T, \sigma, F|H_t) \propto \left(\prod_{k=1}^{t} P(A_k|D_k, T, \sigma, F)\right) P(T)P(\sigma)P(F), \tag{5.17}$$

where $P(T)$, $P(\sigma)$ and $P(F)$ are prior distributions that we again assume are uniform: $P(T = T_i) = N^{-1}$, $i = 1, ..., N$, $P(\sigma) = 1$, $0 \leq \sigma \leq 1$ and $P(F) = 1/3$, $F \in$

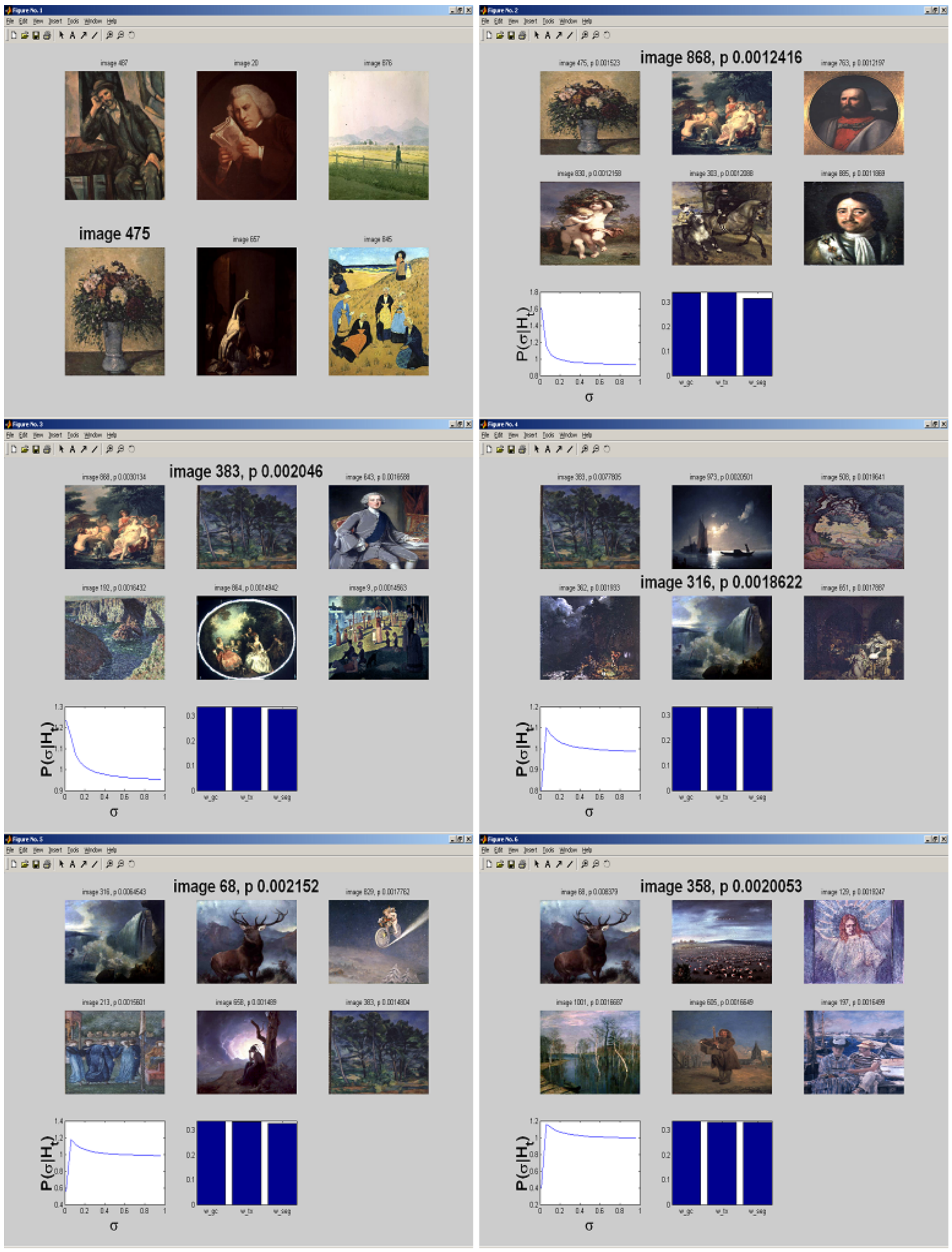


Figure 5.7: The first six iterations of a search using the $P(T, \sigma, w_{gc}, w_{tx}, w_{sg}|H_t)$ model. Selected image is always that in the bold fonts.

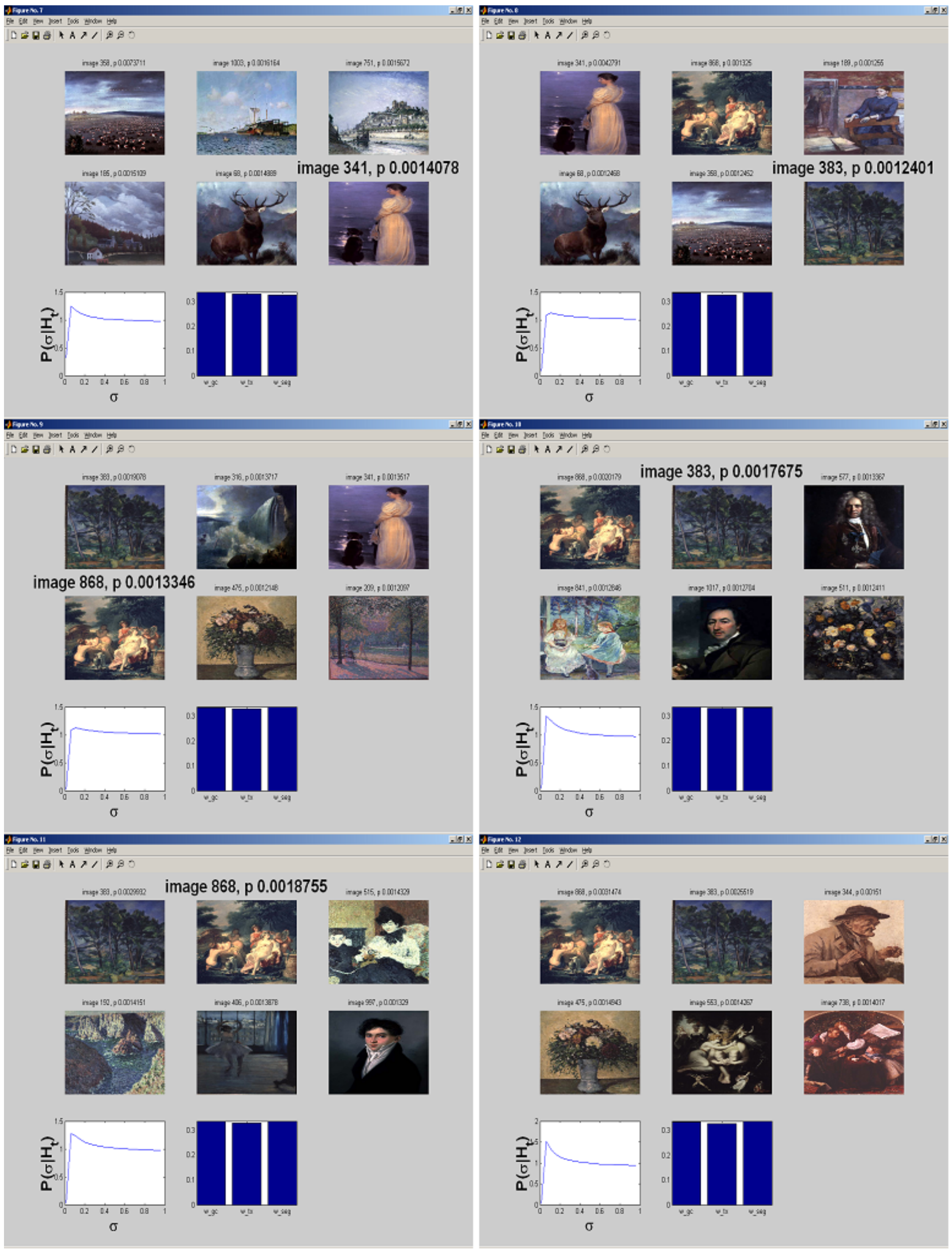


Figure 5.8: The next six iterations of a search using the $P(T, \sigma, w_{gc}, w_{tx}, w_{sg}|H_t)$ model.

$\{GC, TX, SG\}$. We discretise the range of σ into 20 values 0.01, 0.06, 0.11, ..., 0.96 from which we can compute this posterior exactly at all combinations of (T, F, σ) without the need for any approximations. The marginal posterior distribution of T is then:

$$P(T = T_i|H_t) = \int_0^1 \sum_{F \in \{GC,TX,SG\}} P(T, \sigma, F|H_t)\ d\sigma, \quad i = 1, ...n, \tag{5.18}$$

which can also be computed exactly using the discretisation of σ. The marginal of σ is

$$P(\sigma|H_t) = \sum_{i=1}^{N} \sum_{F \in \{GC,TX,SG\}} P(T, \sigma, F|H_t) \tag{5.19}$$

and the marginal of F is

$$P(F|H_t) = \int_0^1 \sum_{i=1}^{N} P(T, \sigma, F|H_t)\ d\sigma. \tag{5.20}$$

This model has two advantages compared to the one that is described in Section 5.3. The first is that it is quite fast, giving retrieval results for the BAL database in less than 15 seconds, which is reasonable for a CBIR system. The second characteristic of this model is that seems to learn more at each iteration.

5.4.1 Implementation with a small Simulated Database

First, we consider a small simulated example of 100 images with 3 features, which we label GC, TX and SEG. Working with this example help us to see more clearly about how the weights of each feature subset behave. We consider the following three cases.

The Global Colour weight becomes large

Our intuition is that when we consistently pick an image with a high value of its global colour feature, then the weight of the global colour feature will be high compared to the others. Figures 5.9 to 5.11 show several iterations where on the left side images that are candidates for picking are displayed as the three possible pairs of features. At the middle column the retrieved images, after user's action, are displayed as three scatter plots for the possible pairs of features. At the right side the marginal posterior distribution of F is shown. Images 21, 10, 38 are picked for the first, second and third iteration respectively. For A_1, A_2, A_3 we have consistently picked high global colour among D_1, D_2, D_3 respectively, but have not consistently picked high texture and segmentation among D_1, D_2, D_3. This implies that the global colour weight becomes large. It is the only feature that is being used consistently by the user.

The Texture weight becomes large

When we consistently pick an image with high values of the textural feature, while not being consistent with our choice of other features, then the weight of the texture feature will be high compared to the others. Figures 5.12 to 5.14 show several iterations where images 2, 27, 73 are consistently picked having high texture.

The Segmentation weight becomes large

Finally, when we consistently pick an image with high value of the segmentation feature, then the weight of the segmentation feature will be high compared to the others. Figures 5.15 to 5.17 show several iterations where images 84, 22, 47 are consistently picked having high value of the segmentation feature.

5.4.2 Implementation with the BAL Database

Figures 5.18 and 5.19 show a search where σ and weights change. The image with the bold title indicates that it has been picked. We set the display set to be six images for ease of display. At the bottom of each display are given the marginal of σ (bottom-left)

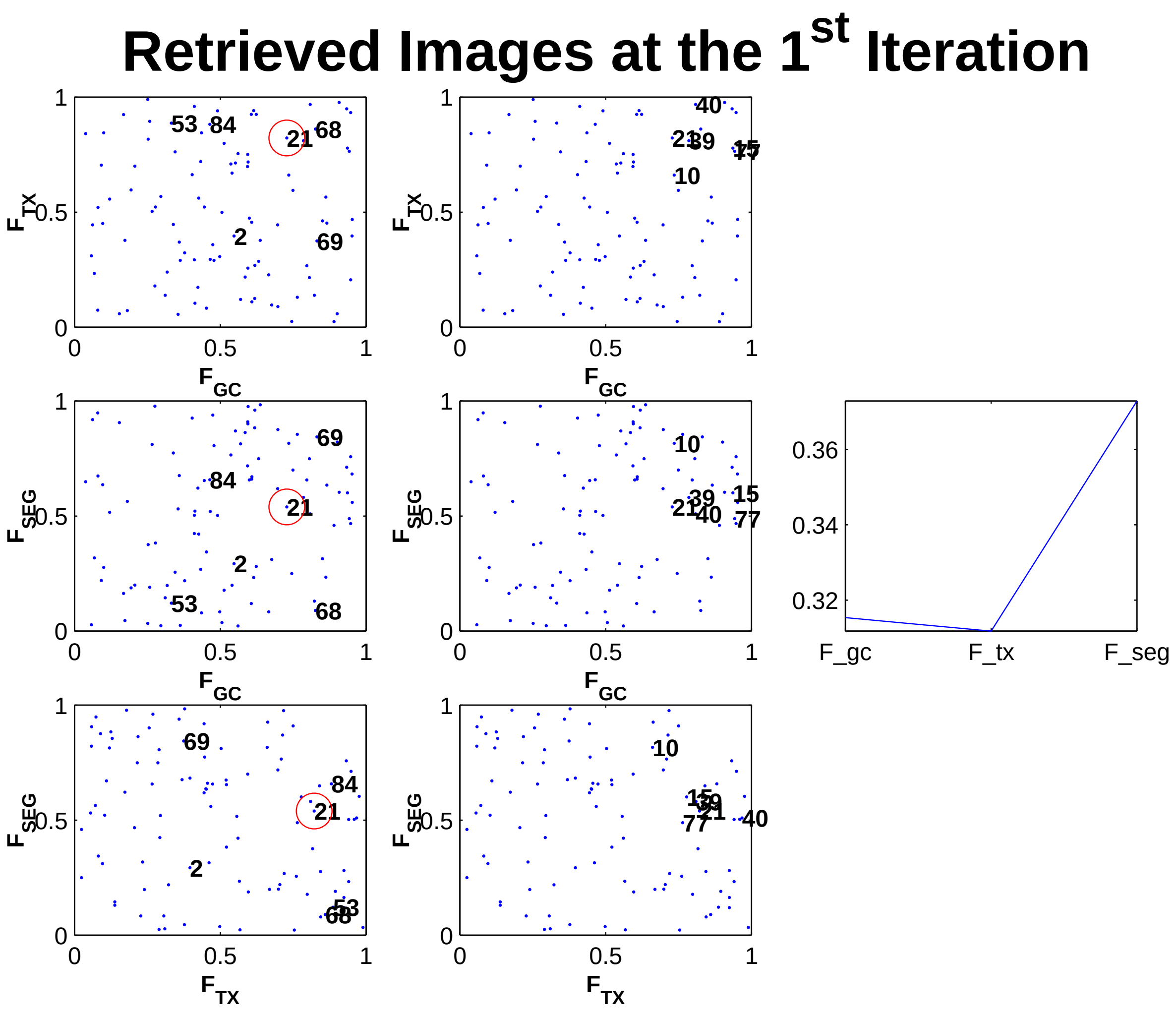


Figure 5.9: The first iteration.

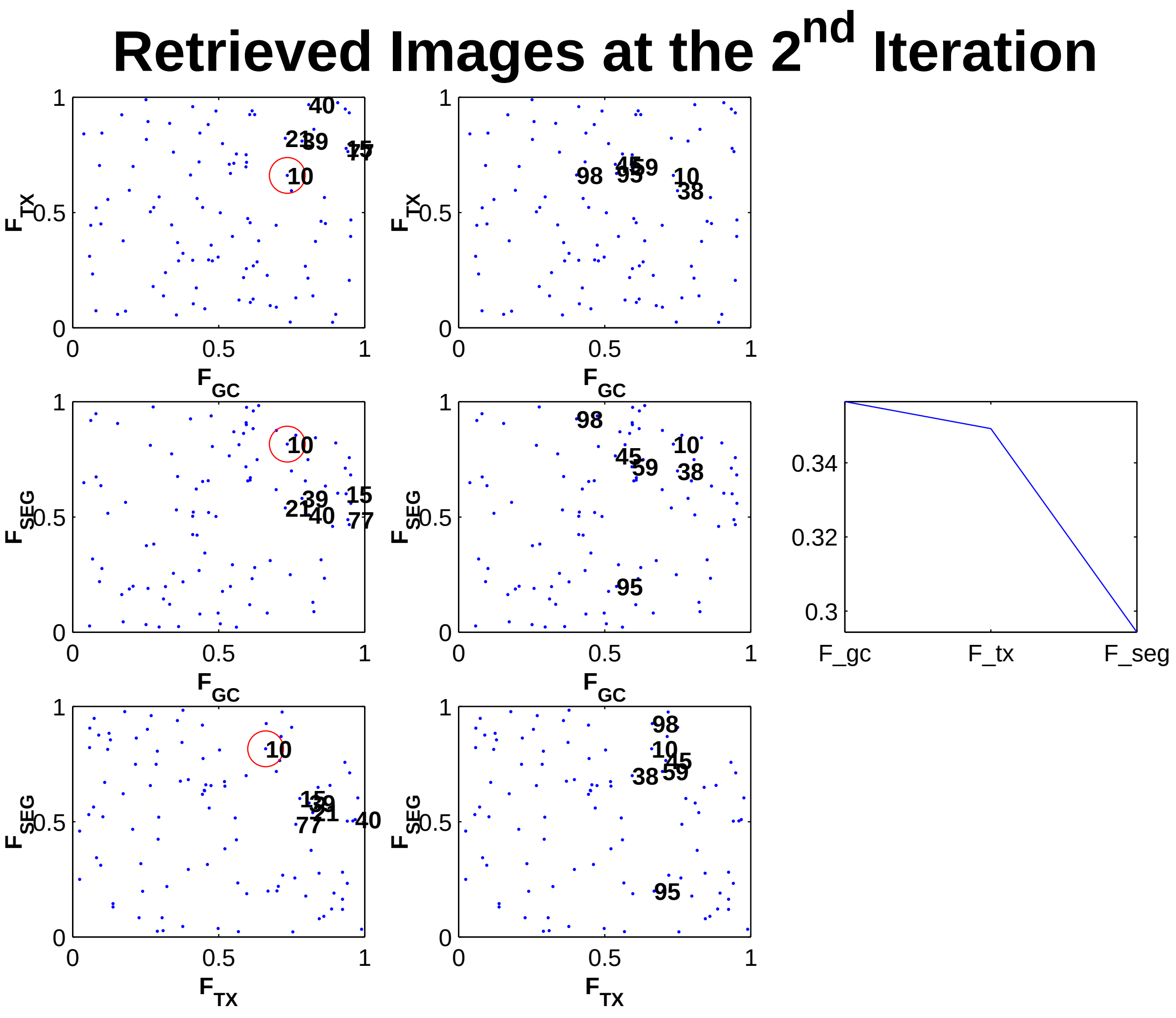


Figure 5.10: The second iteration.

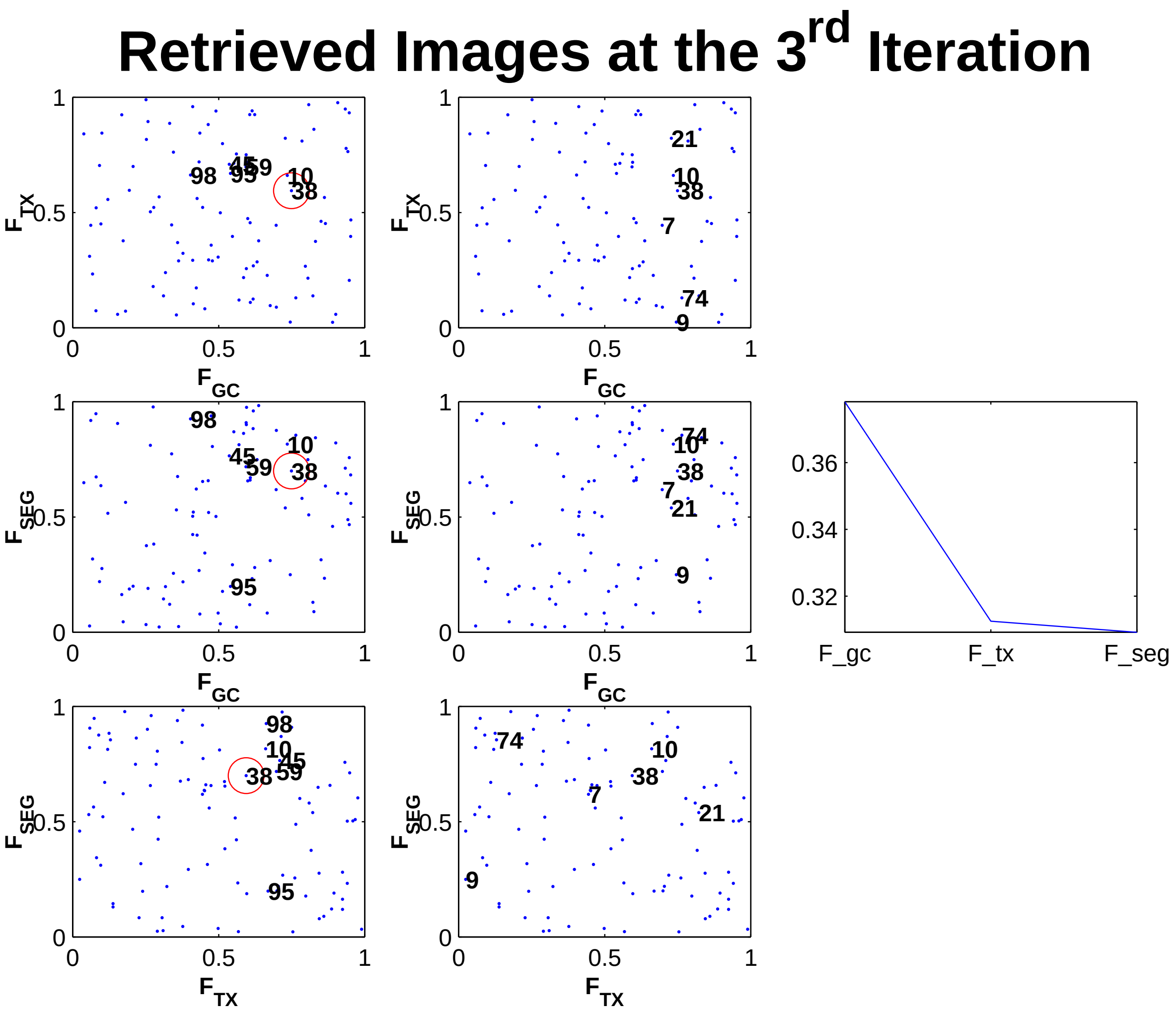


Figure 5.11: The third iteration.

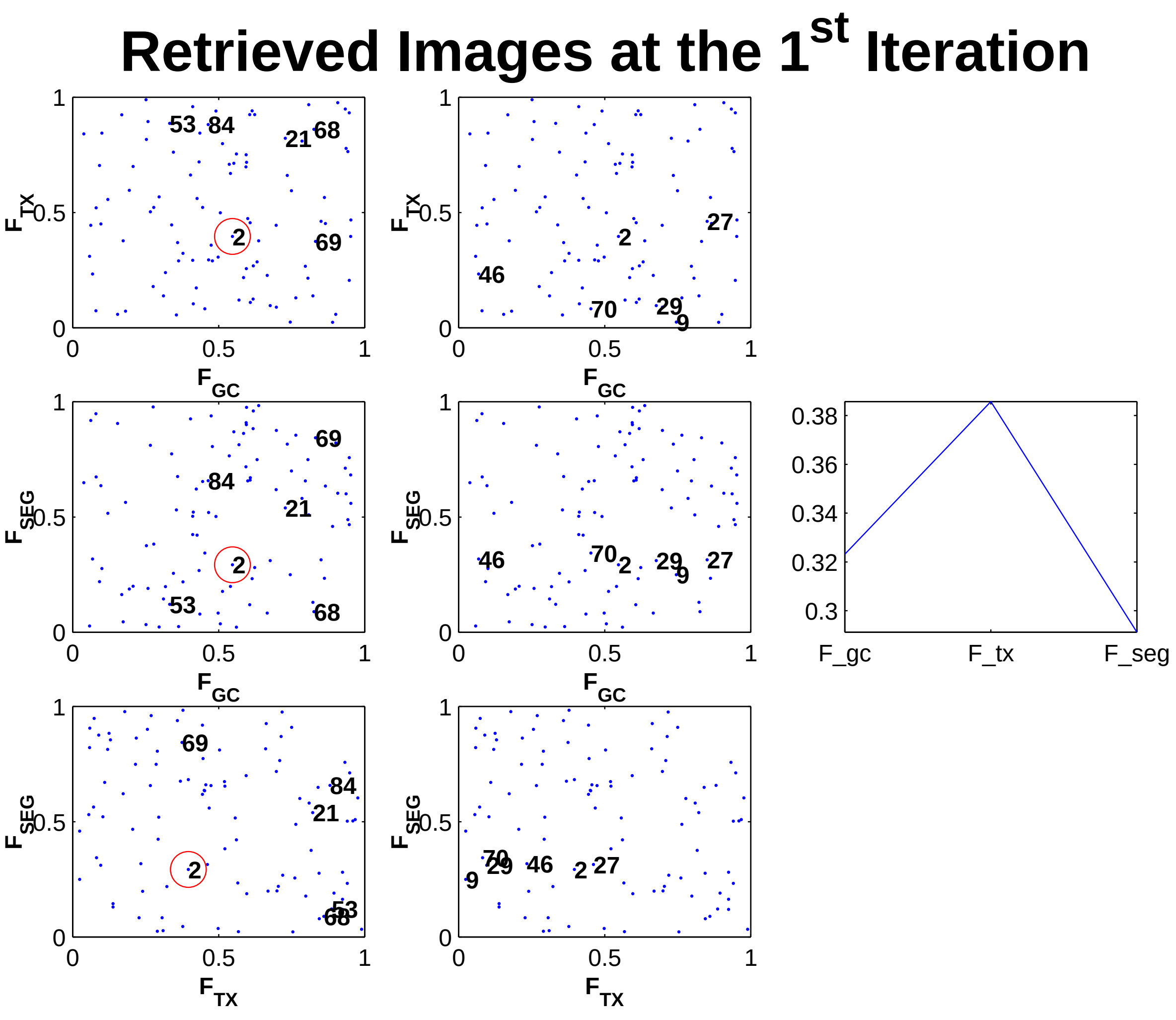


Figure 5.12: The first iteration.

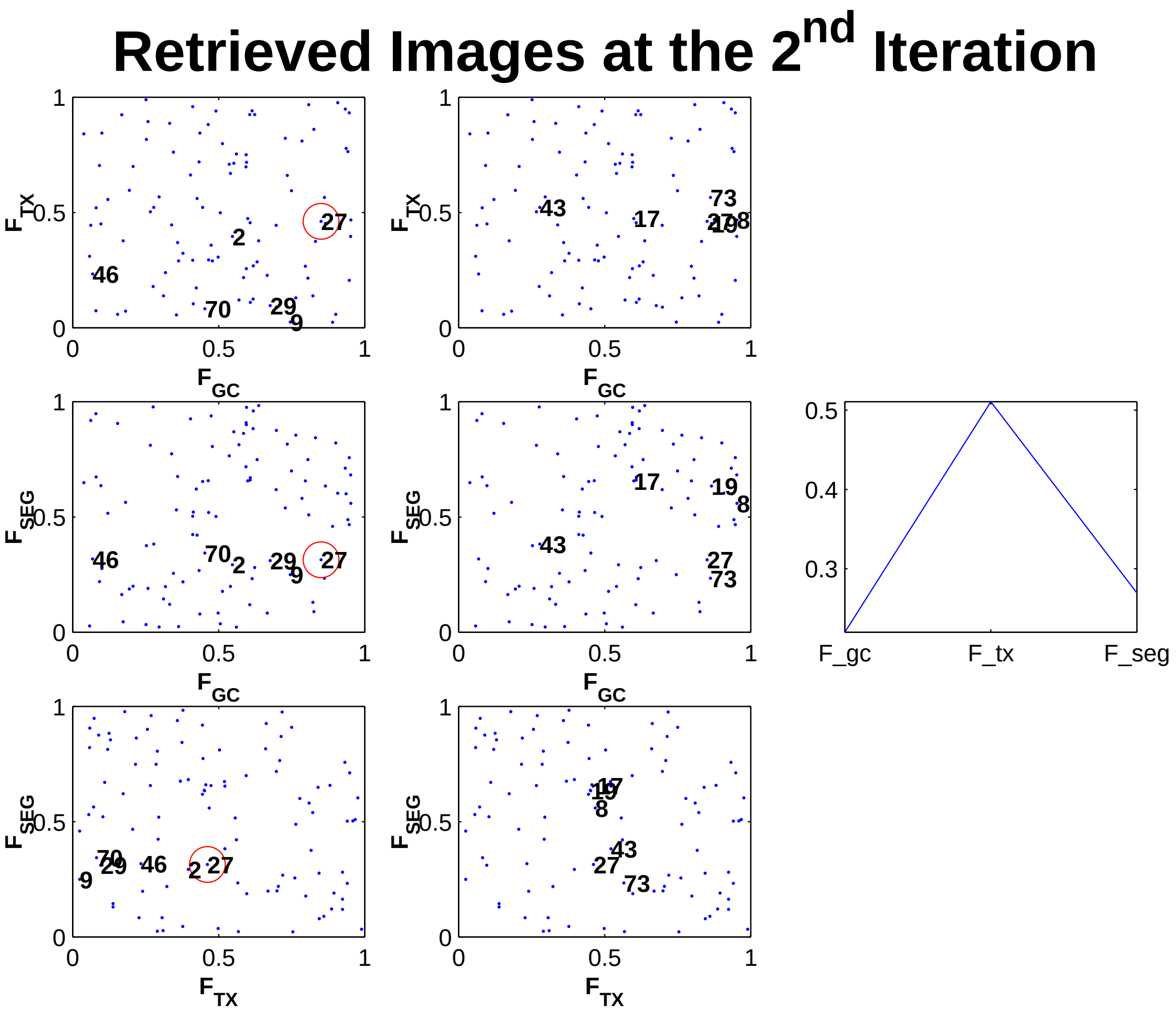


Figure 5.13: The second iteration.

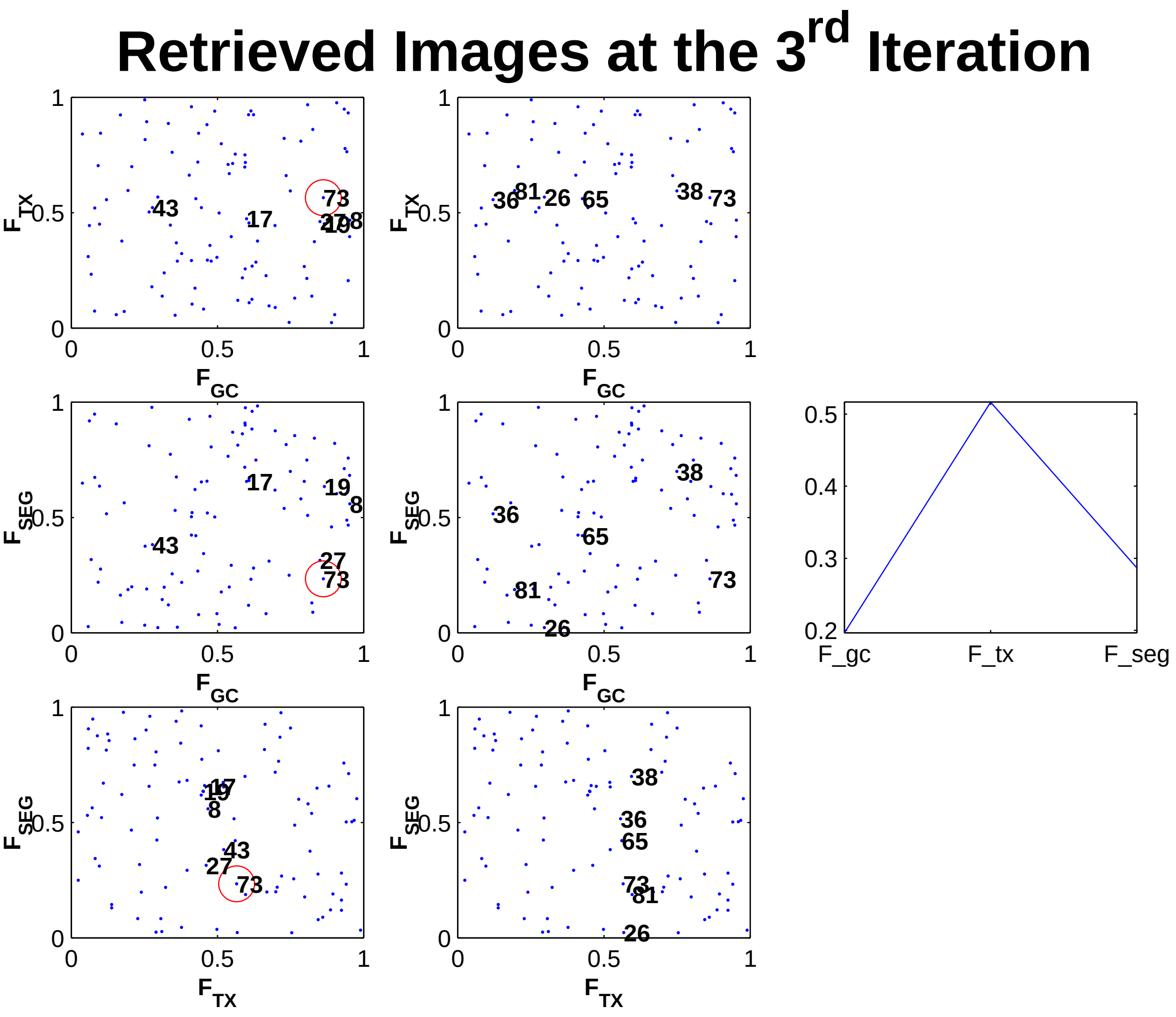


Figure 5.14: The third iteration.

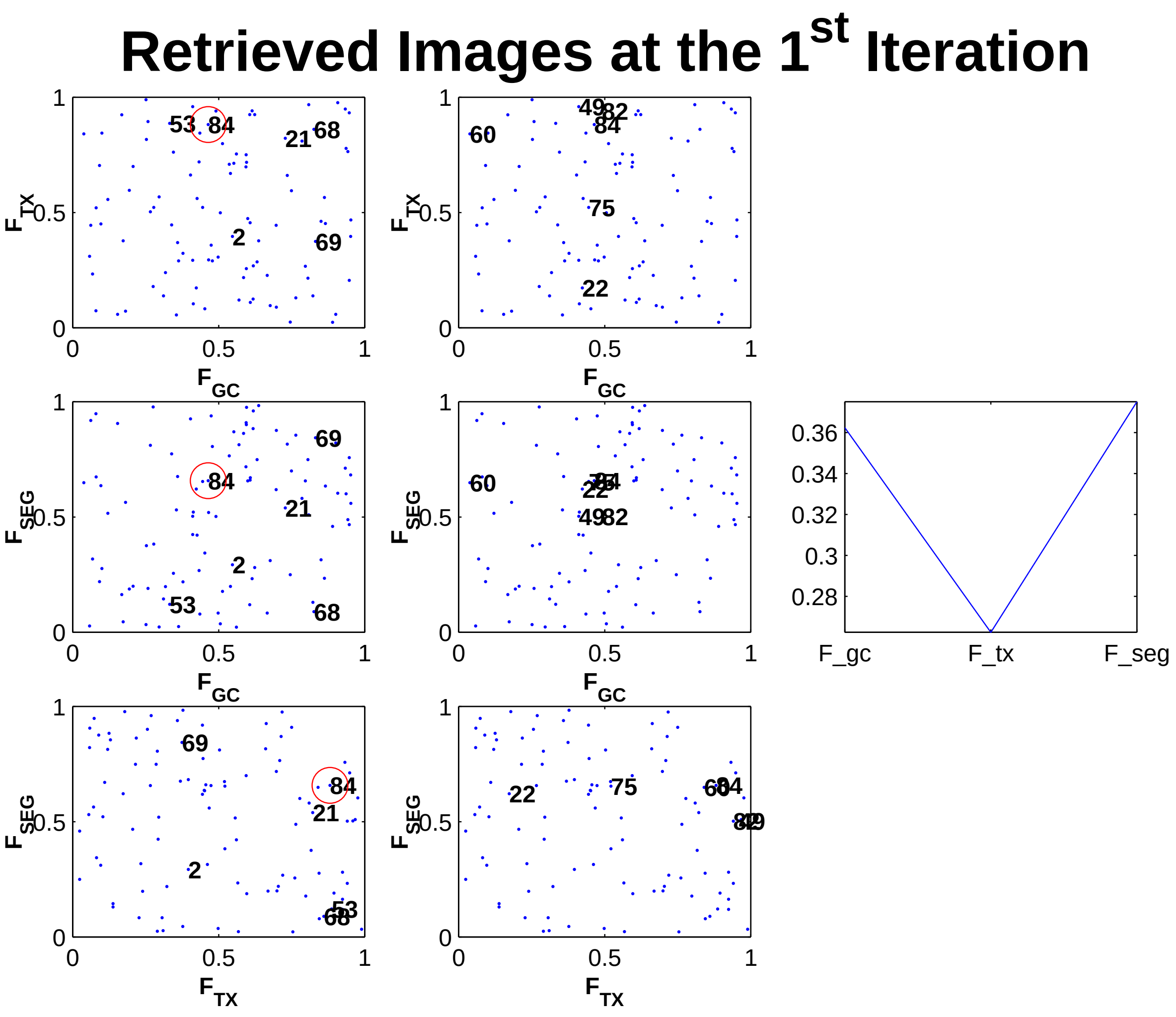


Figure 5.15: The first iteration.

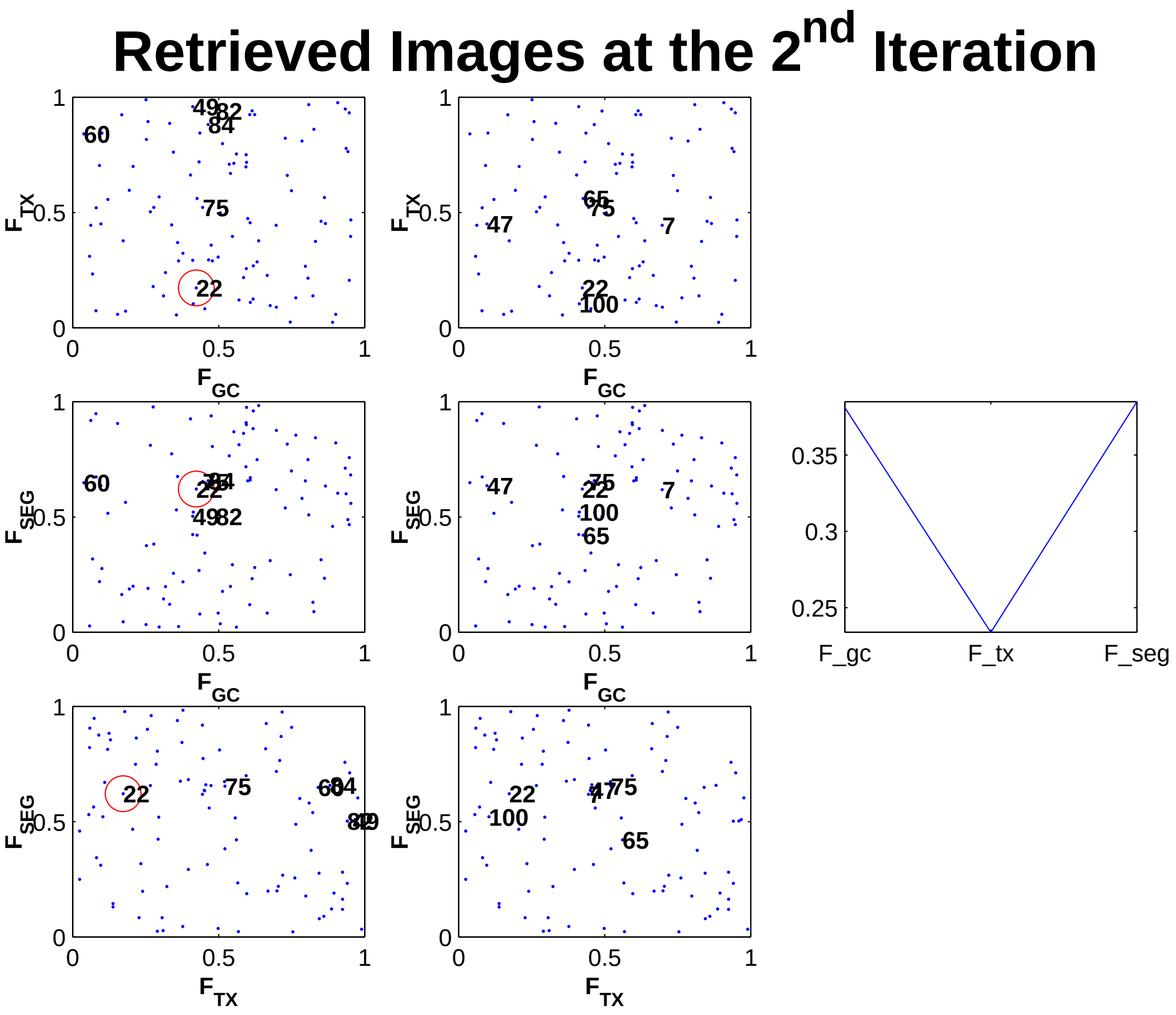


Figure 5.16: The second iteration.

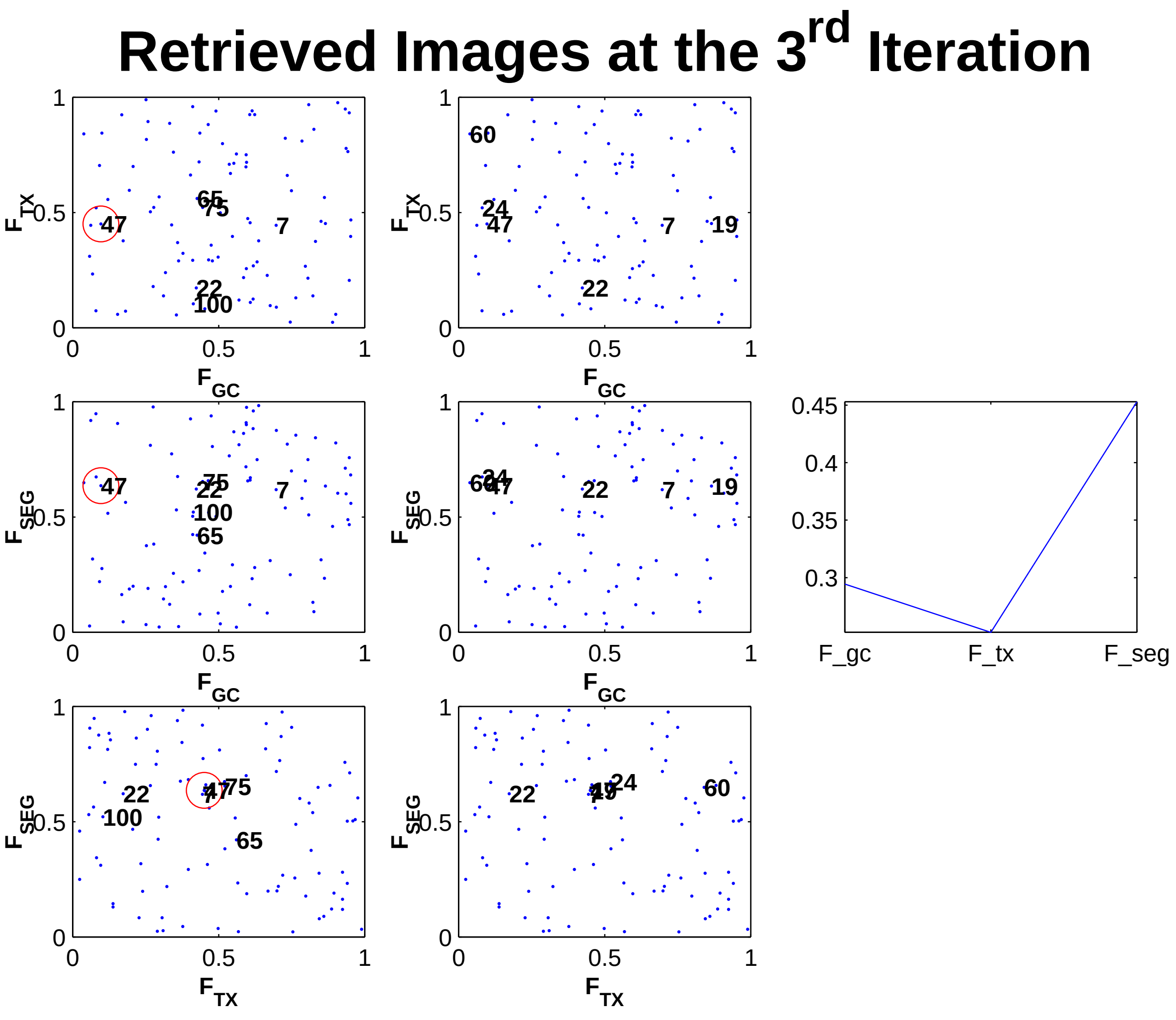


Figure 5.17: The third iteration.

and the marginal of F (bottom centre).

5.5 A More Complex Display Strategy

PicHunter was deliberately designed with a very simple user interface. Using the posterior $P(T, \sigma, F|H_t)$ we can generalise the current system to include a more complex display. This display will show, to the user, retrieval results based on three different similarities; similarity based in global colour features, similarity in texture features and similarity in segmentation features. Giving the posterior $P(T, \sigma, F|H_t)$ we can distill the three conditional probabilities of T given F where $F \in \{GC, TX, SG\}$. Then, the conditional probability of T given F_{GC} is given by

$$P(T|F_{GC}, H_t) = \frac{P(T, F_{GC}|H_t)}{P(F_{GC}|H_t)}, \tag{5.21}$$

where we need the joint marginal of T and F_{GC}, defined as

$$P(T, F_{GC}|H_t) = \int_0^1 P(T, \sigma, F_{GC}|H_t)\ d\sigma, \tag{5.22}$$

and also we need the marginal of F_{GC} which is defined as

$$P(F_{GC}|H_t) = \sum_{i=1}^{N} \int_0^1 P(T, \sigma, F_{GC}|H_t)\ d\sigma; \tag{5.23}$$

similarly, we obtain $P(T|F_{TX}, H_t)$ and $P(T|F_{SG}, H_t)$. The output of a system such as this will be like Figure 5.20, where in the left side is shown the first display set with six random images. The user picks one according to his need and three display sets are returned based on the three different similarities. Obviously, the user model needs to be adjusted accordingly to accommodate the user's feedback. We can see these three different display sets as one large set D_{t+1} or separately. It also may make sense to pick more than one displayed image. Proposing new likelihoods that model more complex data is another possibility for future work.

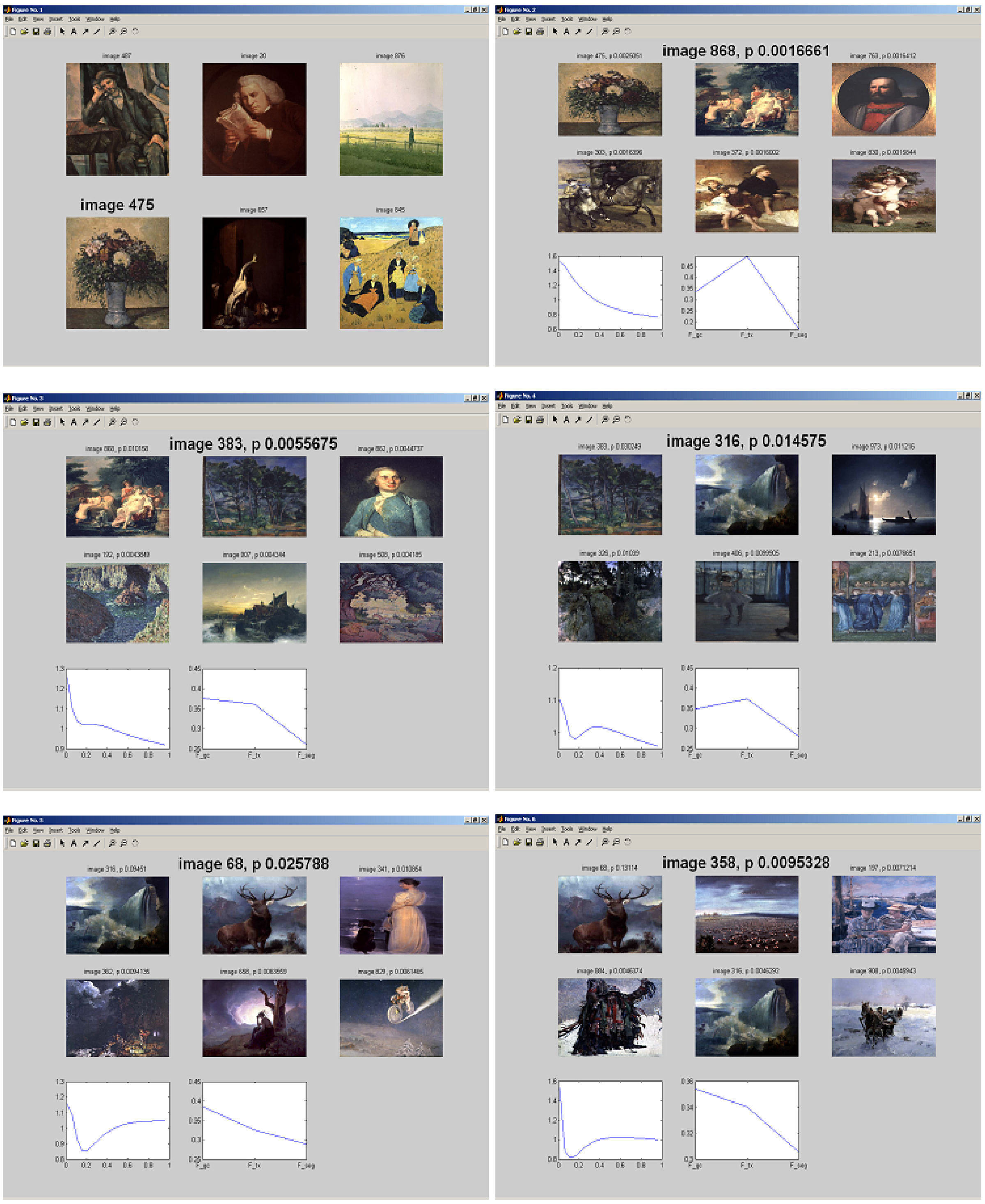


Figure 5.18: The first six iterations of a search using the $P(T, \sigma, F|H_t)$ model.

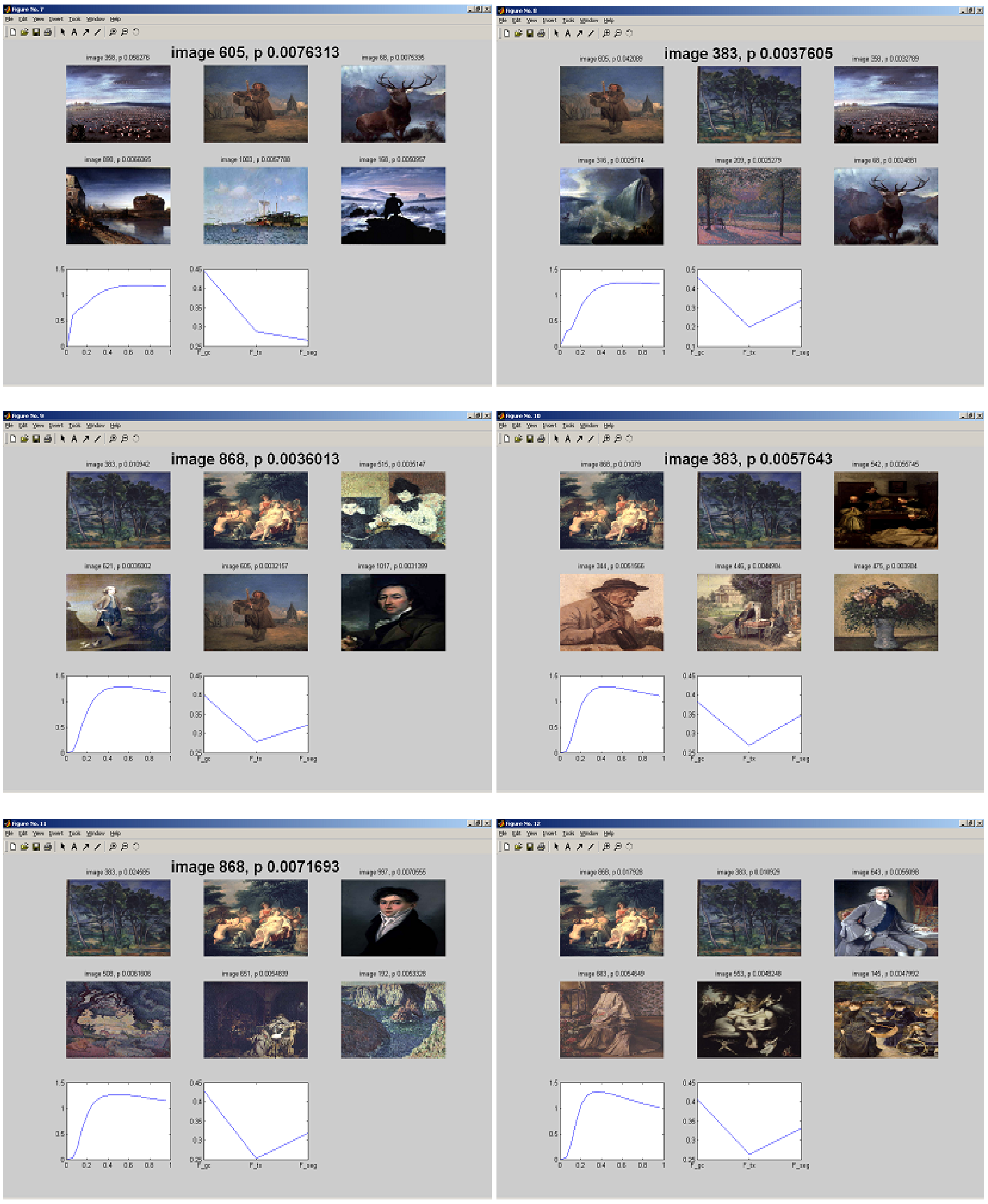


Figure 5.19: The next six iterations of a search using the $P(T, \sigma, F|H_t)$ model.

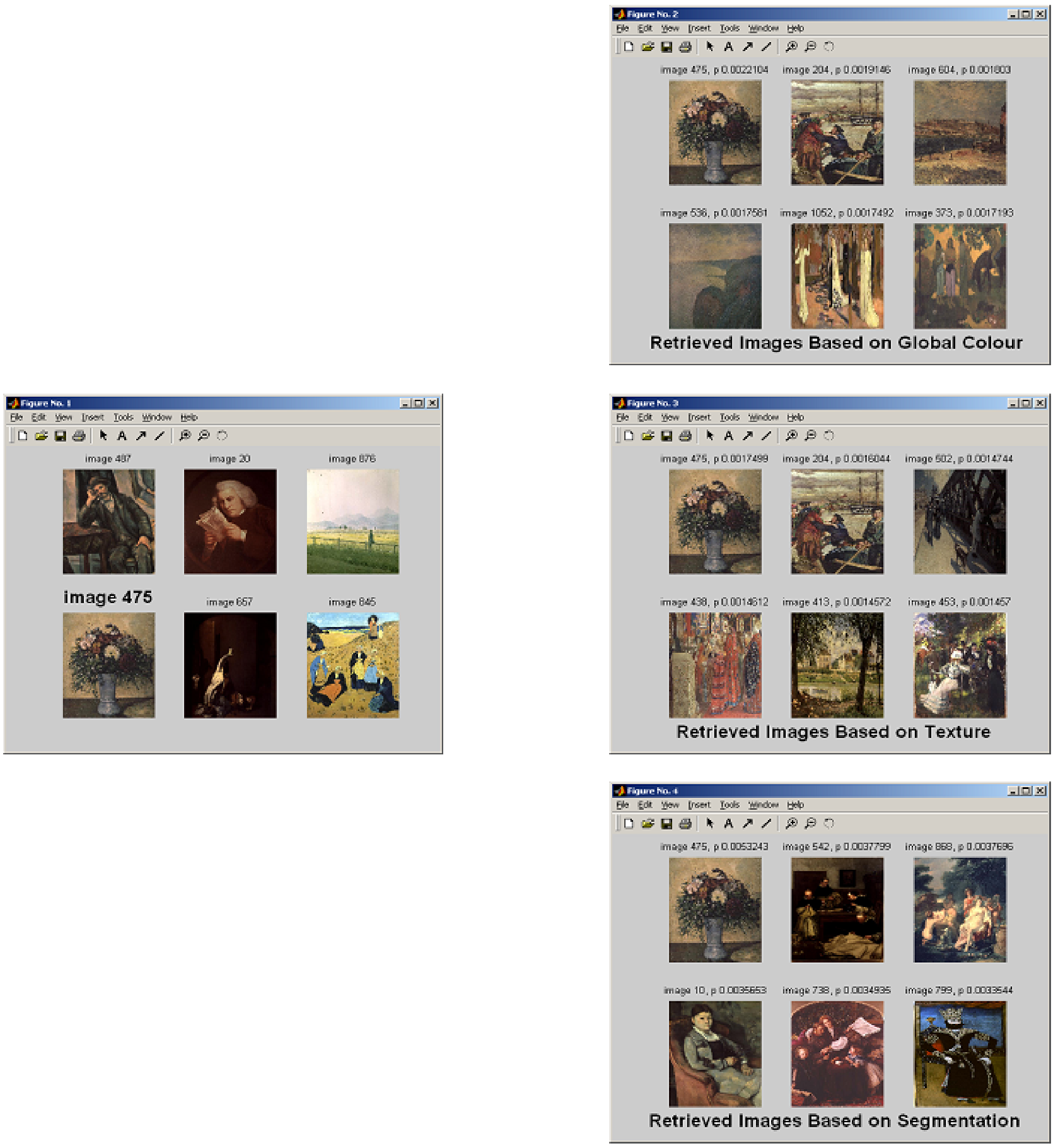


Figure 5.20: A more complex display where three different similarities are displayed.

Chapter 6

Display Updating Schemes

6.1 Introduction

This Chapter describes a decision-theoretic solution to the display strategy problem for the Bayesian image retrieval system that we are concerned with. Once *PicHunter* updates the posterior distribution across the entire database, the next task is to select the N_D images to be shown in the next display. The usual strategy is the most probable scheme, which we are going to present from a decision- theoretic perspective. While the most probable scheme is a reasonable strategy for target-specific search, for category search or open-ended browsing, where users search through a database with a rather broad, nonspecific goal in mind, it may not be appropriate to display the most probable images to a user, because it does not allow us to capture a user's needs in a broader sense. For example, a user may not initially have the query image at hand or the ideal query may evolve during the retrieval process. In other words, by developing other display strategies we are proposing an open-ended search browser rather than a target-specific search system. This contribution fills the need for such a system in applications in graphics, photojournalism and advertising.

6.1.1 Open-ended Browsing

Many graphic design problems have to do with finding an image in situations where the concept of "right image" is not well defined. When a designer is creating, say, a brochure for a new product, he typically starts with an imprecise idea of the content of the final brochure, and then searches catalogues (or database) to look for inspiration.

We take an example from Santini (2001), who describes the story of an image search for a brochure for a new model of washing machine. The main selling point of the new washing machine is that it uses comparatively little electric power and water. Due to the connection with water, an obvious idea is to look for some kind of ocean or sea image. However, in addition the image has to accomodate a picture of the washing machine. After browsing through catalogues of stock images, it might be realised that an image of a river with salmon fighting the current might be more appropriate for the brochure. Such a realisation would not be possible using a most probable image display strategy, which would try to only display ocean images and not give the user the opportunity to see alternatives.

We contend that in this kind of problem the concept of the "right answer" must be abandoned. A search may start by looking for a certain class of images but, while browsing, this choice may change. This may be due to a perceived deficiency in the database or because the act of exploring the database unveiled a new possibility not thought of until that moment.

Graphic design involves a mix of searching and browsing through databases. The designer does not just search for images satisfying a certain query; he needs to explore the database to see what it has to offer and to look for new ideas. The necessity of browsing highlights another interesting aspect of this problem; although all stock images companies provide CDs with a low resolution version of their images (which can be used for the first test before the designer buys the image), all browsing is done with paper catalogues. Computer browsing is still not good enough to substitute for the paper catalogue. The low resolution of the computer screens creates a disadvantage that adequate browsing technologies will have to overcome in order to make this medium successful. Also in Web-oriented applications (online shopping), because of bandwidth limitations pictures take a long time to appear on a Web page. The more detailed the picture, the longer its download time. As a result, Web page designers must use images sparingly and strategically.

Finally, the "query" in this case is not just about the objects contained in the image. The right image must have a space in which the image of the washing machine could fit. The search is no longer just for presence (an image with such and such objects) but for an absence (an area at least this size without any object).

6.1.2 Art Image Retrieval

As we have mentioned in earlier stages, low level features like colour and texture reflect some important characteristics of paintings. However, as shown in Figure 6.1, a user's interests in art images is typically diverse. According to Yu et al. (2003), a user's preference for art images can be described as a union of disjoint regions in the low-level feature space, as shown in Figure 6.1, where consistency of interests is only held in a local region.

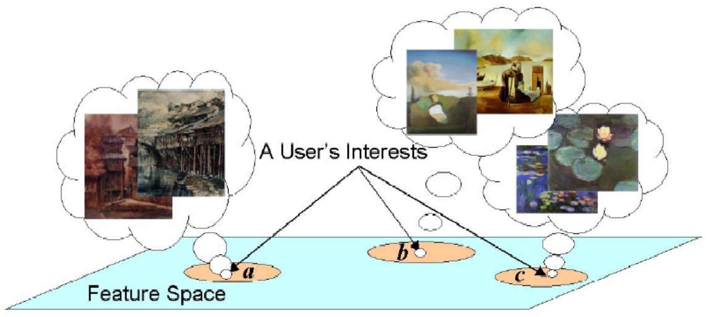


Figure 6.1: A user's diverse preferences for art images, represented as union of disjoint regions in visual feature space. A question is, if we only show the most probable images to the user saying in region a, how to infer the user's preferences for region b or c, (Figure has been taken from Yu et al. (2003)).

6.2 Most Probable Display Updating Scheme

As we mentioned earlier, this is certainly a reasonable strategy in target searching. The marginal posterior distribution of T is used to decide which set of images to display at the next iteration. In this system the N_D images with the highest marginal posterior probability are selected for D_{t+1}. So the CBIR system works as follows:

1. Initialise by randomly or otherwise selecting a subset $D_1 \subset \mathcal{I}$ of N_D images and display to the user. Let $t = 1$.

2. Repeat until the user finds the target image:

   - User selects $A_t \in D_t$;
   - $P(T|H_t)$ is computed;
   - D_{t+1}, the set of N_D highest probability images from $P(T|H_t)$, is displayed;
   - $t = t + 1$.

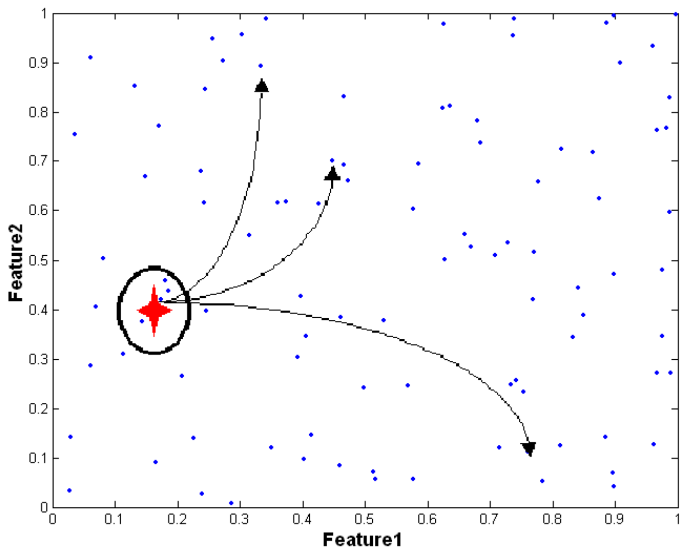


Figure 6.2: Most probable display updating scheme in a 2-dimensional feature space.

In Figure 6.2 we can see how this works in a two dimensional feature space. The red star denotes the image that the user picked and inside the circle the most probable images are returning to him. Clearly the system will not display images that are not close in feature space to those selected.

6.3 Related Work on Display Selection Strategies

There is not a large literature on different display updating schemes for CBIR systems. Probabilistic approaches have, we believe, an advantage over other methods in that decision theory can be naturally applied to address this question, but their implementation to CBIR is relatively new.

In Cox et al. (2000), the most informative display updating scheme tried to achieve this, since each image is associated with a probability value, a maximum entropy display is used to select images. A sampling approach is used to choose images that maximise the entropy.

Another approach relevant to display selection has been proposed by Sia and King (2002) where they estimate the user's target distribution parameters, thus they can select images located in the boundary to display. Since they have estimated the mean μ and variance δ of the user's target distribution, they choose images around $\mu + \kappa\delta$ or $\mu - \kappa\delta$ in each dimension to display, for a given scaling parameter κ.

6.4 Decision Theoretic Approaches To Display Strategies

The objective is to decide on a set of images D to display. Given our marginal posterior of T, which set of images D_{t+1} should next be displayed? This is a decision problem. Any decision theory problem has three components:

- Actions / Alternatives. These are the possible decisions that we can take. In our case it is the sets of images D that we can display, that is to say all possible subsets of N_D images.

- Unknowns / State of Nature. These are the unknown quantities that are going to affect how good is our decision. There is a probability distribution on the unknown; in this case it is the true target image T, and we have the posterior distribution $P(T|H_t)$

- Utility / Consequence. For each combination of an action and unknown, we must define a utility that is the "value" of choosing such an action for such a value of the unknown. So in our case it will be some function $U(D, T)$ of the set of images we display and the target image.

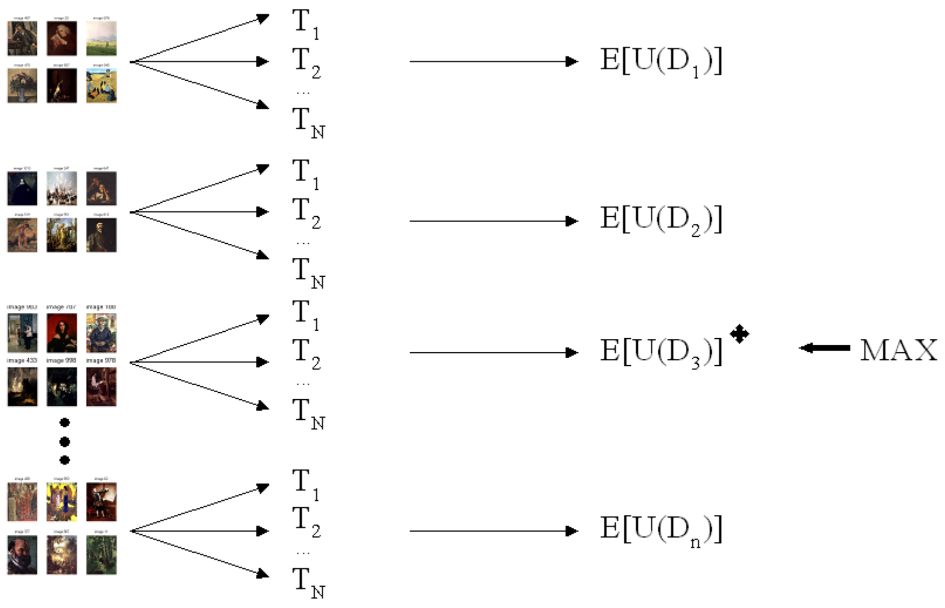


Figure 6.3: A Decision Tree for the display strategy problem.

Given these three components, decision theory tells us that we pick the action that maximizes expected utility, see also the Figure 6.3 which schematically summarises the above components. The principle of maximizing expected utility has long been established as the guide to making rational decisions. The utility function models our preferences for what constitutes a good display strategy. For our scope we model two preferences. First, we want to retrieve images which are similar to the query image that the user provides (most probable scheme). Second, we want a more open-ended search browser system.

Given a utility $U(D,T)$ of picking the set D to display when the target image is T, we compute for each possible D the expected utility with respect to $P(T=T_i|H_t)$:

$$\mathcal{U}(D) = \sum_{i=1}^{N} U(D,T_i)\, P(T=T_i \,|\, H_t). \tag{6.1}$$

The optimal set to display is that D which maximises expected utility:

$$D_{t+1} = \arg \max_{\substack{D \subset \mathcal{I} \\ |D| = N_D}} \mathcal{U}(D), \tag{6.2}$$

see Winston (1987).

6.4.1 The Most Probable Display Scheme Using the Indicator Utility

The most obvious display scheme is to display those N_D images with the highest posterior probability. We observe that if we define

$$U_I(D,T) = \begin{cases} 1, & \text{if } T \in D, \\ 0, & \text{otherwise,} \end{cases}$$

then

$$\mathcal{U}_I(D) = \sum_{T_i \in D} P(T = T_i \mid H_t)$$

which is clearly maximised by those images with highest probability. We call this the indicator utility. This is the strategy used in Chapter 4 and 5.

6.4.2 Variance Utility

In order to allow open-ended browsing, a wider variety of images should be displayed. This gives us more "information" about the user's preferences. We might be able to achieve this by displaying a set of images that are widely dispersed in feature space. We can use the variance of the distances between images in D and T to define a measure of dispersion, thus

$$U_V(D,T) = \frac{1}{N_D - 1} \sum_{T_i \in D} (d(T_i, T) - \bar{d})^2,$$

where $d(T_i, T)$ is a normalised distance measure in feature space and

$$\bar{d} = \sum_{T_i \in D} d(T_i, T)/N_D$$

is the mean distance of images in D to T.

6.4.3 Entropy Utility

An alternative approach is to try to maximise the amount of information that D will give about the user's preferences. One measure of information is the reduction in entropy in the distribution of T by selecting a particular display set. So we can define a utility based on the negative expected entropy of the posterior of T from picking an image in D, expectation over the images in D:

$$U_E(D, T) = -\sum_{A_j \in D} \mathcal{E}(A_j, D)\, P(A_j \mid D, T), \tag{6.3}$$

where

$$\mathcal{E}(A_j, D) = -\sum_{i=1}^{N} P(T = T_i \,|A_j, D)\ \log(P(T = T_i \,|A_j, D))$$

is the entropy of the posterior distribution of T given that A_j is picked from D (following Equations 5.15 and 5.18) and

$$P(A_j \mid D, T) = \frac{\exp\left(-d(A_j, T)/\overline{\sigma}\right)}{\sum_{T_j \in D} \exp\left(-d(T_j, T)/\overline{\sigma}\right)}$$

is the likelihood term as in Equation 5.16 but, in order to reduce the computation time, using the distance measure d over the entire feature space and $\overline{\sigma}$ is the posterior mean of σ.

6.4.4 Combining Utility Functions

The two new utilities that we have proposed — variance and entropy — are primarily of use in the early stages of a query, when the objective is to learn as much as possible about the user's target. Ultimately, one will want to resort to a utility that displays images close to the target, such as the indicator. An obvious way to do this is to consider a utility that is a convex weighted combination of the indicator utility with one of the other two, with the weight on the indicator utility increasing to 1 with the iteration, for example at the t^{th} display set:

$$U(D,T) = \alpha_t U_I(D,T) + (1-\alpha_t)U_E(D,T),$$

with $0 \leq \alpha_t \leq 1$ and $\alpha_t \to 1$, and the entropy utility normalised by linear scaling to unit range (see Section 3.5) from that in Equation 6.3 so that it lies in $[0,1]$ like $U_I(D,T)$.

6.4.5 Optimisation Methods

Because $\mathcal{U}_I(D)$ is separable in each element of D, the optimal D for the indicator utility can be easily computed. This is not the case if one moves to using the variance or entropy utilities. Evaluation of the expected utility for all possible D is not an option as the number is large e.g. for $N = 1000$ and $N_D = 6$ we have about 2×10^{15} possible subsets. For these, we have to resort to numerical methods that are not guaranteed to find the optimal. In the last 20 years, a new kind of approximate algorithm has emerged which basically tries to combine basic heuristic methods in higher level frameworks aimed at efficiently and effectively exploring a search space. These methods are nowadays commonly called *metaheuristics.* The field of metaheuristics for the application to combinatorial optimisation problems is a rapidly growing field of research. This is due to the importance of combinatorial optimisation problems for the scientific as well as the industrial world. A nice survey of such algorithms can be found in Blum and Roli (2003) where there are possibilities for applying algorithms other than simulated annealing. We propose two Monte Carlo optimisation schemes.

Random Generation

We randomly generate without replacement, from a distribution on T, K subsets $D^1, \ldots, D^K$. Then we let

$$D_{t+1} = \arg \max_{k=1}^{K} \mathcal{U}(D^k). \tag{6.4}$$

Each element of a set D can be simulated from any distribution on $\mathcal{I}$; obvious choices are the uniform and $P(T \mid H_t)$. In this work we choose the latter.

Simulated Annealing

Simulated annealing is a Monte Carlo approach for minimizing (or maximizing) functions. Simulated Annealing is commonly said to be the oldest among the metaheuristics and one of the first algorithms that had an explicit strategy to escape from local minima (maxima). The origins of the algorithm are in statistical mechanics (Metropolis algorithm) and it was first presented as a search algorithm for combinatorial optimisation problems in Kirkpatrick et al. (1983). The fundamental idea is to allow moves resulting in solutions of worse quality than the current solution (uphill moves) in order to escape from local minima (maxima). The probability of doing such a move is decreased during the search. The simplicity of the algorithm is very appealing.

For simulated annealing, we define a "neighbour" of a subset D to be another subset with one different image. We begin by sampling one random display set D from $P(T|H_t)$. Then we randomly choose a position in D and replace this image with another one that is randomly selected from $P(T|H_t)$; see Figure 6.4. A simulated annealing algorithm then runs as follows:

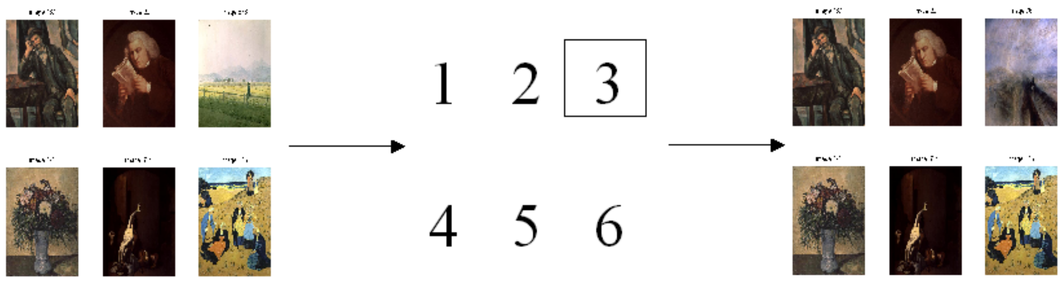


Figure 6.4: Creating a neighbour in Simulating Annealing.

1. Define an initial temperature T_0, a final temperature $T_{\min}$ and a cooling schedule $T_1, T_2, \ldots$. Randomly generate without replacement a set D^0, using $P(T \,|\, H_t)$. Let $k = 0$.

2. While $T_k > T_{\min}$

   - $k = k + 1$.
   - Select at random one image in D^{k-1} and replace with another image in $\mathcal{I} - D^{k-1}$, randomly generated according to $P(T \,|\, H_t)$. Call this new set D^{new}.
   - With probability $\min\{1, \exp((\mathcal{U}(D^{\text{new}}) - \mathcal{U}(D^{k-1}))/T_k)\}$, let $D^k = D^{\text{new}}$ else $D^k = D^{k-1}$.

3. $D_{t+1} = D^k$.

Each iteration (k) requires the computation of one expected utility. Initial and final temperatures were decided on by using the methods of Kirkpatrick et al. (1983) where a suitable initial temperature is argued to be

$$T_0 = -\frac{\frac{1}{n}\sum_{i=1}^{n} E[\mathcal{U}(D_i^{new})] - E[\mathcal{U}(D^i)]}{\ln(0.8)}. \tag{6.5}$$

Other temperature schedules are discussed in Stander and Silverman (1994). Determining the starting temperature is to initially sample the search space. This can be done by applying the next pseudocode (for a big n; 10000 for the simulated example and 1000 for the BAL database due to costly computation) :

```
for $i = 1 : n$
            CREATE $D^i$
            CALCULATE $E[\mathcal{U}(D^i)]$
            CREATE $D^{new}$
            CALCULATE $E[\mathcal{U}(D_i^{new})]$
            diff(iteration)=$|E[\mathcal{U}(D_i^{new})] - E[\mathcal{U}(D^i)]|$
end
$T_0 = -\frac{\text{mean(diff)}}{\ln(0.8)}$
```

Lundy and Mees (1986) proposed stopping temperature when

$$T_{min} \leq \frac{\lambda}{\ln[(|S|-1)/\theta]}, \tag{6.6}$$

where $|S| = \binom{N}{N_D}$ is the number of display sets D. This is designed to produce a solution which is within λ of the optimum with probability θ. The λ which has been set 0.001, is the tolerance parameter (how close to actual solution) and $\theta = 0.99$ (high probability).

In theory the procedure should be continued until the final temperature T_{min} is zero, but in practice it is sufficient to stop when the chance of accepting a downhill move has become negligible. We should note that when S is very large (e.g. combinations of displaying 6 images out of 1066) the calculation of the final temprature needs some more algebra; see Appendix A.1.

Then, we looked at several different cooling schedules such as:

- *Geometric* where $T_k = ab^t$, for some a and $0 \leq b \leq 1$;
- *Inverse log* where $T_k = a/\ln(1+k)$;

- *Inverse linear* where $T_k = a/(1 + bk)$, for suitable a, b.

We found that inverse linear ($T_k = a/(1 + bk)$) performed best; see Appendix A.2. The choice of $P(T \mid H_t)$ to generate D^0 and D^{new} can be changed, to for example the uniform, but we found that the method was not particularly sensitive to this choice.

According to Blum and Roli (2003) of great importance hereby is that a dynamic balance is given between *diversification* and *intensification*. The term diversification generally refers to the exploration of the search space, whereas the term intensification refers to the exploitation of the accumulated search experience. The balance between diversification and intensification is important, on one side to quickly identify regions in the search space with high quality solutions and on the other side not to waste too much time in regions of the search space which are either already explored or which do not provide high quality solutions.

To this end, our definition of neighbour can be made more or less strict, by for example allowing two changes for D^{new} or, conversely, only favouring replacement of one image in D^{new} that is close in feature space to that image replaced. However we found that our choice was a compromise between a too small and too large change that offered a good accept rate.

Finally we note that computation time is limited in a live implementation of either optimisation scheme, so typically we can compute only a small number of expected utilities.

6.5 Comparison of Optimisation Methods

6.5.1 Small Simulated Database

To compare the display strategies and evaluate the performance of the optimisation approaches, we have a simulated database of only $N = 15$ images, each with only 2 features, for which queries are implemented by displaying $N_D = 3$ images. This is clearly an unrealistically small example but it has the advantages of allowing us to display what happens in feature space and, since there are only 455 possible subsets of size 3, to compute the exact optimal subset under all 3 utilities and compare with the results obtained by random sampling and simulated annealing. Figure 6.5 shows an example of the system where one image A_1 is picked from an initial display set D_1, and the resulting choice of D_2 according to the 3 utilities. Upper left of the figure are the 15 images in feature space with $D_1 = \{T_2, T_3, T_5\}$ highlighted. Image 3 is selected. We then see that, under U_I, we have $D_2 = \{T_3, T_8, T_{12}\}$, that is images close to that selected. For U_V we have $D_2 = \{T_5, T_7, T_{13}\}$ and for U_E we have $D_2 = \{T_7, T_{13}, T_{14}\}$, that is images that are widely separated in feature space are chosen. We should note that the posterior of T is non informative (flat) at this very first iteration; see Figure 6.6. To explore the effectiveness of the optimisation methods, we repeated this experiment 1000 times, computing the optimal D_2 according to the random generation and simulated annealing methods. For the random subset generation, we simulated $K = 100$ subsets. For the simulated annealing we used an inverse linear cooling schedule $T_k = a/(1 + bk)$ with a and b chosen so that to go from the initial to the final temperature, from Equations 6.5 and 6.6, took 100 iterations, thus both methods took the same time to compute. The results are compared with the exact calculation in Table 6.1 and we see that both non-exact methods are sub-optimal, but nevertheless do manage on average

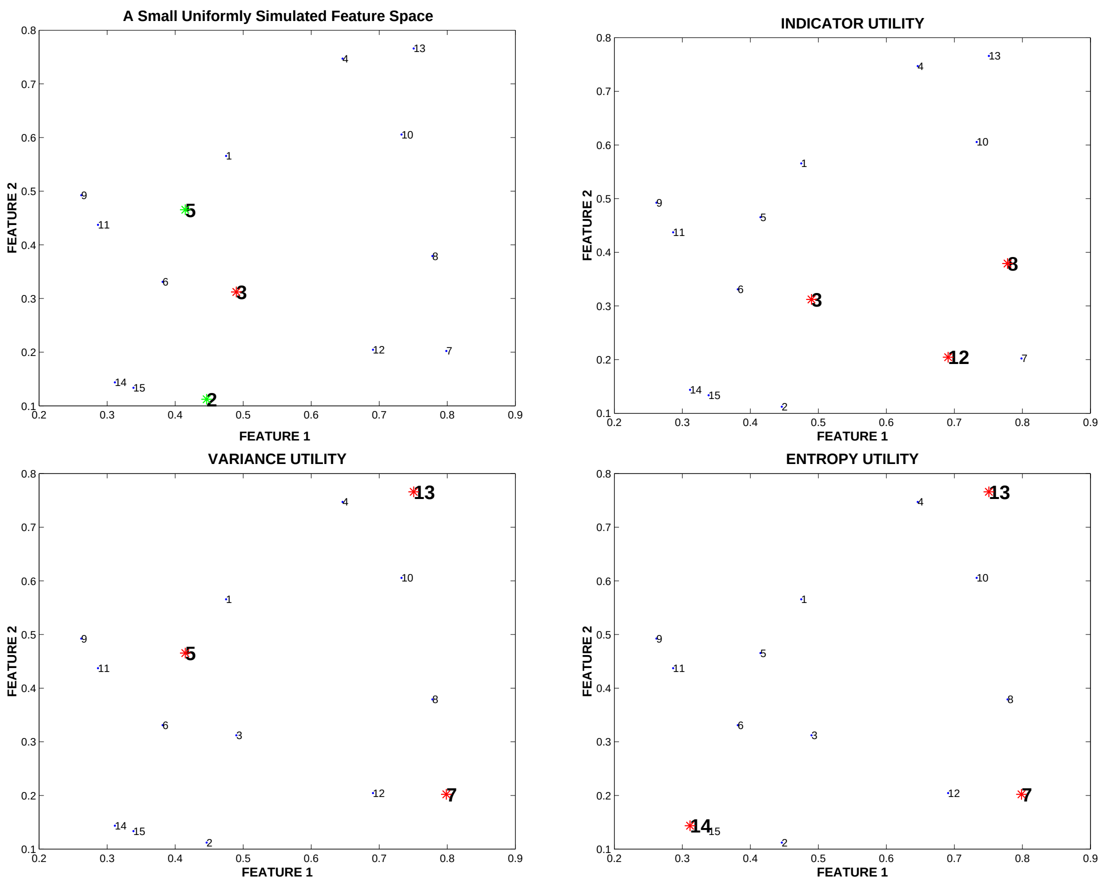


Figure 6.5: Feature space plots for the selection of D_2 for a simple database of 15 images given $D_1 = \{T_2, T_3, T_5\}$ and $A_1 = T_3$. Images in D_2 are highlighted by *.

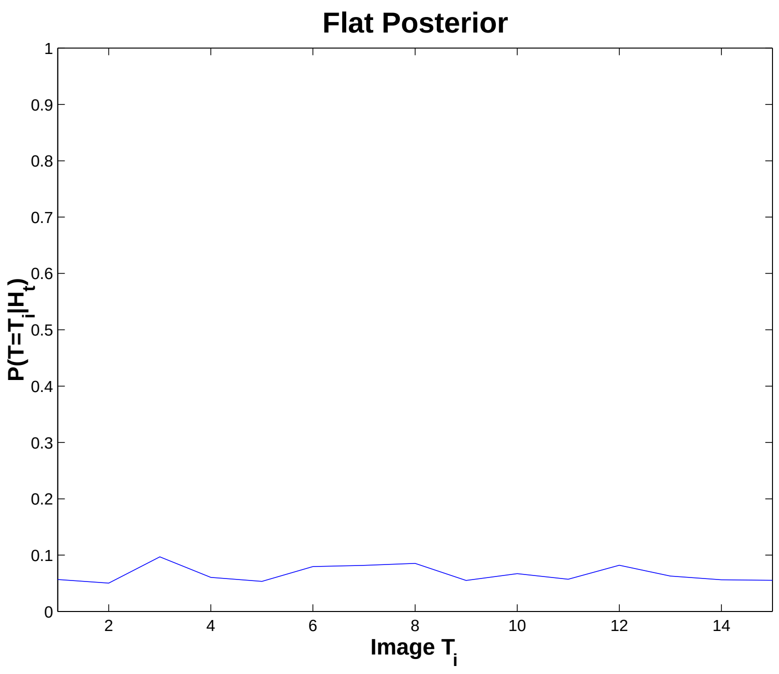


Figure 6.6: A non informative posterior $P(T|H_t)$.

to find subsets with expected utility close to the optimal. Random generation appears to do slightly better than simulated annealing.

Also, we investigated the effect on the optimisation methods of a more informative posterior probability of T. We constructed a more informative posterior of T like the one in Figure 6.7 and repeated the experiment of Table 6.1 for different optimisation schemes and different utilities. The results are shown in Table 6.2. There is an obvious effect for indicator utility relevant to a non informative and informative posterior. This effect is insignificant with respect to variance and entropy utilities. A 2-sample t-test hypothesis testing was conducted to test for the difference in means of two samples. For this specific application, the random generation scheme seems to be statistically significantly better than the simulated annealing when the flat posterior is used.

Utility	Exact Computation over all subsets	Random Generation of 100 subsets (μ_{rg})	Simulated Annealing 100 iterations (μ_{sa})	t-test2 $\alpha = 0.05$ $H_0 : \mu_{rg} = \mu_{sa}$ vs $H_1 : \mu_{rg} \neq \mu_{sa}$
Indicator	0.2643	0.2624	0.2591	Reject H_0 ($p = 0$)
Variance	0.2299	0.2264	0.2246	Reject H_0 ($p =$ 7.3466e-005)
Entropy	-2.6869	-2.6879	-2.6883	Reject H_0 ($p = 0.002507$)

Table 6.1: The average of the expected utility for D_2 over 1000 runs for the 3 utility functions and three computation methods. The average number of unique images scanned was 15 for the random generation scheme and 14.9 for the simulated annealing scheme. All runs use the example of Figure 6.6.

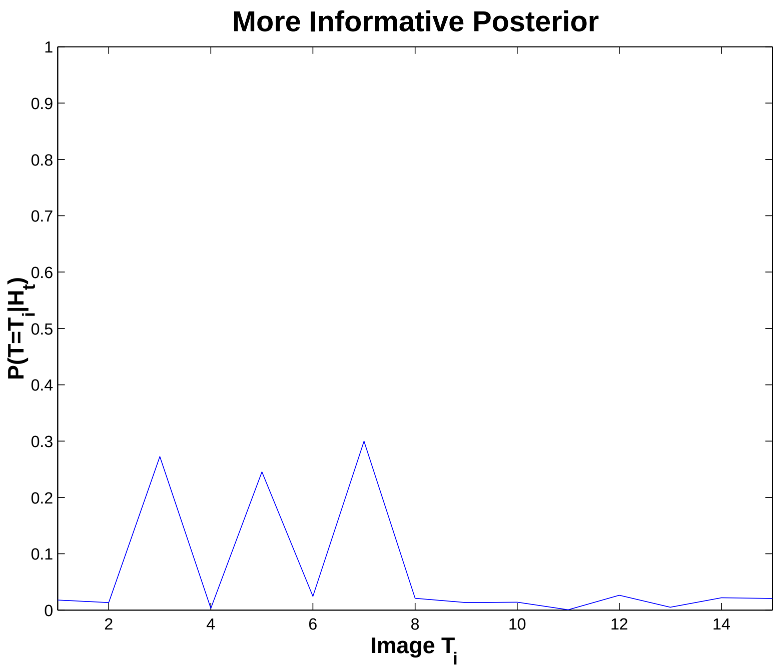


Figure 6.7: A more informative posterior $P(T|H_t)$.

Utility	Exact Computation over all subsets	Random Generation of 100 subsets (μ_{rg})	Simulated Annealing 100 iterations (μ_{sa})	t-test2 $\alpha = 0.05$ $H_0 : \mu_{rg} = \mu_{sa}$ vs $H_1 : \mu_{rg} \neq \mu_{sa}$
Indicator	0.81778	0.81778	0.81778	Not Reject H_0 ($p = 1$)
Variance	0.24033	0.23044	0.2278	Not Reject H_0 ($p = 0.29031$)
Entropy	-2.6875	-2.6928	-2.6920	Reject H_0 ($p = 0.027687$)

Table 6.2: The average of the expected utility for D_t over 1000 runs for the 3 utility functions and three computation methods using the posterior of Figure 6.7. The average number of unique images scanned was 13.9 for the random generation scheme and 13.7 for the simulated annealing scheme.

6.5.2 BAL Database

Finally, we move to the Bridgman Art Library database. In this case we cannot enumerate all possible subsets and so the display set under the variance and entropy utilities can be computed by the two optimisation schemes only. To compare the schemes, we first use a flat posterior of T, as shown in Figure 6.8. Table 6.3 compares the optimisation methods over 10 runs for an example from this database where D_1 consists of 6 images; note that we only have the exact result for the indicator utility. From the results for the indicator utility it appears that both optimisation methods do not do well. From all 3 utilities and for these particular settings it appears that random generation performs better than simulated annealing.

Finally, we investigated the situation where the posterior probability of T is more informative and its effect on the optimisation methods. We constructed a more informative posterior of T as in Figure 6.9 and run again the experiment for different optimisation schemes and different utilities; see Table 6.4. The random generation scheme seems to be statistically significantly better than the simulated annealing in

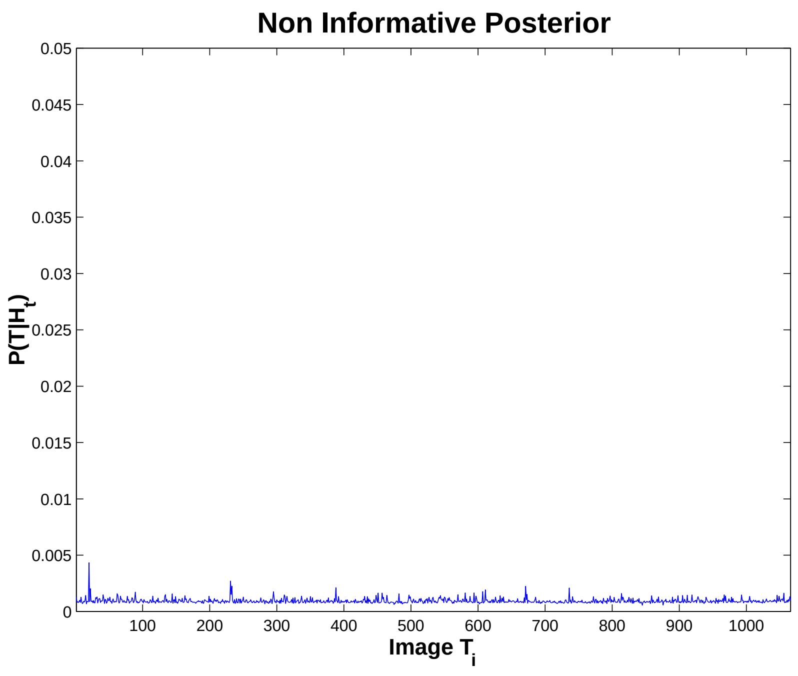


Figure 6.8: A flat posterior $P(T|H_t)$ at the first iteration running the BAL database.

Utility	Exact Computation over all subsets	Random Generation of 100 subsets (μ_{rg})	Simulated Annealing 100 iterations (μ_{sa})	t-test2 $\alpha = 0.05$ $H_0 : \mu_{rg} = \mu_{sa}$ vs $H_1 : \mu_{rg} \neq \mu_{sa}$
Indicator	0.0157	0.0094	0.0090	Reject H_0 ($p = 0.00077859$)
Variance	—	0.7029	0.6564	Not Reject H_0 ($p = 0.061245$)
Entropy	—	-6.9643	-6.9644	Reject H_0 ($p = 0.00040708$)

Table 6.3: The average of the expected utility for D_2 over 10 runs for the 3 utility functions and three computation methods using the BAL database and the posterior of Figure 6.8. The average number of unique images scanned was 450.3 for the random generation scheme and 101.6 for the simulated annealing scheme.

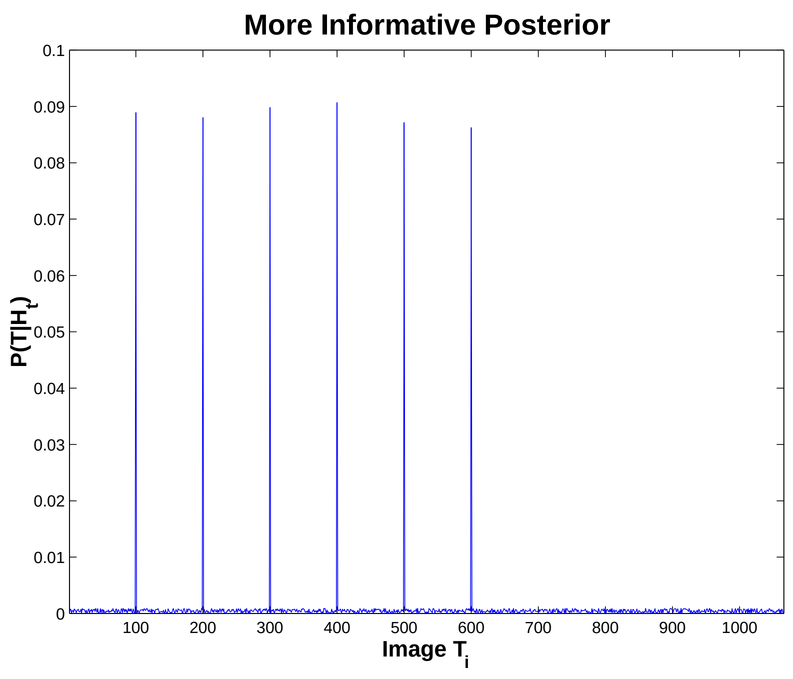


Figure 6.9: An informative posterior $P(T|H_t)$ for the BAL database.

total. Basically, random generation beat simulated annealing, but saw much more of the database, so the comparison is not strictly fair. However, in Table 6.2 both methods saw more or less the same amount of the database, and still random generation was better, which indicates that in spite of its advantage in the other runs, it is a better approach.

6.6 Examples of Display Strategies

We demonstrate three examples from the BAL database, using sets of $N_D = 6$ images. The first example demonstrates the system using the variance utility, the second example demonstrates the system using the entropy utility, and the third example using the combined utilities of indicator and entropy. They show the large effect that the choice of utility has on the search process. Another possibility is applicable using the

Utility	Exact Computation over all subsets	Random Generation of 100 subsets (μ_{rg})	Simulated Annealing 100 iterations (μ_{sa})	t-test2 $\alpha = 0.05$ $H_0 : \mu_{rg} = \mu_{sa}$ vs $H_1 : \mu_{rg} \neq \mu_{sa}$
Indicator	0.5308	0.4362	0.3908	Reject H_0 ($p = 2.7552\text{e-}006$)
Variance	—	0.7337	0.6295	Reject H_0 ($p = 0.045638$)
Entropy	—	-6.9647	-6.9650	Not Reject H_0 ($p = 0.11056$)

Table 6.4: The average of the expected utility for D_2 over 10 runs for the 3 utility functions and three computation methods using the BAL database and the posterior of Figure 6.9. The average number of unique images scanned was 273.9 for the random generation scheme and 88.5 for the simulated annealing scheme.

combination of the indicator and variance utility. The optimisation method used was the random generation, as from Section 6.5 it appears to give us better sub-optimal solutions than the simulated annealing approach.

We have not done any direct comparison of the 3 strategies here. This is because they are not directly comparable; the objective of the search is different with the indicator utility than with the other two. Between the variance and entropy utilities, we can compare them via a measure of information contained in the posterior distribution of T. We define entropy of T the quantity

$$\varepsilon(T) = -\sum_{i=1}^{N} P(T|H_t)\log(P(T|H_t))$$

From the principles of information theory we know that entropy measures the amount of information that we could get after an experiment's run. The experiment's run here is the user's pick at each iteration. As we can see, the entropy's graph is decreasing as the search goes on.

Variance Utility

In Figures 6.10 and 6.11 we show a user's search over 12 iterations using the variance utility. The bold title above an image indicates the user's selection.

Entropy Utility

In Figures 6.12 and 6.13 we show a user's search through 12 iterations using the system with the entropy utility. The bold title above an image indicates the user's selection. As we can see, the entropy's graph in general is decreasing as the search goes on. In other words, the system's uncertainty about the target image is getting low.

Combined utility of Indicator Utility and Entropy Utility

Finally, in Figures 6.14 and 6.15 we show a user's search 12 iterations of length using the system under the combined utility. The scenario here is that in the early stages of a search more weight is put in entropy utility, as the objective is to learn as much as possible about the user's target, where the search goes on, a certain amount of weight leaves from the entropy utility and transmitted to indicator utility. Table 6.5 shows exactly what were the weights α_t for the indicator utility and $(1 - \alpha_t)$ for the entropy utility in the combined utility for each iteration t for this particular search. These weights could be adjustable from the user in a real application. The bold title above an image indicates user's selection (we look for a portrait image class). As we can see, the entropy's graph general is decreasing as the search goes on.

In Table 6.6 we present the actual value of entropy of $P(T|H_t)$ which the system had at the last iteration. Summarising these results we can see that the search conducted using the variance utility system presents bigger uncertainty than the entropy utility system because its entropy has bigger value than the second's entropy. Obviously the

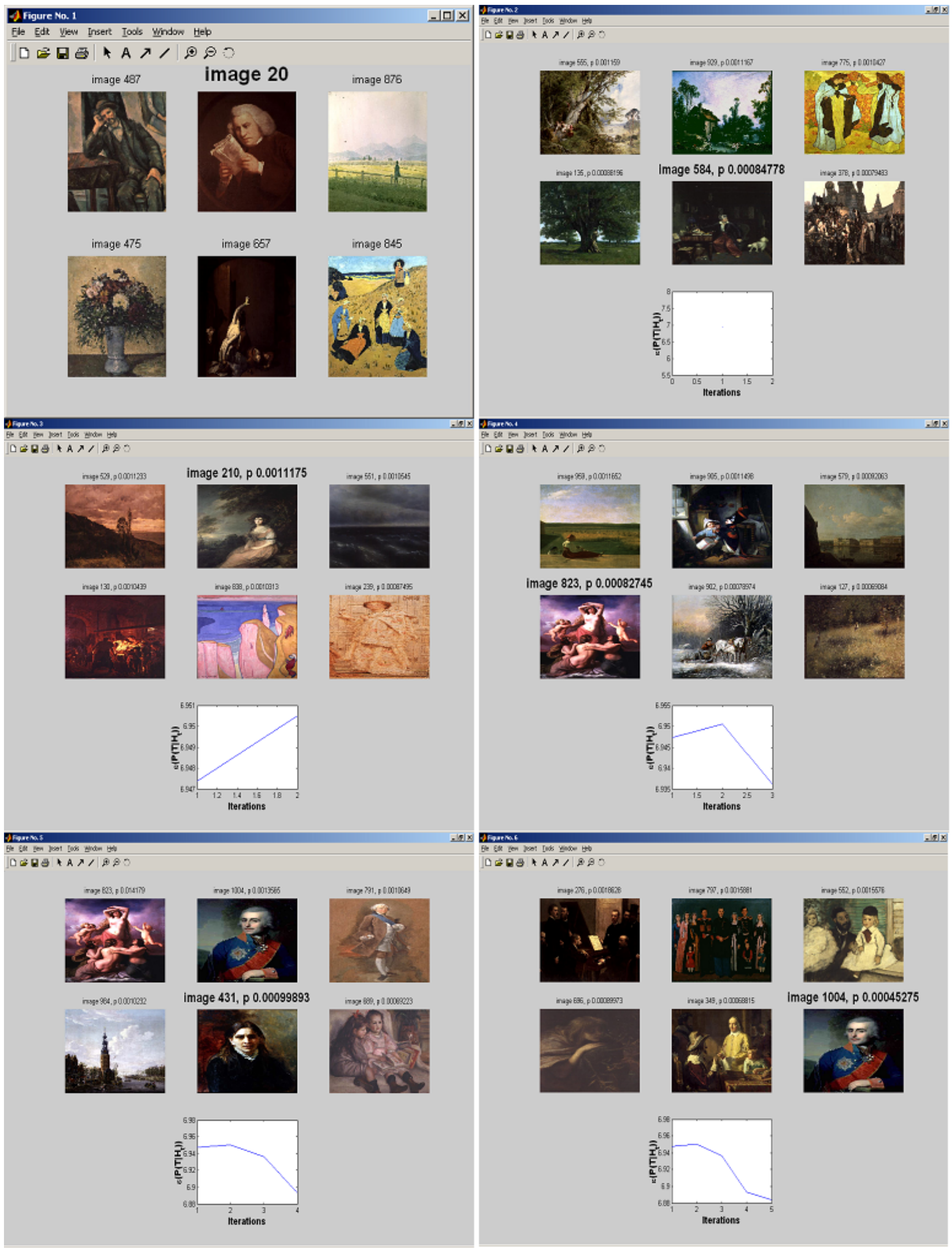


Figure 6.10: The first 6 iterations using the variance utility. Images 20 (top left window), 584 (top right window), 210 (middle left window), 823 (middle right window), 431 (bottom left window), 1004 (bottom right window) are selected.

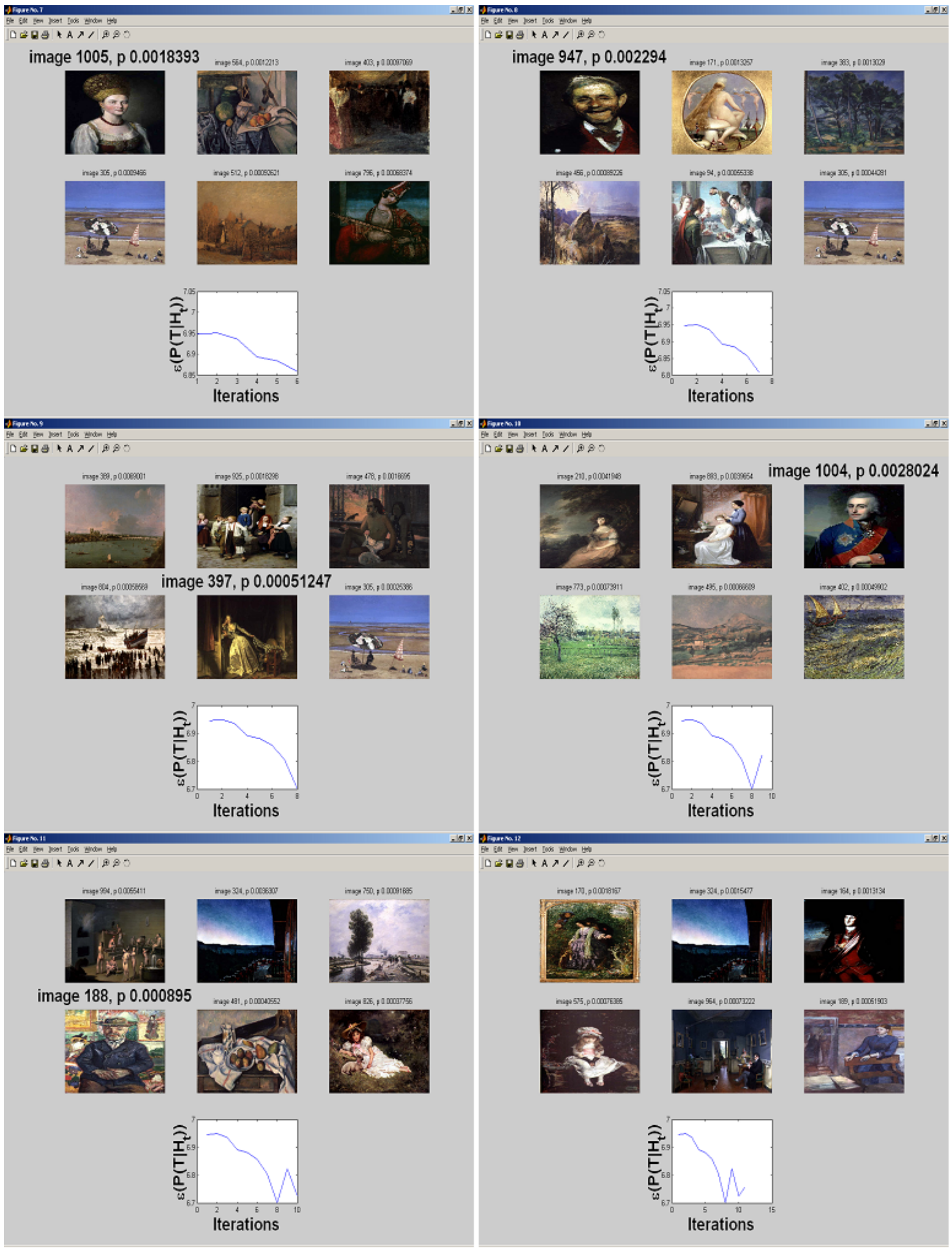


Figure 6.11: The next 6 iterations using the variance utility. Images 1005 (top left window), 947 (top right window), 397 (middle left window), 1004 (middle right window), 188 (bottom left window), 164 (bottom right window).

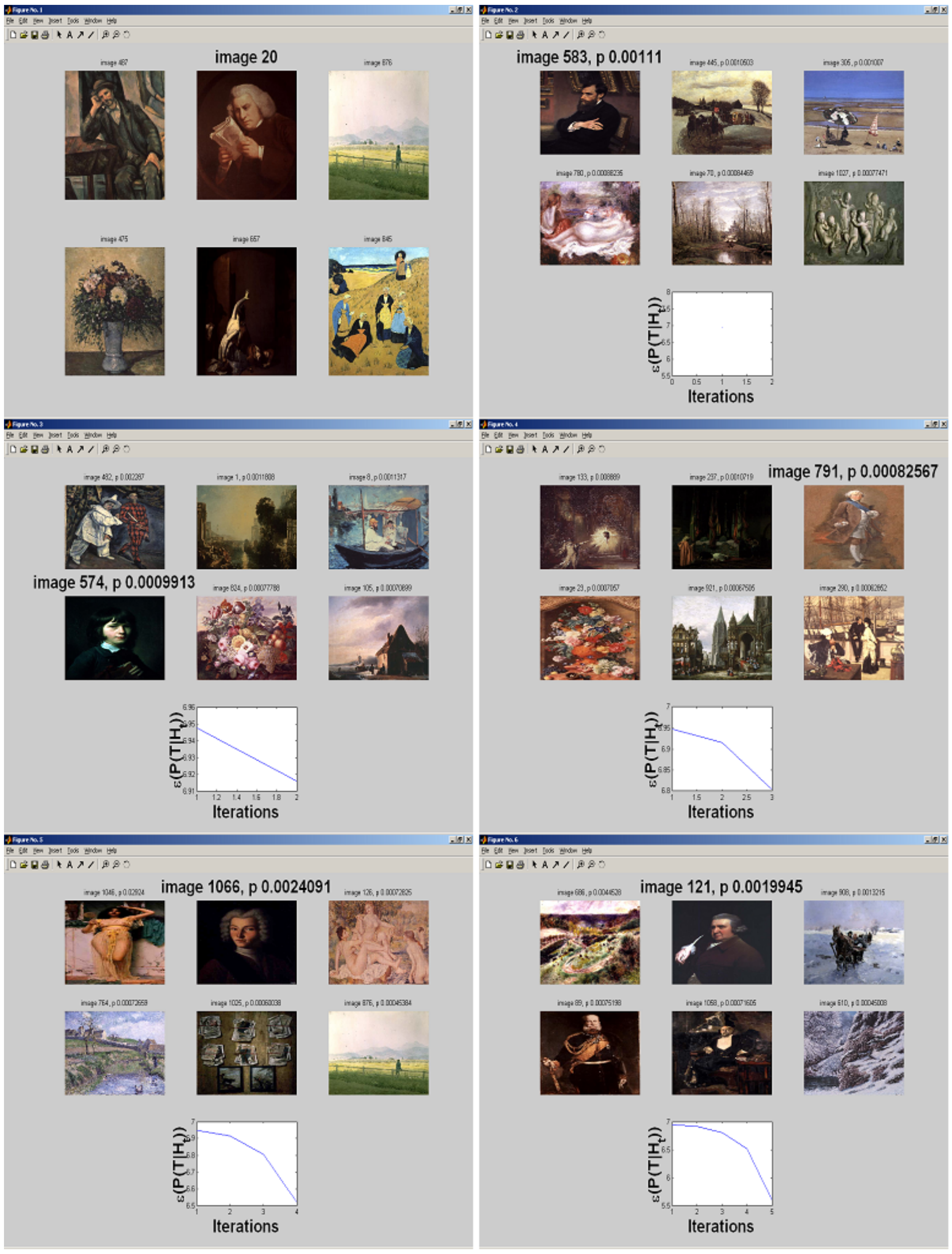


Figure 6.12: The first 6 iterations using the entropy utility. Images 20 (top left window), 583 (top right window), 574 (middle left window), 791 (middle right window), 1066 (bottom left window), 121 (bottom right window) are selected.

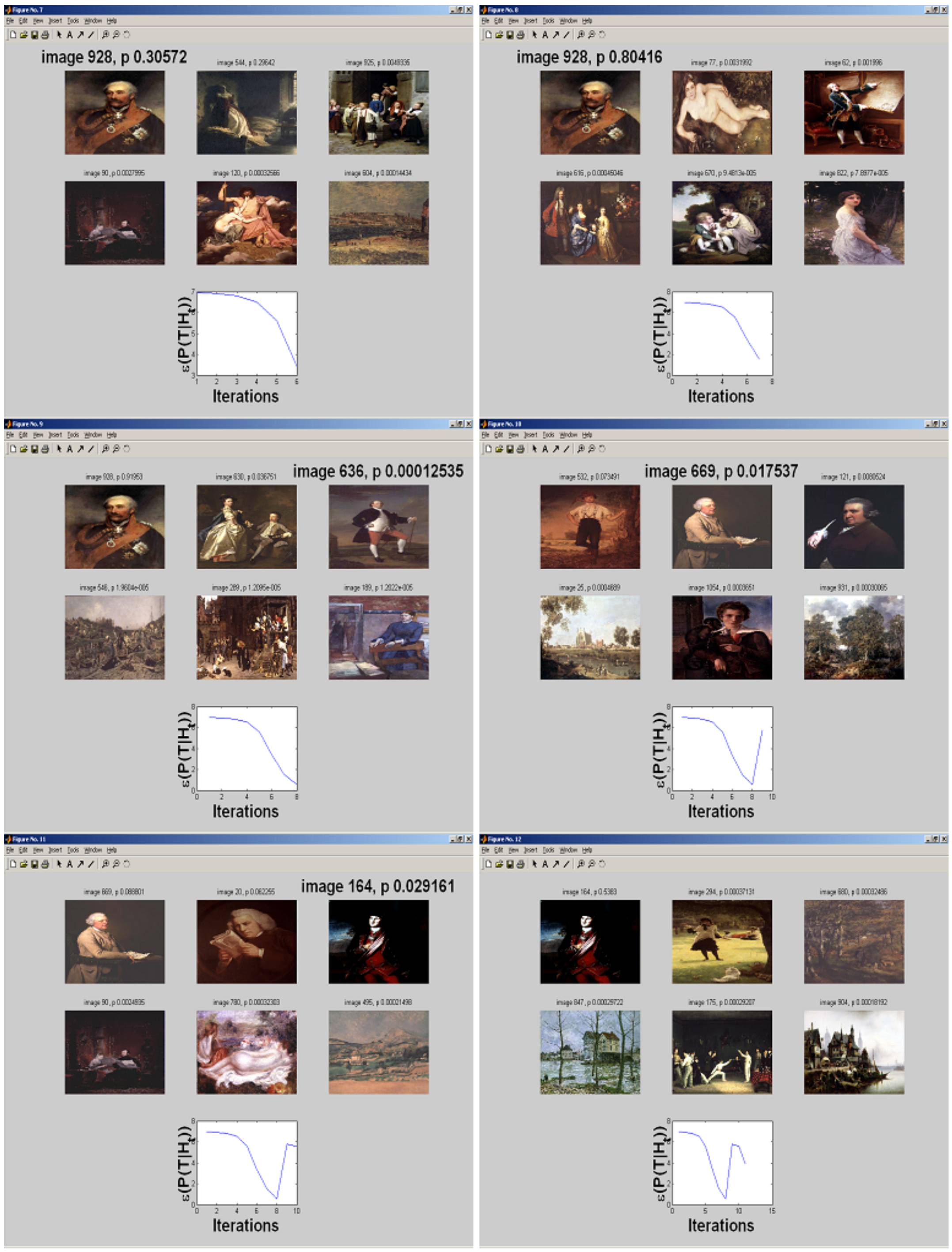


Figure 6.13: The next 6 iterations using the entropy utility. Images 928 (top left window), 928 (top right window), 636 (middle left window), 669 (middle right window), 164 (bottom left window), 164 (bottom right window) are selected.

Iteration	α_t	$(1-\alpha_t)$
1	–	–
2	0.1	0.9
3	0.2	0.8
4	0.3	0.7
5	0.4	0.6
6	0.5	0.5
7	0.6	0.4
8	0.7	0.3
9	0.8	0.2
10	0.9	0.1
11	0.99	0.01
12	0.99	0.01

Table 6.5: The values of weights in the combined utility for the example of Figures 6.14, 6.15.

System	$\varepsilon(T)$
	(over 12 iterations)
Variance Utility	6.75
Entropy Utility	4.0
Combined Utility	1.0

Table 6.6: Entropy's values for the three systems over 12 iterations.

combined utility system has the smallest entropy comparing the other two systems. Its entropy value reach the lowest when all the six retrieved images belong to the same image class (portraits). Of course in all systems we can see these jumps that happened in entropy graph during the search. This happens because either the user disturbs the learning process by picking an image completely irrelevant to his previous selections or because a "new" image comes up in the retrieved display set.

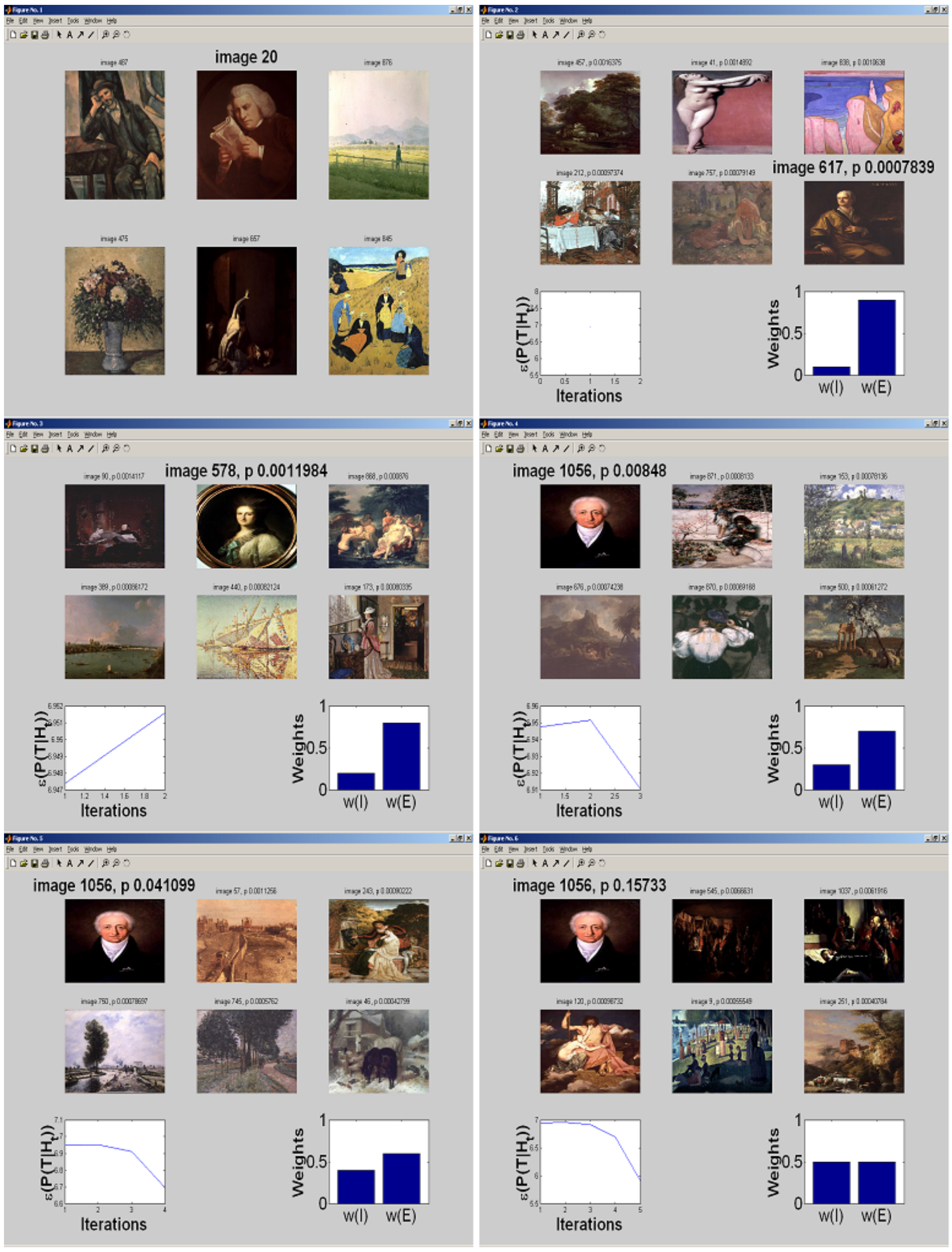


Figure 6.14: The first 6 iterations using the combined utility. Images 20 (top left window), 617 (top right window), 578 (middle left window), 1056 (middle right window), 1056 (bottom left window), 1056 (bottom right window) are selected.

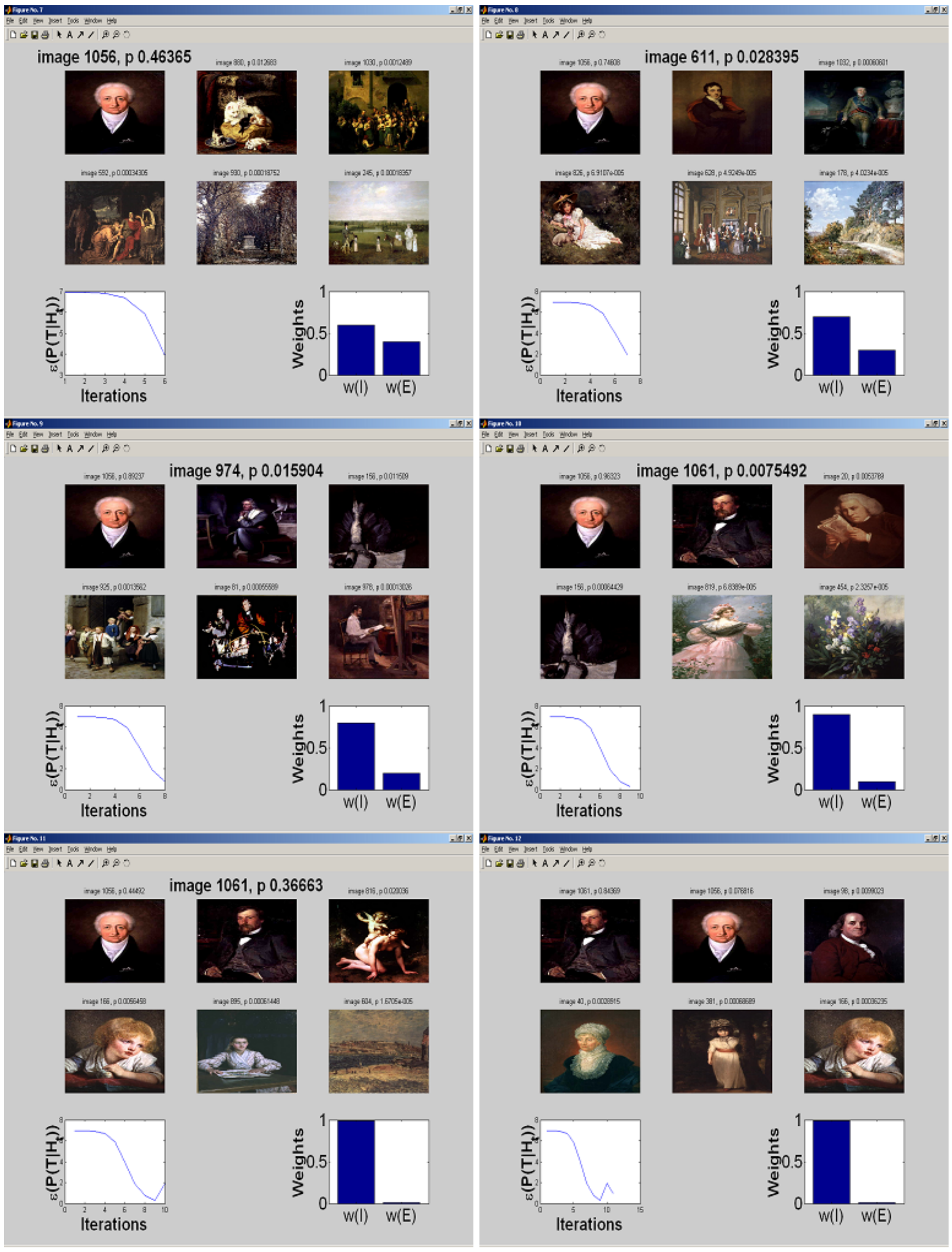


Figure 6.15: The next 6 iterations using the combined utility. Images 1056 (top left window), 611 (top right window), 974 (middle left window), 1061 (middle right window), 1061 (bottom left window), 98 (bottom right window) are selected.

6.7 Future Extensions

As we mentioned at the start of this chapter, a designer will start looking for a certain class of images but, while browsing through a database, he may refocus his choice, or maybe completely change his mind. This may be due to a perceived deficiency in the database or because the act of exploring the database unveiled a new possibility not thought of until that moment. We believe that our proposal helps this process as well as forms the basis for an extended CBIR system where a 2 dimensional search is possible. In Figure 6.16 we show an interface of such a system. On the left side we have an open-ended browser(variance utility or entropy utility) where an expert needs to explore the database to see what it has to offer and to look for new ideas. Once he finds something that looks interesting to him he might go for the most-probable scheme at the right side.

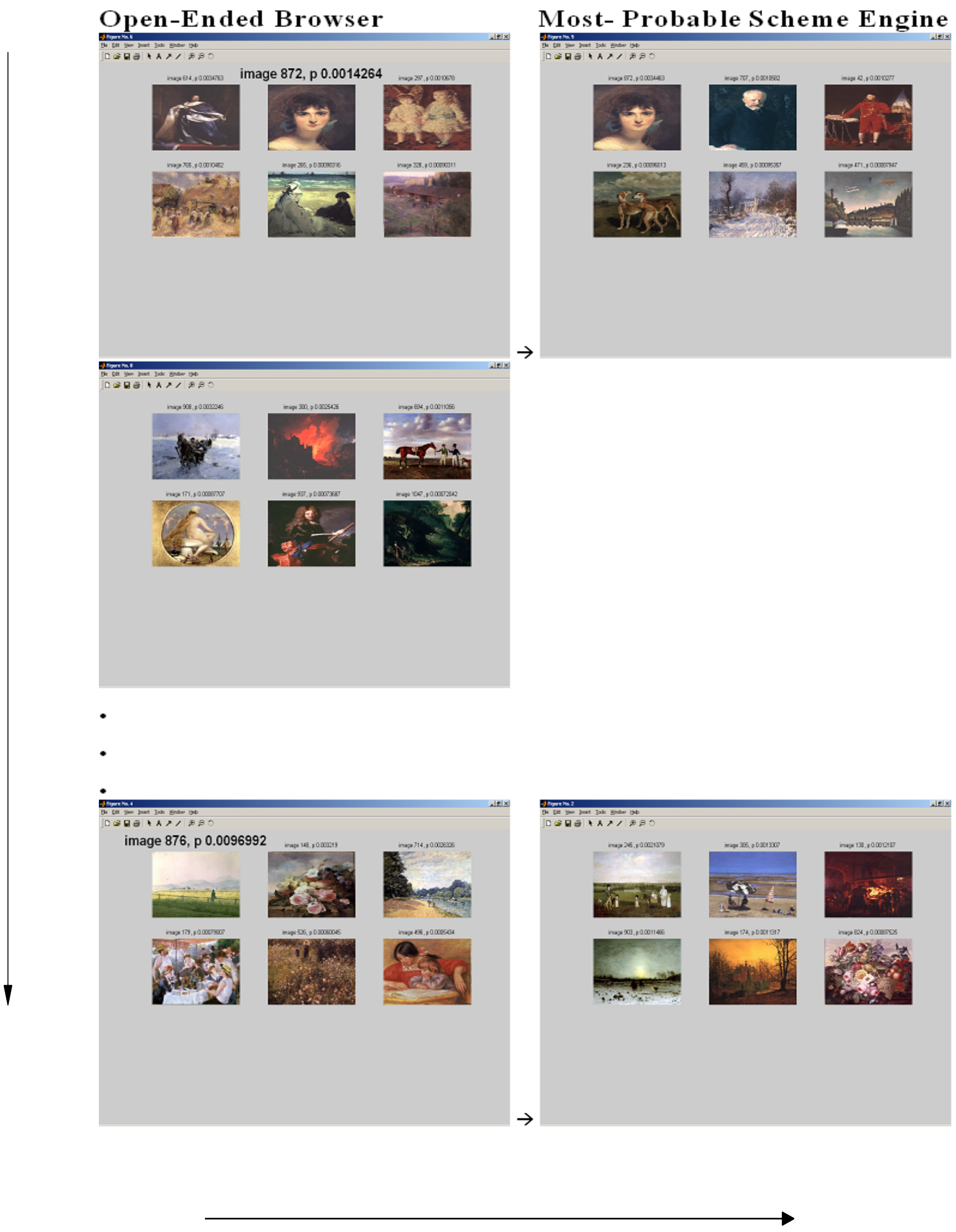


Figure 6.16: A 2 dimensional search using the benefits of open-ended browsing and the most probable scheme.

Chapter 7

Retrieval Evaluation and Emotional Queries

7.1 Introduction

The first part of this chapter deals with image retrieval evaluation. Once a content-based image retrieval application has been developed, the next problem is how to evaluate its performance. In the literature, retrieval performance is often ignored or restricted simply to printing out the results of one or more example queries which are easily tailored to give a positive impression. Statistical analysis of evaluation results is not done or is very basic e.g. computation of means. Some other papers either use performance measures borrowed from text retrieval (such as those used in the Text REtrieval Conference, TREC), or propose a set of performance measures similar to those used in TREC; see Muller et al. (2001).

Here we apply the methods of target testing, using standard ANOVA methods to compare different CBIR systems. We also investigate the feasibility of using visual browsing for the retrieval of emotional content of BAL pictures using our extended system that evaluates $P(T, \sigma, F|H_t)$. We focus on the question of whether the posterior distribution provides useful information that, when used as the prior in a subsequent search, shortens the search time.

7.2 Evaluation based on Ground Truth: Precision and Recall

Most evaluation is done on image databases that consist of images pre-divided into classes by subject e.g. automobiles, clouds etc. This division is done manually and is known as ground truth. Query images are presented one by one to the search engine of the CBIR application and the retrieved results are then compared to the ground truth of the corresponding query image. A successful retrieval is one where the retrieved image and the query image are from the same class. For systems without relevance feedback, the retrieval comes from one iteration of the retrieval algorithm. The performance of the system is then described by the precision and recall where,

$$\text{Recall}(n) = \frac{\text{Number of relevant images retrieved in top } n \text{ images.}}{\text{Number of relevant images in the database.}}$$

$$\text{Precision}(n) = \frac{\text{Number of relevant images retrieved in top } n \text{ images.}}{\text{Number of retrieved images.}}$$

Several different methods can be applied here to compare the ground truth and the actual retrieved images. A single value is computed from the two sets telling us how well the query was retrieved by the system. Examples are: rank of the best match, average rank of relevant images, precision, recall, target testing, error rate, retrieval efficiency, correct and incorrect detection.

A graph can be used to illustrate the relation of two of the above values, for example the precision vs. recall graph, the precision vs. number of retrieved images, recall vs. number of retrieved images graph or retrieval accuracy vs. noise graph, for more details see Muller et al. (2001), Smith (1998) and Liu et al. (2001). For systems with relevance feedback, we can run the system until an image from the same class appears, and count the number of iterations that this takes.

7.3 Problems in CBIR Performance Evaluation

There are two problems that evaluation must overcome:

1. Define a common image collection. A common way of constructing an image database for CBIR evaluation is to use the Corel photo CDs, each of which usually contains 100 broadly similar images. Most research groups use only a subset of the collection, and this can result in a collection of several highly dissimilar groups of images, with relatively high within-group similarity. This can lead to great apparent improvement in retrieval performance: e.g. it is not too hard to distinguish sunsets from underwater images of fish. Another commonly used database is the VisTex database at MIT Media Lab, which contains more than 400 primarily texture images. Some other candidates includes the standard collection of colour images from MPEG (Moving Picture Experts Group). Other commonly used databases are University of Washington at `http://www.cs.washington.edu/research/imagedatabase/groundtruth/` and the Benchathlon collection at `http://www.benchathlon.net/img/done/`; see Muller et al. (2001).

   One of the problems in creating such an image collection is that the size of the database should be large enough, and the images should have enough diversity in different domains. For text-based retrieval, it is quite normal to have millions of documents (TREC, 2002) whereas in CBIR most systems work with only few thousand images, some even with fewer. Ways to get a huge collection of images include collecting them from the internet and sampling image frames from TV channels.

2. Obtain relevance judgments. The collection of real user judgements is time-consuming and only the user knows what he or she expects as a retrieval result of a given query image. Experiments show that user judgements for the same image often differ, see Yu et al. (2003).

3. As we mentioned earlier, most image databases, including the BAL database, are not supplied with ground truth data. Hence, objective evaluation becomes difficult to obtain. In the absence of any objective measure, visual inspection of results is often used. This is hard to quantify, especially in published papers, as only a limited number of retrieval scenarios are usually provided. Other approaches have concentrated on methods to generate ground truth data through the use of multiple human observers. However, it is difficult to generate accurate ground truth as an observer must evaluate all images in the database given a particular reference image. A user will normally give a very hard relevance judgment for an image given a query concept. However, people are always not so confident with their preferences for art images. Experiments were conducted by Yu et al. (2003) observed the following: re-presenting art images to some users who marked the same images two days ago, they then observed that many of them changed their opinions for some of images. This process must be repeated for

multiple reference images. Such a task is not suitable for large image databases, as it is labour intensive, and judgements of relevancy are context and observer sensitive.

An alternative approach to evaluation is taken here, which is based on the concept of Target Testing. This method avoids the need for ground truth and is suited to relevance feedback based systems.

7.3.1 Target Testing with the BAL database

In target testing, an image from the database is displayed and then the user employs the CBIR system to find it. The number of iterations until the image is found is then recorded. If the image is not discovered by 60 iterations, the search is abandoned as being unsuccessful and is recorded as a missing value.

We tested this algorithm against the original system of Cox et al. (2000), which only conducted inference on T. In that system, distance was defined on all features together, rather than by subset, and σ was fixed.

We used a database of $N = 1066$ paintings supplied to us by the Bridgman Art Library, London, with $N_D = 9$ images displayed per iteration. Eight participants recruited from the Statistics Department volunteered to look for 15 images. We recorded the number of iterations taken by each person to find each image, with the original system and with the system described in Section 5.4. The results of all the tests are shown in Table 7.1. Figure 7.1 graphs the results for the eight human subjects, showing the percentage of searches that were completed successfully within a given number of iterations for the two systems. The red line indicates the corresponding percentages that would be expected if the database were searched at random. The results from the eight participants clearly show that both systems make a substantial improvement

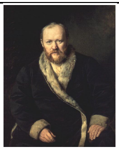
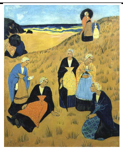
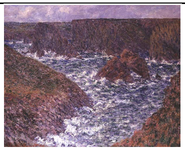
BAL Image					
System 1 System 2	9.1 7.6	23.5 26.2	25.1 25	37.2 28	12 15.8
BAL Image					
System 1 System 2	8.8 14.1	11.3 14	21.8 29.4	14.7 17.6	14.8 15.2
BAL Image					
System 1 System 2	12 16.7	7.7 6.1	24.1 16.6	9.8 16.5	12.8 12.2

Table 7.1: The average number of iterations of the successful Target- testing procedure carried out over 15 images.

over a random search.

Table 7.2 shows the results of an ANOVA analysis of the number of iterations (successful and unsuccessful searches) taken as a function of the 3 factors: system, image and person and their interactions. The missing cells (unsuccessful searches) are replaced by 60 iterations. An analysis based only on the successful searches may cause substantial bias in the results. Also, survival analysis could be applied in such circumstances.

Interpreting the results of the ANOVA table (Table 7.2) we found that image is the most significant main effect. System is also significant (System 1 is better than System 2). There is significant interaction between person and image e.g. different people find different images in more or less iterations. All other effects are not significant.

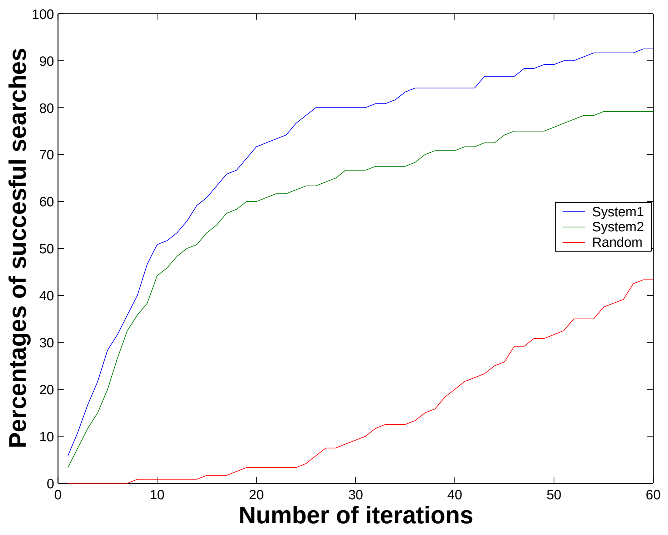


Figure 7.1: Percentage of succesful searches as a function of search length for the simple pichunter (System 1), our extended version (System 2) of it and simulations by random search.

We also observe that the mean number of iterations to find an image is 16.35 using the system 1 and 17.43 using the system 2, which is significantly less than the number expected by simply randomly displaying N_D images from n without replacement, in this case $\frac{1}{p} \times \frac{1}{2} = 59.2$, where $p = \frac{9}{1066}$.

Our system failed to find more images than the original PicHunter. We concluded that the problem lies with the relatively small amount of information that the data give us. We have defined the simplest possible model; one image is selected from the display set. More complicated selection procedures, involving more than one image, perhaps ordered by preference, and identifying objects in the image, would yield more information and perhaps allow us to exploit our more general model.

Factor	SS	df	p-value for F test
System	2.4897e+003	1	0.0019
Person	1.9852e+003	7	0.3305
Image	2.8668e+004	14	1.3896e-011
System*Person	1.4754e+003	7	0.5369
System*Image	4.0542e+003	14	0.2962
Person*Image	3.4591e+004	98	0.0340
Error	2.3876e+004	98	
Total	9.7140e+004	239	

Table 7.2: Analysis of variance for number of iterations in successful and unsuccessful searches to find images as a function of system, image and person and their interaction effects.

7.4 Retrieving Emotional Content of BAL pictures

In Art Image Retrieval there is a gap between low-level visual features (e.g colour and texture) and the higher-level abstract properties, (e.g. painting styles and expressed feelings or emotions). In some sense, this gap might be more important for AIR since the higher-level abstract properties are always more indicative of the expression of art images (especially for some modern art images); see Yu et al. (2003). This section explores how our extended system, which computes the posterior probability of T, σ and F, behaves with issues related to the retrieval of emotional content of pictures. Particular we concentrate on the joint marginal posterior of F and σ as a simple description of how a user is conducting a search for a particular emotional content and how the feature space is able to describe that content. We do not use information on T, as this may depend significantly on the initial images displayed; we concentrate on the more general information about the query that is given by F and σ. This research begins without any previously known characteristics or variables that are related to the retrieval of emotional content of pictures. This study is an attempt to elicit fundamental information about the characteristics associated with the search of the emotional content of images. This study is the first of its kind to our knowledge; therefore it is an exploratory study, see Lee (2001).

7.4.1 Experiment Design

Two participants were employed for this experiment. They had been asked to search for three types of emotional queries. We try to define these higher abstract properties as follows: a "romantic" image will include a couple as an important theme in the image, a "violent" image will be an image including one of fighting, wars, death or murders and a "sad" image will be one with a sad or depressing theme, including sombre landscapes. The experiment consist of two stages. At the first stage (stage A) the subject was asked to search for the given query using the following model:

$$\begin{aligned} P(T,\sigma,F|H_t) &\propto \left(\prod_{k=1}^{t} P(A_k|D_k,T,\sigma,F)\right) P(T)P(\sigma)P(F) \\ &\propto \prod_{k=1}^{t} P(A_k|D_k,T,\sigma,F); \end{aligned} \tag{7.1}$$

that is a uniform prior distribution on F and σ. He was asked to repeat the search 10 times. He was advised to stop directly the search when he found an image relevant (according to him) to the emotional query. The number of iterations taken was recorded. At every trial, we store his joint marginal of F and σ:

$$P(F,\sigma|H_t) = \sum_{i=1}^{N} P(T_i,\sigma,F|H_t). \tag{7.2}$$

We consider that some information on the emotional content of the image is quantified within this joint marginal. When the 10 trials are finished, we have already stored 10 joint marginals $P(F,\sigma)$. By multiplying them we summarize all this information in a large joint marginal $P(F, \sigma | H_{t_1}, ..., H_{t_{10}})$ where H_{t_k} is the data from the k^{th} trial, where

$$P(F, \sigma | H_{t_1}, ..., H_{t_{10}}) \propto \sum_T \prod_{s=1}^{10} \left(\prod_{k=1}^{t_s} P(A_{k_s} | D_{k_s}, T, \sigma, F) \right) \times P(T)P(\sigma)P(F). \quad (7.3)$$

At the second stage (stage B) the user was asked to search for the same set of queries (romantic, violent, sad) but now using the following model:

$$\begin{aligned} P(T, \sigma, F | H_t) &\propto \left(\prod_{k=1}^{t} P(A_k | D_k, T, \sigma, F) \right) P(T)P(F, \sigma | H_{t_1}, ..., H_{t_{10}}) \\ &\propto \left(\prod_{k=1}^{t} P(A_k | D_k, T, \sigma, F) \right) P(F, \sigma | H_{t_1}, ..., H_{t_{10}}). \end{aligned} \quad (7.4)$$

that is, using the posterior distribution of F and σ as the prior in the new search. Again, 10 trials for each of the 3 emotional queries were conducted. The number of iterations to find a relevant image was recorded. To attempt to reduce any effect in the order of these two sets of searches, stage B was conducted several weeks after stage A.

7.4.2 Results

The following subsections present the results and their statistical analysis for each user's run as well as for a user's interaction run. Tables 7.3, 7.4, 7.5 and 7.6 presents the results of this experiment for the two users, where the Table 7.13 shows the results for User 1 for Stage B of the experiment using User 2 - Stage A as a prior information.

Statistical Analysis for User 1

Figures 7.2, 7.3 and 7.4 show the posterior of F and σ at the end of stage A and stage B. We can see that after stage B $P(F, \sigma)$ is concentrated over a smaller region of the F and σ plane. The behaviour of $P(F, \sigma)$ reflects the user's actions of what he consider about a romantic or violent or a sad image and each display set that each

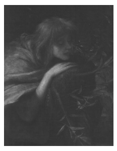
USER 1 - STAGE A					
ROMANTIC		VIOLENT		SAD	
Images	Iterations	Images	Iterations	Images	Iterations
	6		2		3
	6		4		4
	7		13		2
	8		17		3
	2		2		7
	6		2		2
	16		2		3
	5		2		9
	18		2		3
	20		2		2
Mean ($\tilde{x}$)	9.4	Mean ($\tilde{x}$)	4.8	Mean ($\tilde{x}$)	3.8
Std (s)	6.2	Std (s)	5.4	Std (s)	2.3

Table 7.3: The results for user 1 for stage A of the experiment. The images as well as the number of iterations conducted to find each image are recorded.

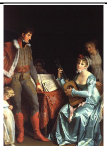
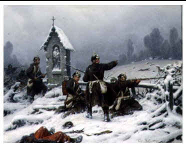
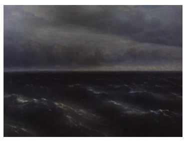
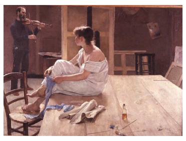
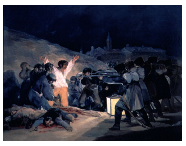
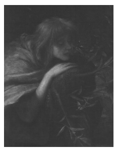
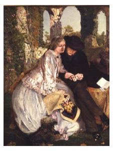
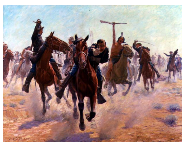
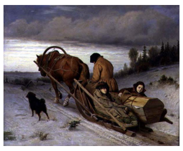
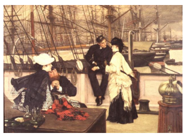
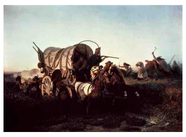
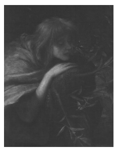
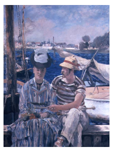
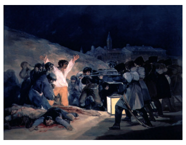
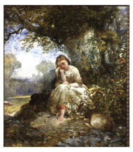
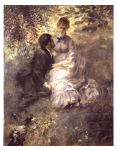
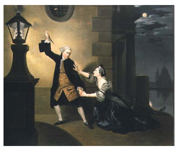
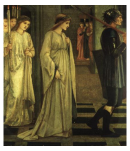
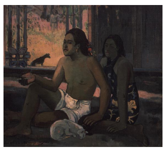
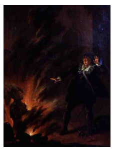
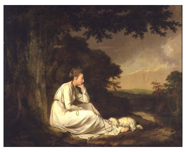
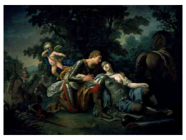
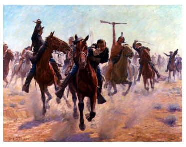
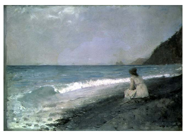
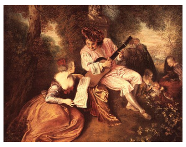
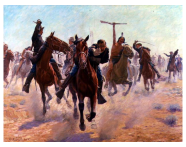
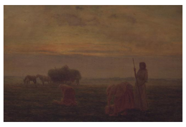
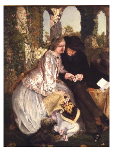
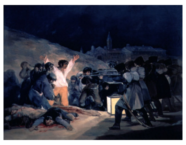
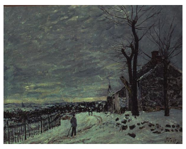
USER 1 - STAGE B					
ROMANTIC		VIOLENT		SAD	
Images	Iterations	Images	Iterations	Images	Iterations
	3		14		5
	3		13		2
	2		6		3
	16		2		4
	2		7		2
	3		2		2
	3		2		4
	2		3		6
	2		5		3
	2		2		4
Mean ($\tilde{x}$)	3.8	Mean ($\tilde{x}$)	5.6	Mean ($\tilde{x}$)	3.5
Std (s)	4.3	Std (s)	4.5	Std (s)	1.3

Table 7.4: The results for user 1 for stage B of the experiment. The images as well as the number of iterations conducted to find each image are recorded.

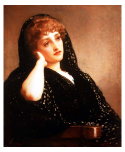
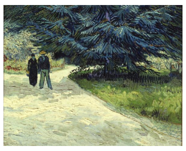
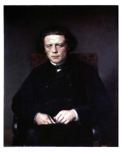
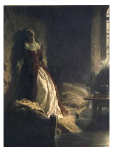
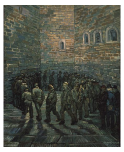
USER 2 - STAGE A					
ROMANTIC		VIOLENT		SAD	
Images	Iterations	Images	Iterations	Images	Iterations
	2		5		5
	6		4		7
	16		2		12
	4		10		12
	3		4		3
	21		12		7
	2		2		5
	5		10		6
	4		5		4
	11		5		5
Mean ($\tilde{x}$) Std (s)	7.4 6.5	Mean ($\tilde{x}$) Std (s)	5.9 3.5	Mean ($\tilde{x}$) Std (s)	6.6 3.1

Table 7.5: The results for user 2 for stage A of the experiment. The images as well as the number of iterations conducted to find each image are recorded.

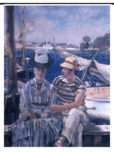
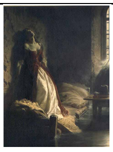
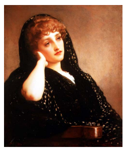
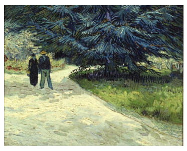
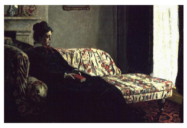
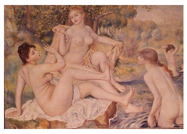
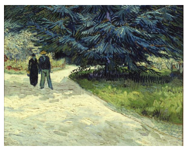
USER 2 - STAGE B					
ROMANTIC		VIOLENT		SAD	
Images	Iterations	Images	Iterations	Images	Iterations
	2		2		2
	6		3		9
	3		4		5
	2		5		6
	2		11		2
	8		10		5
	4		2		5
	2		2		7
	7		5		3
	5		5		3
Mean ($\tilde{x}$) Std (s)	4.1 2.3	Mean ($\tilde{x}$) Std (s)	4.9 3.2	Mean ($\tilde{x}$) Std (s)	4.7 2.3

Table 7.6: The results for user 2 for stage B of the experiment. The images as well as the number of iterations conducted to find each image are recorded.

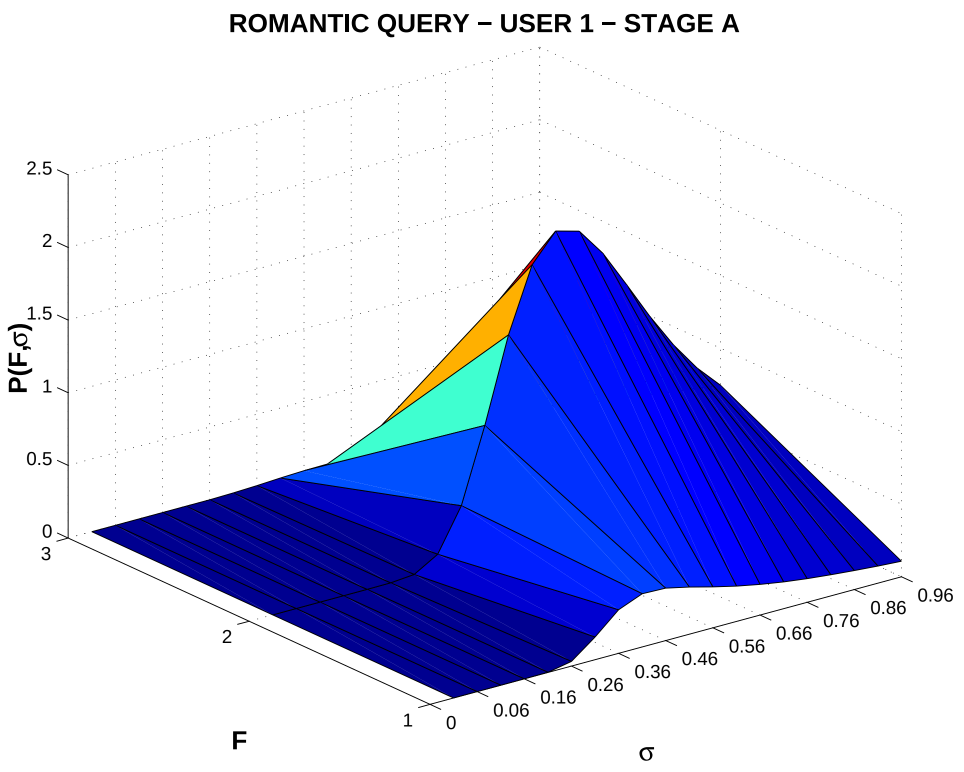
USER 1			
Hypothesis	ROMANTIC	VIOLENT	SAD
H_0 vs. H_1	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$
p-value	0.0153	0.6365	0.3651
Results	REJECT H_0	REJECT H_1	REJECT H_1

Table 7.7: The results for three hypothesis tests (2-sample t-tests) for user 1, where μ is the mean number of the iterations for each emotional query search.

time appears. We can see differences among these three posteriors, thus $P(F, \sigma)$ does distinguish between the different emotional queries. Also, we observe that the posterior distributions after each stage are consistent; that after stage B is more peaked version of that after stage A.

We continue the statistical analysis applying two sample t-tests for each emotional query for the user 1 given the data (number of iterations) of Tables 7.3 and 7.4. Table 7.7 summarizes the results of the three two sample t-tests for each emotional query. There is a statistically significant difference only for the "romantic" query. We conclude that the use of the joint marginal of F and σ as a prior helped the user to find "romantic" images, spending less iterations than stage A of the experiment.

Finally, we did two-way Analysis of Variance to determine the effect of query and the two stages of the experiment on the number of iterations to find an image. The standard ANOVA table is shown in Table 7.8. The p-value for the emotional query effect is 0.1136. This is an indication that the iterations do not vary from one emotional query to another. The p-value for the stage of experiment effect is 0.139 which is not significant. This indicates that one stage of experiment is not out-performing the other in the number of iterations. Finally, a stage of experiment might help us to find an



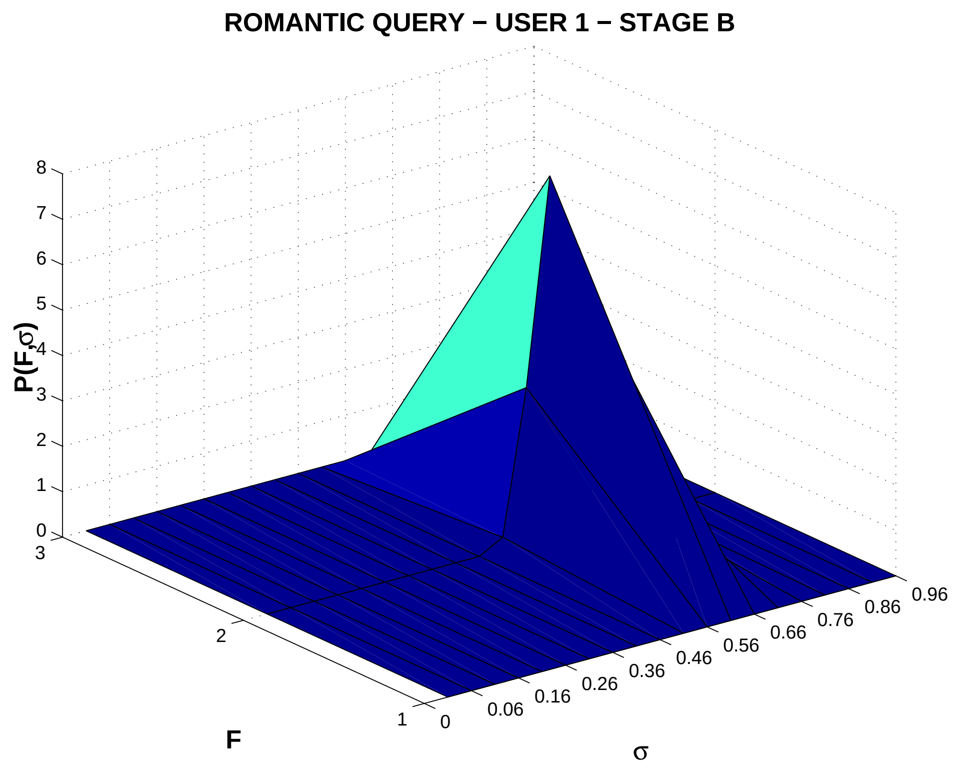
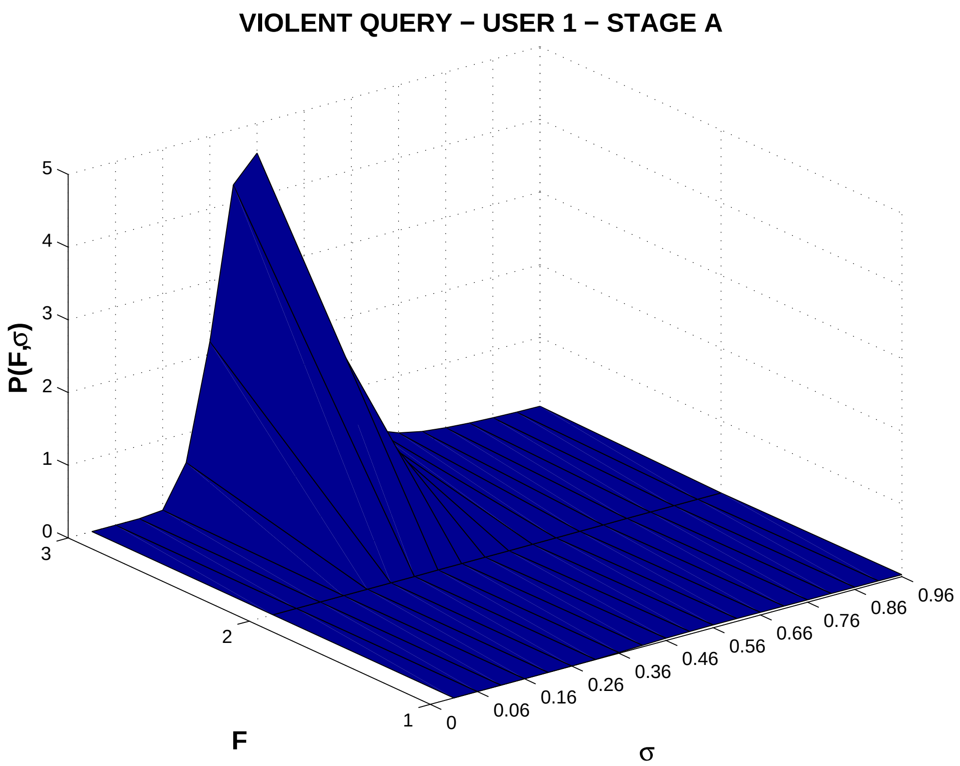


Figure 7.2: The joint marginal posterior probability of F and σ, $P(F, \sigma|H_{t_1}, ..., H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a "romantic query", where $F = 1, 2, 3$ and 1 indicates the global colour features, 2 indicates the textural features and 3 indicates the segmentation features, and $\sigma \in (0, 1)$.



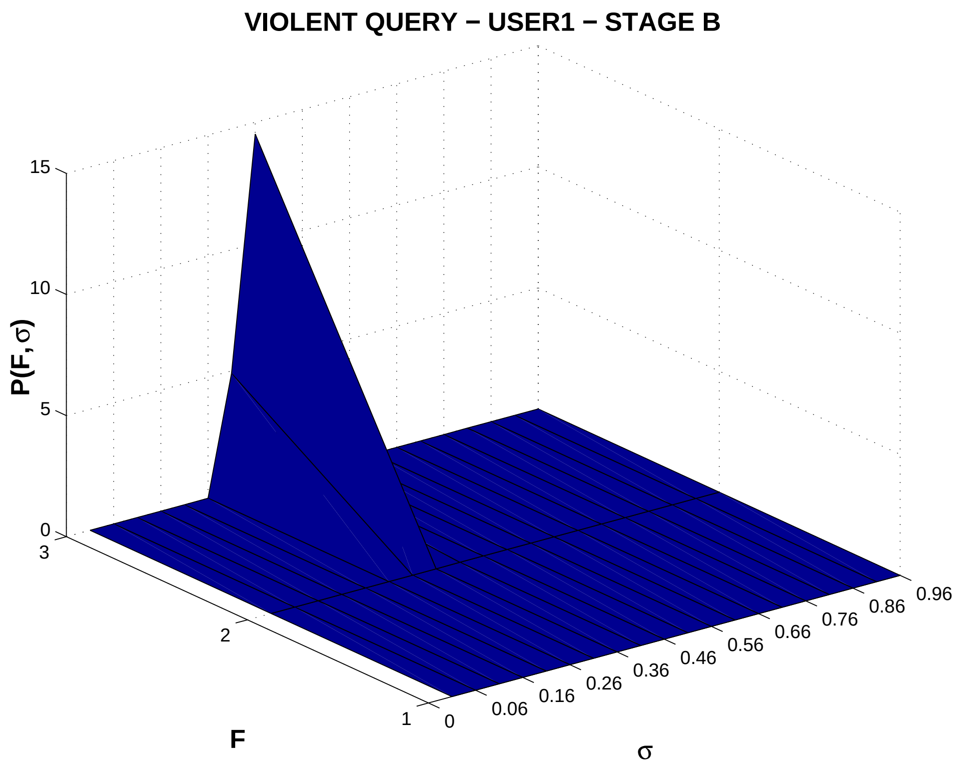
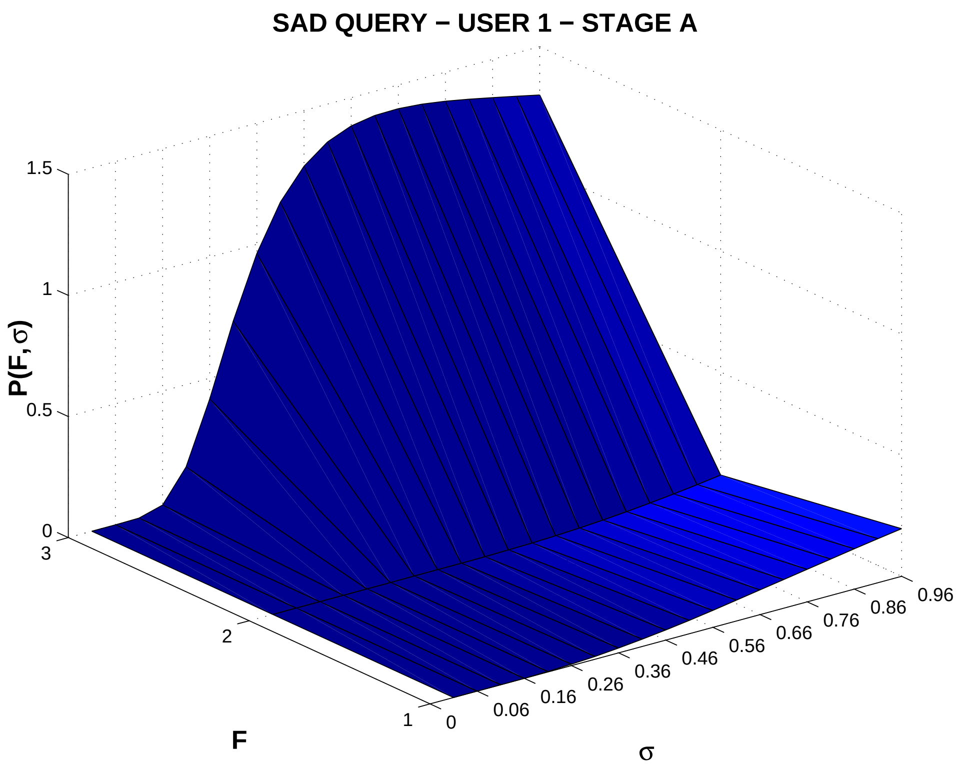


Figure 7.3: The joint marginal posterior probability of F and σ, $P(F, \sigma | H_{t_1}, ..., H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a "violent query".



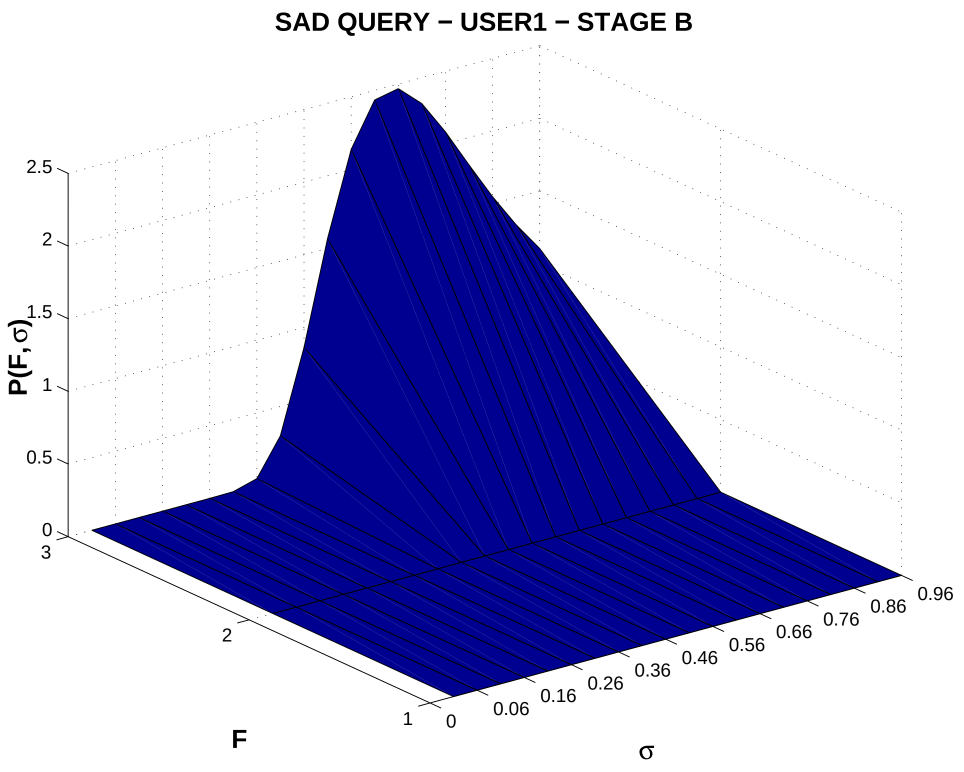


Figure 7.4: The joint marginal posterior probability of F and σ, $P(F, \sigma|H_{t_1}, ..., H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a “sad query”.

Factor	SS	df	p-value for F test
Emotional query	87.1	2	0.1136
Stage of experiment	43.35	1	0.139
Interaction	117.1	2	0.0558
Error	1038.1	54	

Table 7.8: Two-way analysis of variance to determine the effect of emotional query the two stages of the experiment and their interaction on the number of iterations to find an image.

image in less iterations in one emotional query, but not be different from the other stage of experiment for other emotional queries. This effect is called an *interaction*. It is impossible to detect an iteraction unless there are duplicate observations for some combination of experiment's stage and emotional query. There does appear to be an interaction between stages of experiment and emotional queries, because the value 0.0558 is close to critical p-value 0.05.

Looking at the residuals from fitting the ANOVA model (Figure 7.5) we can see they are not normal. We have tried several transformations, such as the logarithmic and square root but they did not work. A transformation that gave us reasonable results was the power transformation (1/X). Running now again the analysis of variance for the transformed data we get the standard ANOVA table in Table 7.9, where there does appear to be an interaction between stages of experiment and emotional queries. This significant interaction confirms the result of Table 7.7 where searching for a romantic query the stage B of the experiment (use of prior information) helps one to find a romantic image in fewer iterations. Looking again on the normal probability plot for the residuals we can see that now they are much better behaved.

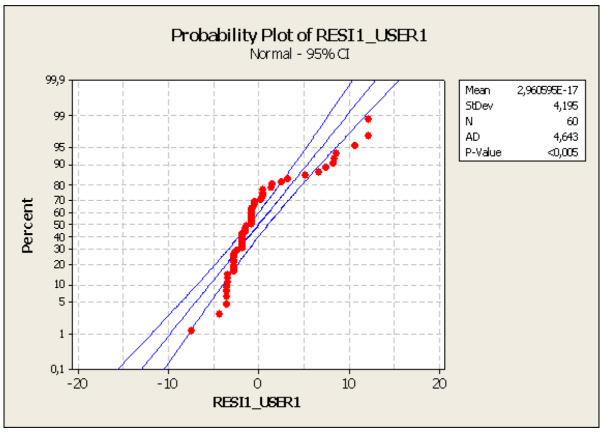


Figure 7.5: Normal probability plot for the residuals of ANOVA Table 7.8 of User 1.

Factor	SS	df	p-value for F test
Emotional query	0.05102	2	0.349
Stage of experiment	0.02866	1	0.277
Interaction	0.26685	2	**0.006**
Error	1.28481	54	

Table 7.9: Two-way analysis of variance to determine the effect of emotional query, the two stages of the experiment and their interaction on transformed number of iterations finding an image.

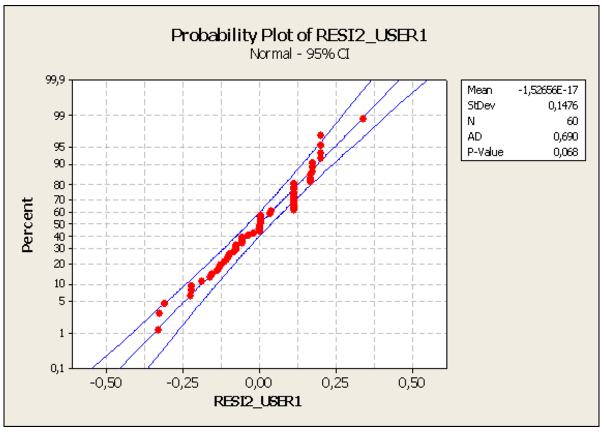


Figure 7.6: Normal probability plot for the residuals of ANOVA Table 7.9 of User 1.

USER 2			
Hypothesis	ROMANTIC	VIOLENT	SAD
H_0 vs. H_1	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$	$\mu_{stageA} = \mu_{stageB}$ vs. $\mu_{stageA} > \mu_{stageB}$
p-value	0.0736	0.2573	0.0673
Results	REJECT H_1	REJECT H_1	REJECT H_1

Table 7.10: The results for three hypothesis testing for the user 2, where μ is the mean number of the iterations conducted for each emotional query search.

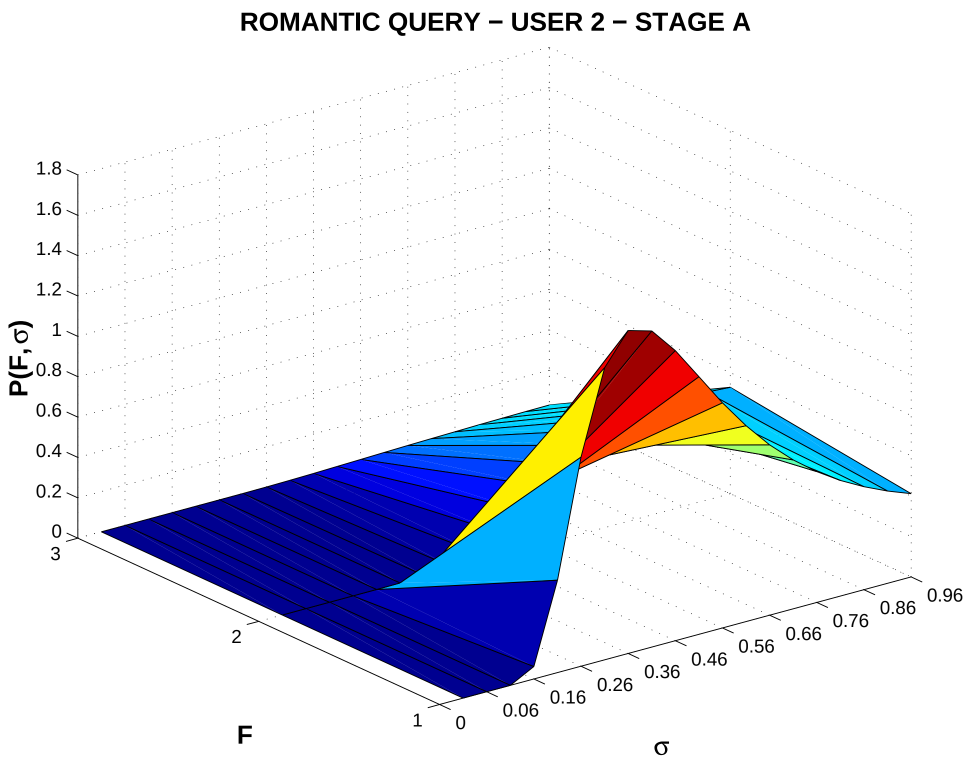
Statistical Analysis for User 2

Figures 7.7, 7.8 and 7.9 show the $P(F, \sigma | H_{t_1}, ..., H_{t_10})$ at the end of stage A of the experiment and at the end of stage B for user 2. We can see that there is a difference with those of user 1. However, even for user 2, the three posteriors are still different, thus $P(F, \sigma | H_{t_1}, ..., H_{t_10})$ does distinguish between the different emotional queries. Thus it appears that the posterior of F is consistent for each user and query, but in particular different users have distinct posteriors for the same query.

We continue the statistical analysis, applying two sample t-tests for each emotional query for user 2 given to the data of Tables 7.5 and 7.6. Table 7.10 summarizes the results of the three two sample t-tests for each emotional query.

There is no statistically significant difference for each query, between the stage A and stage B of the experiment, although both "Romantic" and "Sad" are close to being significant.

Finally, we implemented a two-way Analysis of Variance to determine the effect of emotional query, experiment stage and their interaction on the number of iterations to find an image for the user 2. The standard ANOVA table is shown in Table 7.11. The p-value for the emotional query effect is 0.9551. This is an indication that the



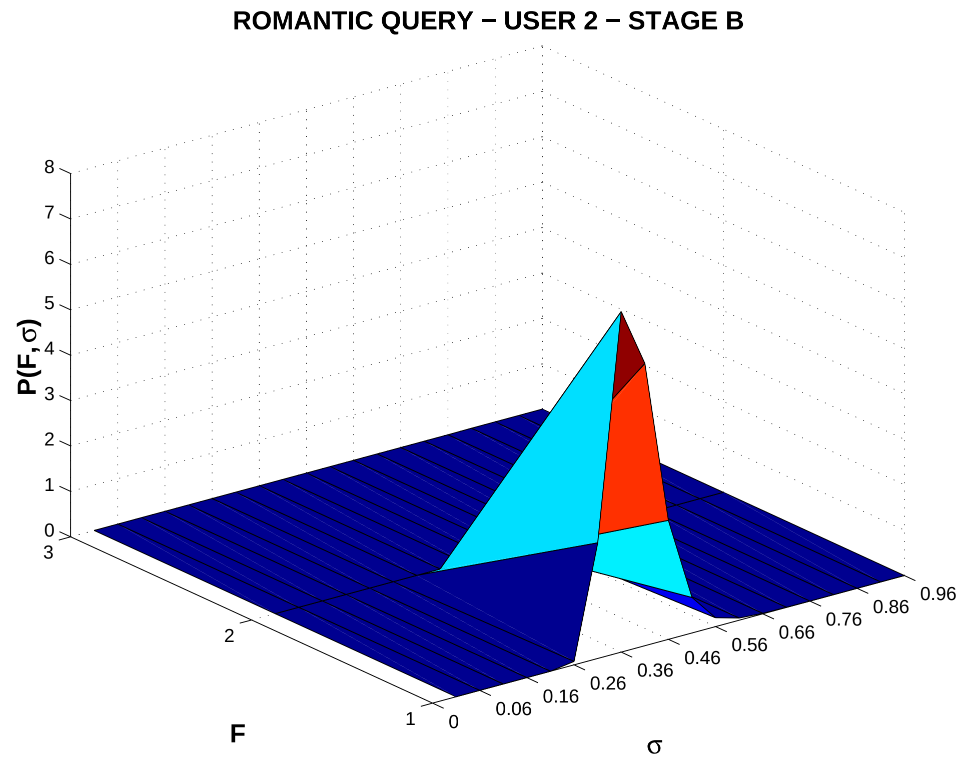
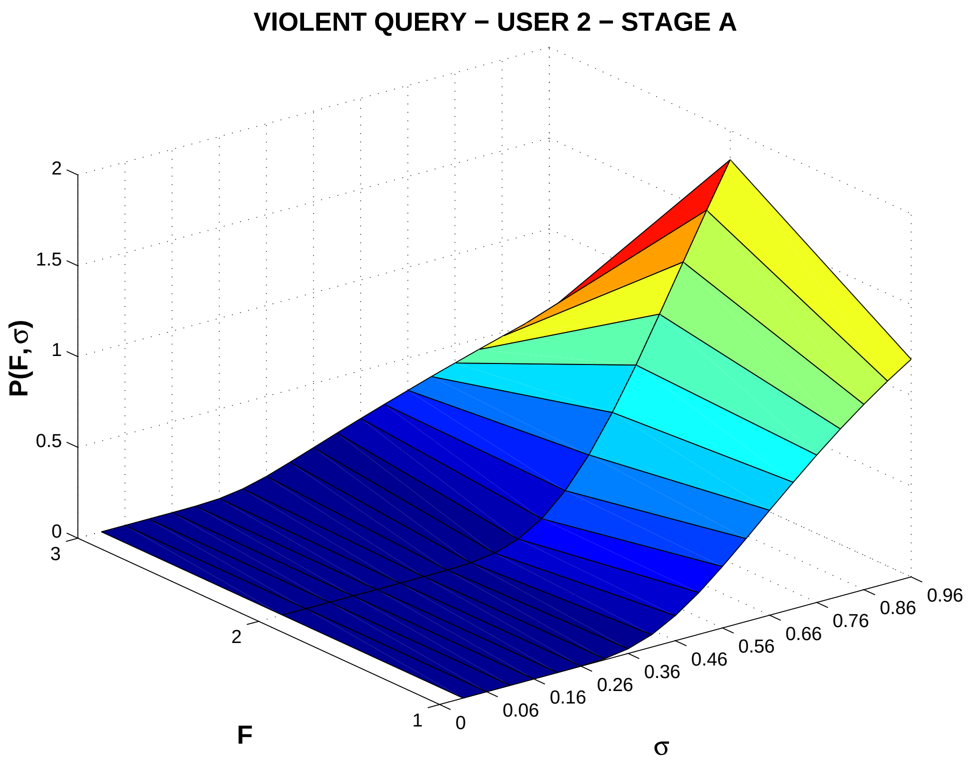


Figure 7.7: The joint marginal posterior probability of F and σ, $P(F,\sigma|H_{t_1},...,H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a "romantic query".



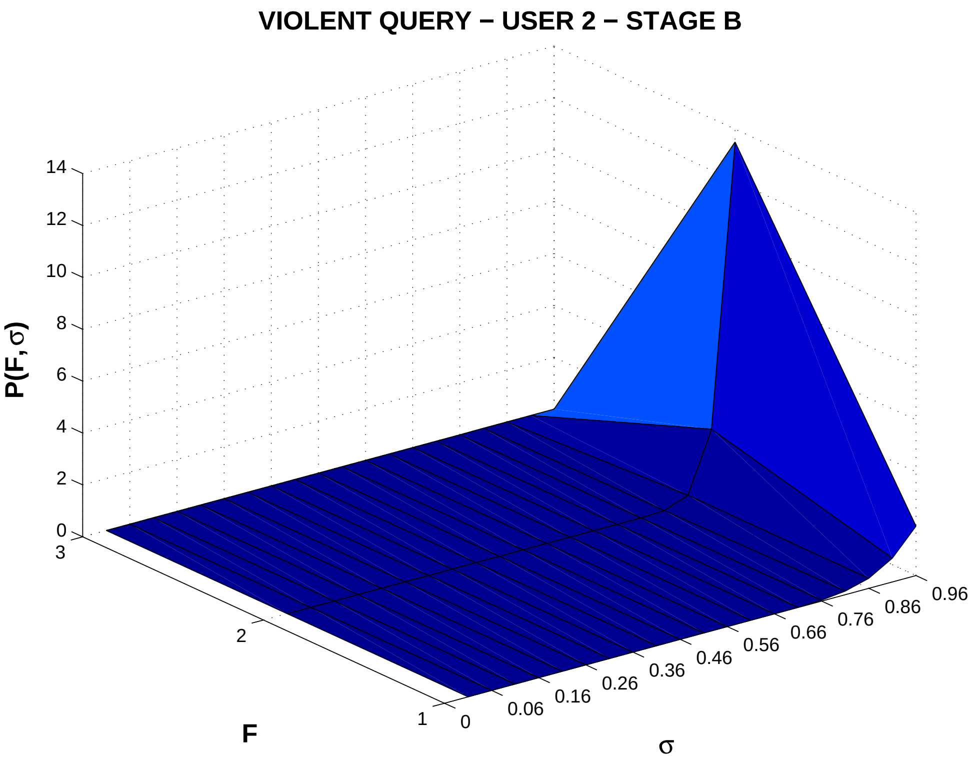
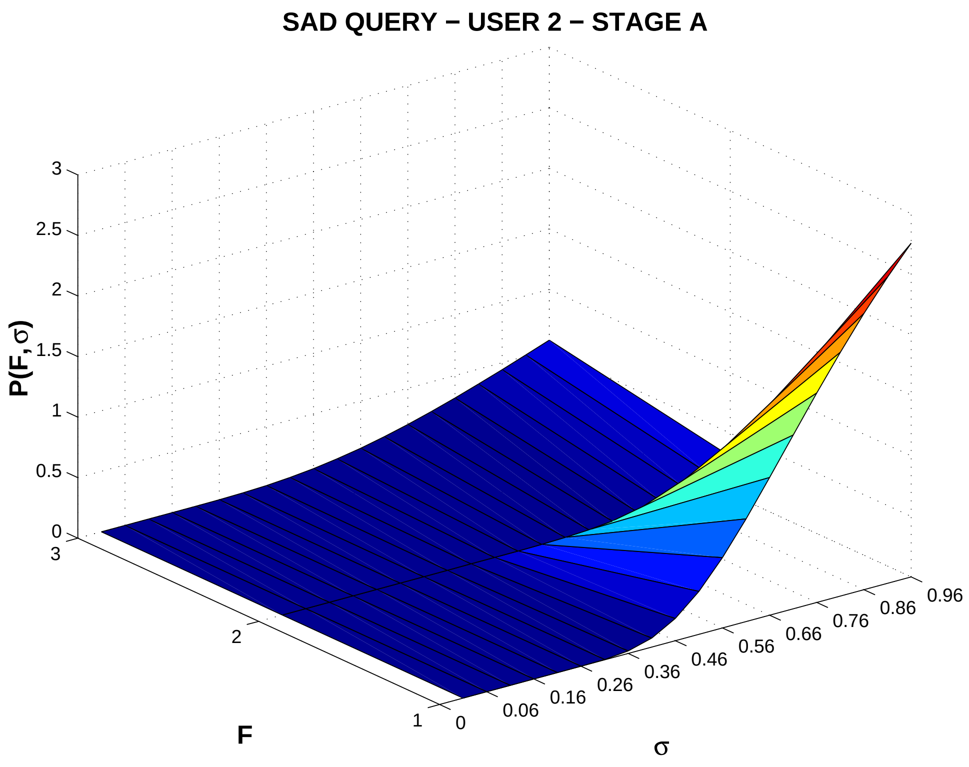


Figure 7.8: The joint marginal posterior probability of F and σ, $P(F, \sigma|H_{t_1}, ..., H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a "violent query".



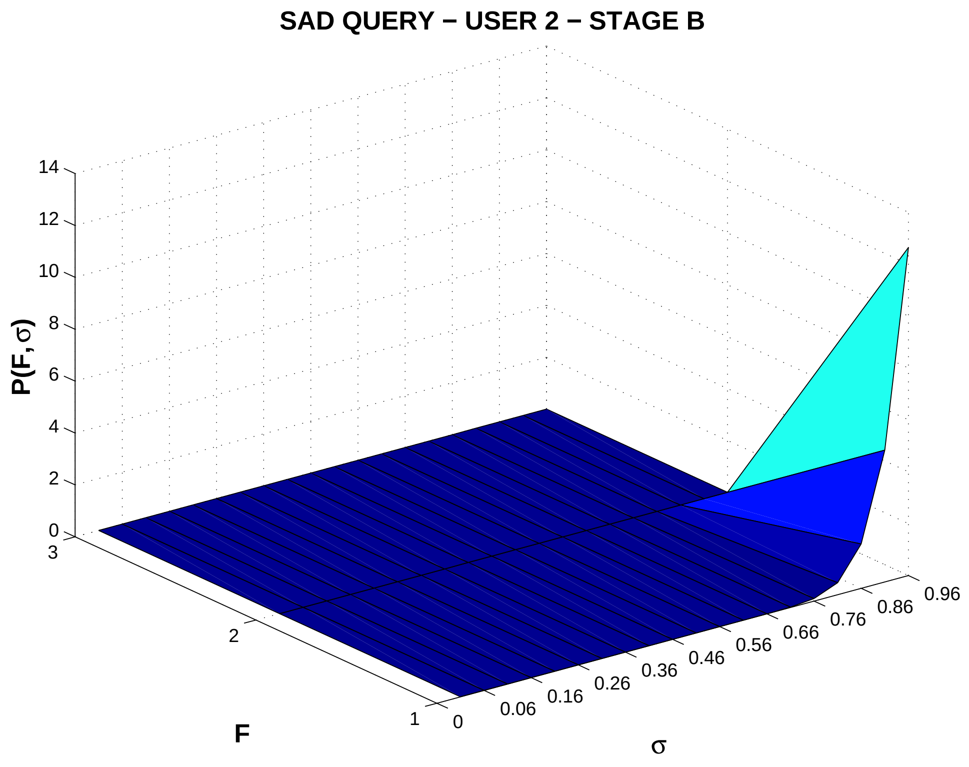


Figure 7.9: The joint marginal posterior probability of F and σ, $P(F, \sigma | H_{t_1}, ..., H_{t_10})$ after the run of stage A of the experiment (top) and after the run of stage B of the experiment (bottom) for a "sad query".

Factor	SS	df	p-value for F test
Emotional query	1.3	2	0.9551
Stage of experiment	64.067	1	**0.0379**
Interaction	13.433	2	0.6245
Error	763.6	54	

Table 7.11: Two-way analysis of variance to determine the effect of emotional query, the two stages of the experiment and their interaction on the number of iterations finding an image.

iterations do not vary from one emotional query to another. The p-value for the stage of experiment effect is 0.0379 which is significant. This indicates that one stage of experiment is out-performing the other in the number of iterations. Stage B helped the user 2 to find images in less iterations. The overall mean for the stage B was 4.56 when for the stage A was 6.63. There does not appear to be any interaction between stages of experiment and emotional queries.

We continue to test the normality assumption of the residuals as the theory of the Analysis of Variance demands. Plotting a normal probability plot (see Figure 7.10) for the residuals of ANOVA Table 7.11 we can judge about their normality. Looking again the points we can see a wavy form which indicates that the residuals are not normal. Also the p-value (Anderson-Darling statistic using MINITAB) is less than 0.05 which is also a strong indication that the residuals are not normal. So the assumption about residuals' normality is violated, which means that the ANOVA results may not be accurate. The next step is to transform the original data and run again the Analysis of Variance. The logarithmic transformation with base 10 was selected. We run again the analysis of variance for the transformed data and we get the standard ANOVA table in Table 7.12. The only significant factor appears to be the stage of experiment where the p-value is less than 0.05. This indicates that the stage B of the experiment

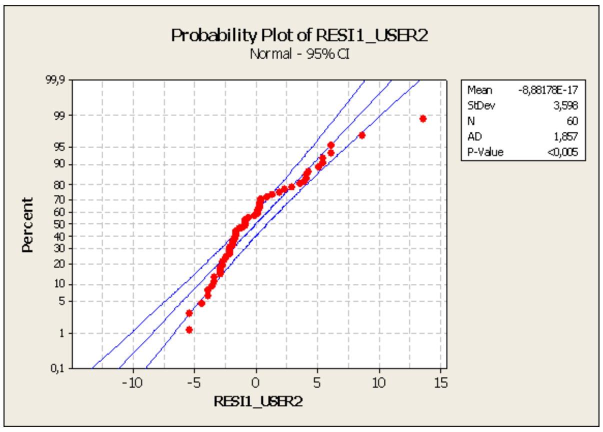


Figure 7.10: Normal probability plot for the residuals of ANOVA Table 7.11 of User 2.

Factor	SS	df	p-value for F test
Emotional query	0.03949	2	0.756
Stage of experiment	0.30151	1	**0.043**
Interaction	0.02472	2	0.839
Error	3.79057	54	

Table 7.12: Two-way analysis of variance for the transformed data.

helps the user 2 to find images in less iterations. This result agrees with the result of ANOVA table in Table 7.11 but now the residuals are normal, see normal probability plot for the residuals of the transformed data, see Figure 7.11.

Interaction Between the Two Users

The purpose of this subsection is to find if the data on a particular query from one user can be applied to another. The reason for trying this is to see if we can pick something generally useful about the abstract (emotional) space or if it is person specific. If knowledge from one user is transferable to another then it may be that this method could be used in practice to make searches easier for a new individual. Otherwise each individual has to go through a “training” stage with each system. Table 7.13

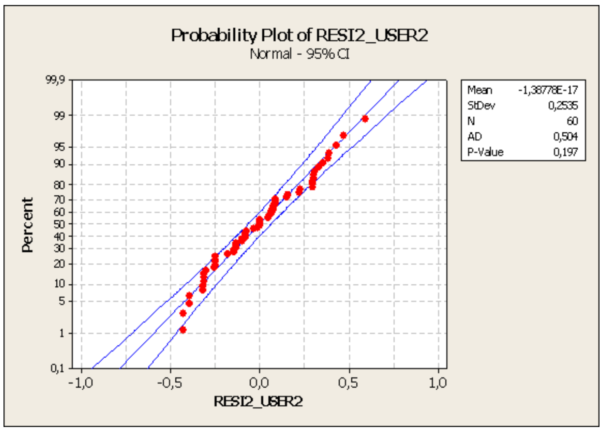


Figure 7.11: Normal probability plot for the residuals of ANOVA Table 7.12 of User 2.

shows the results for user 1 for stage B of the experiment using user 2's stage A information, in other words the joint marginal posterior of F and σ. An analysis of variance (see Table 7.14) were conducted to find if the user 2's information helped user 1 to find images in less iterations. The interaction effect seems to be most significant. This means that for some queries user 1 found images in less iterations using user 2's information but for other queries not. In particular for "romantic" and "sad" queries the average number of iterations was only 3.7 and 3.1 respectively when user 2's information was employed. This is better than the average number of iterations where no information was employed at all (it was 9.4 and 3.8 for the "romantic" and "sad" queries respectively). For the "violent" query there was not any difference.

7.5 Potential Application - Conclusions

The idea of summarizing the knowledge of previous searches or group user's posteriors could be applied to every kind of a CBIR system. A characteristic example could be applied for instance in an art gallery where a user standing in front of a touchscreen

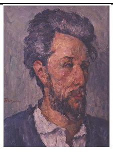
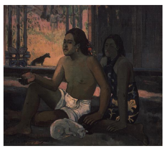
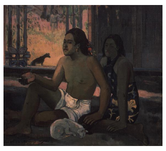
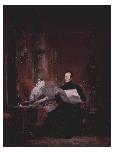
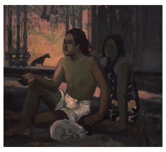
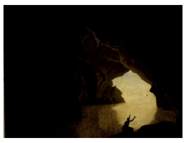
USER 1 - STAGE B using USER 2 - STAGE A					
ROMANTIC		VIOLENT		SAD	
Images	Iterations	Images	Iterations	Images	Iterations
	3		8		2
	2		7		4
	2		14		2
	7		11		2
	3		2		2
	7		2		7
	6		4		4
	3		2		4
	2		2		2
	2		2		2
Mean ($\tilde{x}$) Std (s)	3.7 2.1	Mean ($\tilde{x}$) Std (s)	5.4 4.4	Mean ($\tilde{x}$) Std (s)	3.1 1.6

Table 7.13: The results for User 1 for Stage B of the experiment using User 2 - Stage A information. The images as well as the number of iterations conducted to find each image are recorded.

Factor	SS	df	p-value for F test
Emotional query	96.23	2	0.066
Stage of experiment	56.07	1	0.073
Interaction	110.63	2	0.045
Error	907.00	54	

Table 7.14: Two-way analysis of variance for the user's interaction data.

machine could start searching for a romantic painting using other's belief or without other's belief and comparing the results at the end.

In this chapter we have conducted evaluation of our CBIR system on the BAL database without ground truth. Comparing our system with the basic PicHunter, using target testing, we found that different image types had a significant effect on the number of iterations to find the image. We then looked at the effect of using posterior information to aid searches in subjective queries based on emotional content. Here we found some evidence that the information can speed up a search even between different individuals.

Chapter 8

Conclusions - Contributions - Future Research

8.1 Conclusions

This research dealt with the problem of content-based image retrieval through a Bayesian perspective. For the construction of such a system several stages and tools are involved.

Given an image database, the first important stage is to extract quantitative information from images. Image segmentation and feature extraction algorithms help us to extract quantitative information from images. The supervised classification method that was used to classify the IGN satellite images was convenient, not very costly to implement and quite fast. We can evaluate statistical properties of the classes. Yet it does seem clear that it is unlikely that a single classification strategy will be "best" for all purposes. Rather, certain approaches (e.g. thresholding) may be more effective for certain classes of landscapes (narrow domains), while others (e.g. k-means clustering, Bayes's classification) may be more effective in a broad domain of images. The supervised method is quite effective for identifying urban areas in the IGN images.

Image feature description is the basis of CBIR. The most common visual characteristics are low-level features that represent colour, texture and spatial properties. Automated feature computation can be computationally demanding but can typically take place off-line prior to the retrieval process. Due to the subjective nature of visual information, no optimal feature descriptor exists for all tasks. Most of the features were calculated based on the PicHunter system. We have extended those features in the way that we accommodated features based on segmentation, recognising that there is no feature set can hope to distinguish well for all possible queries. Research in new features is on-going.

The next stage needs the implementation of an image similarity matching algorithm to achieve image retrieval. We adopted the Bayesian approach because it is a new area for research in the retrieval literature, and offers a natural way to describe the learning process that takes place with relevance feedback. PicHunter's new approach is its formulation within the Bayesian framework, which tries to predict the user's desired target image through a posterior probability distribution on the database that represents the probability of it being the target given the relevance feedback history. Like all CBIR systems, within the model is a rudimentary knowledge of humans' judgments of image similarity, based on empirically derived pictorial features. The image similarity is achieved through a Euclidean distance metric function.

The third major component is the display-updating scheme which is concerned with what images would be shown to the user. The most-probable display updating scheme, an open-ended browsing scheme as well as a scheme which uses both strategies were presented.

Finally, we conducted evaluation of our CBIR system on the BAL database without ground truth. Comparing our system with the basic PicHunter, using target testing, we found that different image types had a significant effect on the number of iterations to find the image. We then looked at the effect of using posterior information to aid searches in subjective queries based on emotional content. Here we found some evidence that the information can speed up a search. However, the posterior distribution used appears to be dependent on the user, making it less clear that data on a particular query from one user can be applied to another.

According to Eakins (1996), there seems little doubt that automatic image content retrieval will be seen as a desirable, if not essential, facility in the next generation of multimedia systems. Given current research trends, it should be feasible to equip such systems with effective low-level automatic image content retrieval facilities. The development of systems capable of matching users' real needs is a much more formidable task. However, the problems involved in meeting these needs, do not appear insoluble, and a number of possible ways of tackling these problems have already been identified. They should provide a worthwhile challenge to the research community for some time to come.

8.2 Contributions

- An extension to a more complex model for Bayesian relevance feedback has been proposed. Those proposals deal with learning about some new parameters in the current PicHunter model. A new display shows on the iterative session window which features caused images to be displayed. Expert users can benefit from knowing what are the system's current "beliefs", what feature subset is playing the most important role in the retrieval result. This will give them an idea of how their choices affect the system. Learning about the weight of each feature subset, we can investigate the feasibility of using visual browsing for the retrieval of emotional content of BAL images. Particularly, we focus on the question of whether the posterior distribution $P(F, \sigma|H_t)$ provides useful information that, when used as the prior in a subsequent search, shortens the search time.

- A decision-theoretic solution to the display strategy problem for a probabilistic image retrieval system has been proposed. Once *PicHunter* updates the posterior distribution across the entire database, the next task is to select the N_D images to be shown in the next display. The usual strategy is the most probable scheme, which we are going to present from a decision- theoretic perspective. While the most probable scheme is a reasonable strategy for target-specific search, for category search or open-ended browsing, where users search through a database with a rather broad, nonspecific goal in mind, it may not be appropriate to display the most probable images to a user, because does not allow us to capture a user's needs in a broader sense. For example, a user may not initially have the query image at hand or the ideal query may evolve during the retrieval process. In other words, by developing other display strategies we are proposing an open-ended search browser rather than a target-specific search system. This contribution fills the need for such a system in applications in graphics, art, photojournalism and advertising.

8.3 Future Research

We have described a Bayesian CBIR system and a decision-theoretic approach to the problem of the display set strategy. The model that we described is for the simplest possible relevance feedback. One result of this work is the realisation that there is not much information per iteration in such a scheme. Psychological studies have shown that we cannot expect a user to complete more than 30 iterations before becoming frustrated or bored. So one should attempt to extract more information per iteration by, for example, allowing more than one to be selected, ranking of images, images to be chosen that are to be excluded in future, and identification of objects in images that are of specific relevance. The problem here is to define a likelihood that models all these different types of more complicated data. Once we have the likelihood then we apply Bayes law to get the posterior.

- **Object Identification**. Assume image X_i is partitioned into n_i objects $\mathcal{O}_i = \{O_{i1}, ..., O_{in_i}\}$; user can pick one of these as most relevant part of image. The likelihood now will be something like:

$$P(A_i, O_{ij}|D_i, T, \mathcal{O}_T, \sigma, F) = P(A_i|D_i, T, \sigma, F)P(O_{ij}|A_i, \mathcal{O}_T, \sigma, F),$$

  where

$$P(O_{ij}|A_i, \mathcal{O}_T, \sigma, F) = \frac{\exp(-\min_k d_F(O_{ij}, O_{Tk})/\sigma)}{\sum_l \exp(-\min_k d_F(O_{il}, O_{Tk})/\sigma)};$$

  is a function of the distance between features of selected object and closest object to it in T.

- **More than one image may be selected**. Selecting m_i images $A_{i1}, ..., A_{im_i}$ from D_i has likelihood:

$$\begin{aligned} P(A_{i1}, ..., A_{im_i}|D_i, T, \sigma, F) &= \prod_{j=1}^{m_i} P(A_{ij}|A_{i1}, ..., A_{i(j-1)}, D_i, T, \sigma, F) \\ &= \prod_{j=1}^{m_i} \frac{\exp(-d_F(A_{ij}, T)/\sigma)}{\sum_{X \in D_i - \{A_{i1}, ..., A_{i(j-1)}\}} \exp(-d_F(X, T)/\sigma)}. \end{aligned}$$

Better specification of a prior for T , by colour and object shapes for example, could also help.

- Prior for T based on an image sketch S of query could be:

$$P(T) \propto \exp(-d(S,T)/\sigma).$$

In evaluation much more to be done, especially for subjective queries. It needs to test our emotional query experiments on more people before making firm conclusions. The basic problem is that, without ground truth, it is difficult to test without using real humans which implies high cost and not many samples. A possible solution is automatic testing where one simulates different types of user e.g. the "ideal user" is one who always chooses the image closest to the target. But given the semantic gap one cannot simulate users that are looking for emotional content.

Computation time is always a problem that prevents the use of complex statistical models. Parallel processing offers a solution if the computation of $P(T|H_t)$ can be partitioned into computation over subsets of the database with small communication load.

Appendix A

Simulated Annealing

A.1 Finding Final Temperature For A Large Solution Space

The task is to calculate the final temperature for a big $|S|$, where $|S|$ is the number of solutions, the number of sets D. We want to display N_D images out of N. In general we have,

$$|S| = \binom{N}{N_D} = \frac{N!}{(N-N_D)!N_D!}$$

For example, we decide to display $N_D = 6$ images from $N = 1066$ then the solution space will be

$$|S| = \binom{1066}{6} = \frac{1066!}{1060!6!} = 2 \times 10^{15}.$$

We want to calculate stopping temperature as it has been proposed by Lundy and Mees (1986),

$$T_{min} \leq \frac{\lambda}{\ln[(|S|-1)/\theta]}. \tag{A.1}$$

From the denominator of the above equation we have,

$$\ln\left(\frac{|S|-1}{\theta}\right) = \ln(|S|-1) - \ln(\theta). \tag{A.2}$$

Continuing with the $\ln(|S|-1)$ term we get,

$$\begin{aligned} \ln(|S|-1) &\approx \ln(|S|) \\ &= \ln\left(\frac{N!}{(N-N_D)!N_D!}\right) \\ &= \ln(N!) - \ln((N-N_D)!) - \ln(N_D!). \end{aligned} \tag{A.3}$$

We know that $n! = \prod_{i=1}^{n}(i)$ and taking the logarithms on both sides we have that

$$\begin{aligned} \ln(n!) &= \ln(\prod_{i=1}^{n} i) \\ &= \sum_{i=1}^{n} \ln(i). \end{aligned} \tag{A.4}$$

So, applying this to $\ln(N!) - \ln((N-N_D)!) - \ln(N_D!)$ we get,

$$\begin{aligned} \ln(N!) - \ln((N-N_D)!) - \ln(N_D!) &= \sum_{i=1}^{N} \ln(i) - \sum_{i=1}^{N-N_D} \ln(i) - \sum_{i=1}^{N_D} \ln(i) && (A.5) \\ &= \sum_{i=N-N_D+1}^{N} \ln(i) - \sum_{i=1}^{N_D} \ln(i) && (A.6) \\ &= \ln(|S|-1) && (A.7) \end{aligned}$$

For the BAL database, we have $N = 1066$ and $N_D = 6$. The final temperature is therefore:

$$\begin{aligned} T_{min} &\leq \frac{\lambda}{\ln[(|S|-1)/\theta]} && (A.8) \\ &= \frac{\lambda}{\ln(|S|-1) - \ln(\theta)} && (A.9) \\ &= \frac{0.001}{35.237 - \ln(0.99)} && (A.10) \\ &= 0.000028371. && (A.11) \end{aligned}$$

A.2 Testing Different Cooling Schedules for Simulated Annealing

Figure A.1 shows the runs that have been made using the posterior of T of Figure 6.8.

A.3 Principal Component Analysis output for each feature subset

The objective of PCA is to reduce the dimensionality of the feature vector while conserving or even enhancing the discriminatory information. In the following three subsections we show the PCA output for each of the three feature subsets.

PCA output for the segmentation features (F_{SG})

The F_{SG} variable is a data matrix with 1066 rows and 40 columns. To get a quick impression of the F_{SG} data, we made a box plot. As we can see in Figure A.2 that there is substantially more variability in f_{38} (horizontal edge length), f_{39} (vertical edge length) and f_{40} (total edge length) than the other features. The next step is to standardize the data which is reasonable when the variables are in different units or when the variance of the different columns is substantial (as in this case). We can standardize the data by dividing each column by its standard deviation. We now compute the principal components, that is linear combinations of the feature values that best describe their variation. Trying to see what features contribute to which principal component, we can look at the 10 largest component loadings. Table A.1 summarises the 10 largest component loadings in the first 5 principal components. The top 10 largest weights in the first principal component correspond with the *number of classes*, *horizontal edge length*, *total edge length*, *vertical edge length*, *number of centroid in (1,1) block*, *shortest distance between centroids*, *standard deviation of relative areas*, *minimum relative area*, *standard deviation of x-coordinates* and *standard deviation of y-coordinates*. It is difficult to attach an interpretable meaning to this component. However, PCA is not designed with this in mind, so it is hardly surprising. A more important objective in PCA lies in attempting to reduce the dimensionality of the data as a prelude to further tasks (image segmentation, CBIR etc.).

PC	Top 10 features
1^{st}	f_1 f_{38} f_{40} f_{39} f_{16} f_{10} f_5 f_2 f_6 f_7
2^{nd}	f_{37} f_{36} f_{35} f_{10} f_{20} f_{28} f_{27} f_{26} f_{25} f_{17}
3^{rd}	f_{39} f_{40} f_{38} f_{19} f_{11} f_{31} f_9 f_{30} f_{29} f_{24}
4^{th}	f_5 f_{16} f_3 f_8 f_{24} f_2 f_{14} f_9 f_4 f_6
5^{th}	f_{22} f_{26} f_{32} f_{25} f_{33} f_2 f_{34} f_6 f_{13} f_{12}

Table A.1: The 10 largest component loadings for F_{SG} in the first 5 principal components.

PCA output for the global colour features (F_{GC})

The F_{GC} variable is a data matrix with 1066 rows and 77 columns. Table A.2 summarises the 10 largest component loadings in the first 5 principal components. The top 10 largest weights in the first principal component correspond with the *mean saturation*, *percentages of pixels that fall into predefined ranges of HSV colourspace* particular red and orange colour, and 7 colour buckets from the HSV histogram.

PCA output for the global textural features (F_{TX})

The F_{TX} variable is a data matrix with 1066 rows and 385 columns. Table A.3 summarises the 10 largest component loadings in the first 5 principal components. The top

PC	Top 10 features
1st	f_{106} f_{110} f_{50} f_{49} f_{53} f_{54} f_{51} f_{111} f_{97} f_{101}
2nd	f_{113} f_{114} f_{81} f_{62} f_{85} f_{61} f_{77} f_{106} f_{107} f_{66}
3rd	f_{115} f_{114} f_{77} f_{94} f_{93} f_{97} f_{81} f_{78} f_{99} f_{102}
4th	f_{105} f_{99} f_{94} f_{98} f_{64} f_{109} f_{83} f_{68} f_{80} f_{79}
5th	f_{64} f_{68} f_{67} f_{110} f_{63} f_{93} f_{94} f_{71} f_{89} f_{66}

Table A.2: The 10 largest component loadings for F_{GC} in the first 5 principal components.

10 largest weights in the first principal component belong to the *HSV colour autocorrelogram.*

PC	Top 10 features
1st	f_{458} f_{394} f_{330} f_{474} f_{410} f_{490} f_{346} f_{454} f_{426} f_{266}
2nd	f_{438} f_{374} f_{441} f_{310} f_{458} f_{377} f_{394} f_{442} f_{313} f_{330}
3rd	f_{454} f_{441} f_{390} f_{377} f_{457} f_{326} f_{393} f_{313} f_{470} f_{329}
4th	f_{486} f_{438} f_{422} f_{374} f_{358} f_{310} f_{490} f_{426} f_{470} f_{362}
5th	f_{441} f_{377} f_{490} f_{426} f_{313} f_{362} f_{445} f_{478} f_{381} f_{249}

Table A.3: The 10 largest component loadings for F_{TX} in the first 5 principal components.

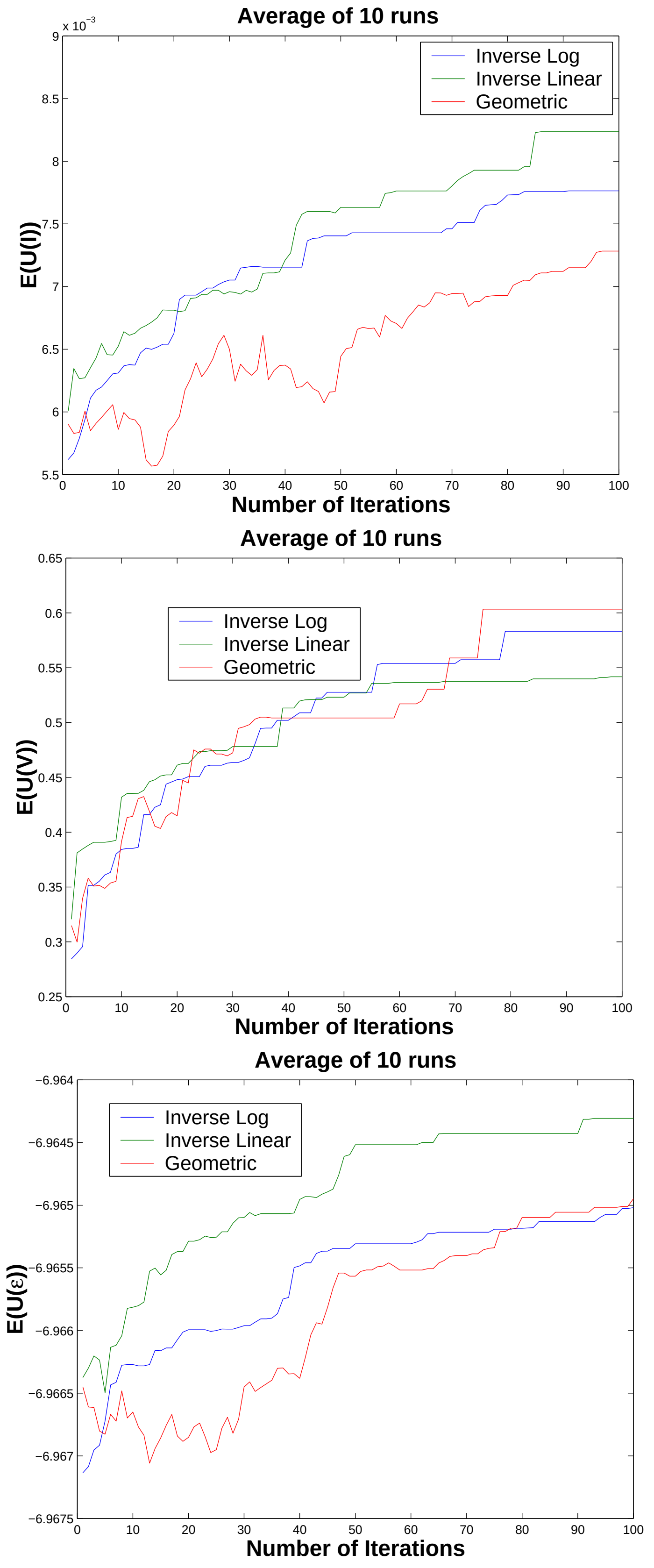


Figure A.1: Test runs for 3 different cooling schedules and 3 utilities.

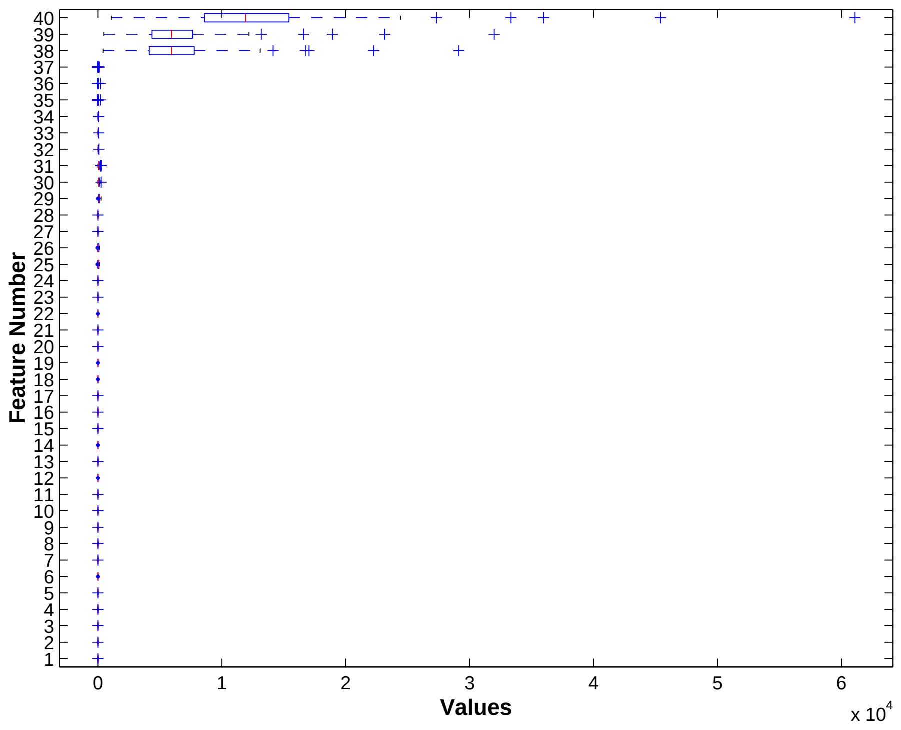


Figure A.2: A box plot for the segmentation features.

Bibliography

Aksoy, S. and R. Haralick: June 1998, Textural Features for Image Database Retrieval. *Proceedings of IEEE Workshop on Content-Based Access of Image and Video Libraries, in conjunction with CVPR'98, Santa Barbara, CA*, 45–49.

Aksoy, S. and R. M. Haralick, 2000: Probabilistic vs. Geometric Similarity Measures for Image Retrieval. *Computer Vision and Pattern Recognition*, **2**, 357–362.

Bach, J. R., C. Fuller, A. Gupta, A. Hampapur, B. Horowitz, R. Humphrey, R. Jain, and C.-F. Shu: 1996, The Virage Image Search Engine: An Open Framework for Image Management. *Storage and Retrieval for Image and Video Databases IV*, I. K. Sethi and R. C. Jain, eds., IS&T/SPIE Proceedings, San Jose, CA, USA.

Baddeley, A. J.: 1992, An error metric for binary images. *Robust Computer Vision: Quality of Vision Algorithms*, W. Förstner and H. Ruwiedel, eds., Wichmann, Karlsruhe, 59–78.

Berens, J., G. D. Finlayson, and G. Qiu: 2000, Image Indexing using Compressed Colour Histograms. *IEE Proceedings Vision Image and Signal Processing*, volume 147, 349–355.

Bimbo, A. D., 1999: *Visual Information Retrieval*. Morgan Kaufmann, San Francisco, California.

Blum, C. and A. Roli, 2003: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, **35**, 268–308.

Buhmann, J. M., D. W. Fellner, M. Held, J. Ketterer, and J. Puzicha, 1998: Dithered color quantization. *Computer Graphics Forum*, **17**, 219–231.

Campbell, J. B., 2002: *Introduction to remote sensing*. The Guilford Press, Spring Street, New York, third edition.

Castleman, K. R., 1996: *Digital Image Processing*. Prentice - Hall.

Celentano, A. and S. Chiereghin, 1999: Multiple strategies for relevance feedback in image retrieval. Techn. Rep. CS-99-8, Department of Computer Science, Ca' Foscari University, Venice.

Ciocca, R. and R. Schettini: 1999, Using a relevance feedback mechanism to imrove content-based image retrieval. *Lecture Notes in Computer Science*, Springer-Verlang, Heidelberg.

Cox, I., M. Miller, T. Minka, T. Papathornas, and P. Yianilos, 2000: The Bayesian Image Retrieval System, PicHunter: Theory, Implementation and Psychophysical Experiments. *IEEE Trans. Image Processing*, **9**, 20–37.

Cox, I. J., M. L. Miller, S. M. Omohundro, and P. N. Yianilos: 1996, Target Testing and the Pichunter Bayesian Multimedia Retrieval System. *Advances in Digital Libraries*, 66–75.

Deng, Y., B. S. Manjunath, C. Kenney, M. S. Moore, and H. Shin, 2001: An efficient color representation for image retrieval. *IEEE Transactions on Image Processing*, **10**, 140–147.

Eakins, J., 1996: Automatic Image Content Retrieval - Are We Getting Anywhere? *In Proc. of Third International Conference on Electronic Library and Visual Information Research*, 123–135.

Eakins, J. P. and M. E. Graham, 1999: Content-Based Image Retrieval, a report to the JISC Technology Applications Programme. Technical report, Institute for Image Data Research, University of Northumbria.

Enser, P. G. B., 1993: Query analysis in a visual retrieval context. *Journal of Document and Text Management*, **1**, 25–52.

Faloutsos, C., R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, 1994: Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, **3**, 231–262.

Fleck, M. M., D. A. Forsyth, and C. Bregler: 1996, Finding naked people. *European Conference on Computer Vision 2*, 593–602.

Forsyth, D. A., J. Malik, M. M. Fleck, H. Greenspan, T. K. Leung, S. Belongie, C. Carson, and C. Bregler: 1996, Finding pictures of objects in large collections of images. *Object Representation in Computer Vision*, 335–360.

Fuertes, J. M., M. Lucena, N. P. de la Blanca, and J. Chamorro-Martínez, 2001: A scheme of colour image retrieval from databases. *Pattern Recognition Letters*, **22**, 323–337.

Galbiati, L. J., 1990: *Machine Vision and Digital Image Processing Fundamentals*. Prentice - Hall.

Giacinto, G. and F. Roli, 2004: Bayesian relevance feedback for content-based image retrieval. *Pattern Recognition*, **37**, 1499–1508.

Glasbey, C. A. and G. W. Horgan, 1995: *Image Analysis for the Biological Sciences*. John Wiley & Sons, West Sussex, England, third edition.

Gupta, A. and R. Jain, 1997: Visual information retrieval. *Communications of the ACM*, **40**, 70–79.

Haralick, R. M., K. Shanmugam, and I. Dinstein, 1973: Textural features for image classification. *IEEE Trans. on Systems Man and Cybernetics*, **2**, 610–621.

Huang, J., S. Kumar, M. Mitra, W. Zhu, and R. Zabih: 1997, Image indexing using color correlograms. *In Proc. IEEE Comp. Soc. Conf. Comp. Vis. and Patt. Rec., pages 762–768*.

Hunt, R. W. G., 1998: *Measuring Colour*. Kingston-upon-Thames, third edition.

Jain, A. K., 1989: *Fundamentals of Digital Image Processing*. Prentice - Hall, Englewood Cliffs, New Jersey.

Jain, A. K., R. P. W. Duin, and J. Mao, 2000: Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**, 4–37.

Kato, T., 1992: Database architecture for content-based image retrieval. *Image Storage and Retrieval Systems, Proc SPIE 2185*, 112–123.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi, 1983: Optimization by simulated annealing. *Science*, **220**, 671–680.

Lee, J. Y.: 2001, Visual Access for Retriving Emotional Content of Pictures. *Proceedings of the First International Workshop on Multimedia Data and Document Engineering*, M.-S. Hacid, ed.

Lewis, R., 1990: *Practical Digital Image Processing*. Prentice - Hall.

Li, B., E. Chang, and C.-T. Wu, 2002: DPF — A Perceptual Distance Function for Image Retrieval. *In Proceedings of International Conference on Image Processing*, **2**, 597–600.

Liu, W. Y., Z. Su, Y. F. S. St. Li, and H. J. Zhang: 2001, Performance evaluation protocol for content-based image retrieval algorithms/systems. *In IEEE CVPR Workshop on Empirical Evaluation Methods in Compputer Vision*, Hawaii, USA.

Lundy, M. and A. Mees, 1986: Convergence of an annealing algorithm. *Mathematical Programming*, **34**, 111–124.

Ma, W.-Y. and B. S. Manjunath, 1999: Netra: A toolbox for navigating large image databases. *Multimedia Systems*, **7**, 184–198.

Manjunath, B. S., J.-R. Ohm, V. V. Vasudevan, and A. Yamada, 2001: Color and texture descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, **11**, 703–715.

Meghini, C., F. Sebastiani, and U. Straccia, 2001: A Model of Multimedia Information Retrieval. *ACM*, **48**, 909–970.

Muller, H., W. Muller, D. M. Squire, S. Marchand-Maillet, and T. Pun, 2001: Performance Evaluation in Content-Based Image Retrieval: Overview and Proposals. *Pattern Recogn. Lett.*, **22**, 593–601.

Nixon, M. S. and A. S. Aguado, 2002: *Feature Extraction and Image Processing.* Newnes, An imprint of Butterworth-Heinemann, Jordan Hill, Oxford.

Pass, G., R. Zabih, and J. Miller: 1996, Comparing images using color coherence vectors. *ACM Multimedia*, 65–73.

Pentland, A., R. Picard, and S. Sclaroff, 1994: Photobook: Content-based manipulation of image databases.

Petrou, M. and P. Bosdogianni, 1999: *Image Processing, The Fundamentals.* John Wiley & Sons.

Pietikainen, M., 1982: *Image Texture Analysis and Segmentation.* University of Oulu.

Ravela, S. and R. Manmatha: 1997, Retrieving images by similarity of visual appearance. *Workshop on Content Based Access of Image Databases (with CVPR) 2:311–347.*.

Rue, H. and M. A. Hurn, 1997: Loss functions for bayesian image analysis. *In the Art and Science of Bayesian Image Analysis, (editors K. V. Mardia, C. A. Gill and R. G. Aykroyd), Leeds: Leeds University Press*, 72–79.

Rui, Y., T. S. Huang, M. Ortega, and S. Mehrotra, 1998: Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, **8**, 644–655.

Santini, S., 2001: *Exploratory Image Databases: Content-Based Retrieval.* Academic Press, San Diego, California, USA.

Santini, S. and R. Jain, 1997: Similarity is a geometer. *Multimedia Tools and Applications*, **5**, 277–306.

Santini, S. and R. Jain: 1998, Beyond query by example. *ACM Multimedia*, 345–350.

Santini, S. and R. Jain, 1999: Similarity Measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, **21**, 871–883.

Schott, J. R., 1997: *Remote Sensing, The Image Chain Approach*. Oxford University Press, New York.

Schowengerdt, R. A., 1997: *Remote Sensing, Models and Methods for Image Processing*. Academic Press, San Diego, CA, second edition.

Shaffrey, C. W., N. G. Kingsbury, and I. H. Jermyn: September 2002, Unsupervised image segmentation via Markov trees and complex wavelets. *Proceedings International Conference Image Processing*.

Sia, K. C. and I. King: 2002, Relevance feedback based on parameter estimation of target distribution. *Proceedings of the 2002 International Joint Conference on Neural Networks*, volume 2, 1974 – 1979.

Smeulders, A. W. M., M. Worring, S. Santini, A. Gupta, and R. Jain, 2000: Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**.

Smith, J. and S. Chang: 1995, Single color extraction and image query. *In Proc. IEEE Int. Conf. on Image Proc.*, 528–531.

Smith, J. R.: 1998, Image retrieval evaluation. *In IEEE Workshop on Content-Based Access of Image and Video Libraries*, Santa Barbara, California, 112–113.

Smith, J. R. and S.-F. Chang: 1996a, Tools and techniques for color image retrieval. *Storage and Retrieval for Image and Video Databases (SPIE)*, volume 2670, 426–437.

— 1996b, Visualseek: A fully automated content-based image query system. *ACM Multimedia*, 87–98.

— 1997, Image and video search engine for the world wide web. *Storage and Retrieval for Image and Video Databases (SPIE)*, 84–95.

Stander, J. and B. W. Silverman, 1994: Temperature schedules for simulated annealing. *Statistics and Computing*, **4**, 21–32.

Stricker, M. A. and M. Orengo: 1995, Similarity of color images. *Storage and Retrieval for Image and Video Databases SPIE*, 381–392.

Swain, M. J. and D. H. Ballard: 1990, Indexing via color histograms. *Third International Conference on Computer Vision*, 390–393.

Tanner, M. A., 1996: *Tools for Statistical Inference*. Springer Verlag, New York.

Vailaya, A., M. A. T. Figueiredo, A. K. Jain, and H.-J. Zhang, 2001: Image Classification for Content-Based Indexing. *IEEE Transactions On Image Processing*, **10**, 117–130.

Veltkamp, R. and M. Tanase, 2000: Content-Based Image Retrieval Systems: A Survey, Technical Report, Utrecht University, Department of Computing Science.

Wang, Y., Z. Liu, and J.-C. Huang, 2000: Multimedia Content Analysis - using both Audio and Visual Clues. *IEEE Signal Processing Magazine*, **17**, 12–36.

Wilson, D. L., A. J. Baddeley, and R. A. Owens, 1997: A new metric for grey-scale image comparison. *International Journal of Computer Vision*, **24**, 5–17.

Wilson, R. and M. Spann, 1988: *Image Segmentation and Uncertainty*. Research Studies Press.

Winston, W. L., 1987: *Operations Research, Applications and Algorithms*. Duxbury Press, Boston.

Yu, K., W.-Y. Ma, V. Tresp, Z. Xu, X. He, H. Zhang, and H.-P. Kriegel: 2003, Knowing a tree from the forest: art image retrieval using a society of profiles. *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, ACM Press, 622–631.