UNIVERSITY OF CALIFORNIA
Santa Barbara

# Managing Large-scale Multimedia Repositories

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

by

Jelena Tešić

Committee in Charge:

Professor B.S. Manjunath, Chair

Professor Shivkumar Chandrasekaran

Professor Kenneth Rose

Dr. Chandrika Kamath

September 2004

The Dissertation of
Jelena Tešić is approved:

_____

_____

_____

Committee Chairperson

September 2004

To my parents,

Milomir and Lepa Tešić,

for their love, patience, and support.

# Acknowledgements

Completing this doctoral work has been a memorable, and often an overwhelming experience. I thank my advisor, Professor Manjunath for his guidance, teaching, patience, and support. Also, I thank Professor Chandrasekaran, and Professor Rose, for many fruitful discussions, and research motivations. I thank Professor Sanjit Mitra for his professional and personal support, specially throughout my first year at UCSB. I thank Dr. Chandrika Kamath and Dr. Imola K. Fodor for many useful comments and helpful points during the course of this research. Throughout my graduate studies, my research work was supported by the Lawrence Livermore National Laboratory Institute of Scientific Computing Research, and the National Science Foundation Information Technology Research Program.

I thank my colleague Kristoffer N. Bruvold for being a constant source of support and encouragement, for proof reading my thesis, and for raising a bar for all Teaching Assistants in the department. Professor Manjunath and Professor Mitra fostered certainly the most open, friendly, diverse, and collaborative research group in the department. I thank Peng Wu, Shawn Newsam, Yining Deng, Sitaram Bhagavathy, and Bariş Sümengen for collaboration, dynamic brainstorming sessions, and help. Also, I would like to thank John Berger, Jiyun Byun, Ching-Wei Chen, Gabriel Gomez, Myléne Farias, Dmitry Fedorov, Mike Moore, Kaushal Solanki, Ken Sullivan, Xinding Sun, Marco Zuliani, and all other former and current students and visitors of the Vision Research and Image Processing Lab for their valuable assistance during my stay at UCSB. From the Electrical and Computer Engineering Department, Valerie De Veyra, Ken Dean, Gylene Gadal, and Tim Robinson are specially thanked for their care and attention.

I am grateful to all friends in Santa Barbara for being my surrogate family during the many years I stayed there, and for their continued moral and emotional support. This was a smooth and fun journey also thanks to: Lilijana Dukić, Milisav Pavlović, Lila & Srba Topalski, Branka Božović, Nikolina Čingel, Una Matko, Ioanna Pagani, Roxana Stanoi, Iva Božović, Doca Popović, Zoran Dimitrijević, Costin Iancu, and Karlo Berket.

I am forever indebted to my sister Mirjana Tešić, my parents Lepa & Milomir Tešić, my grandparents Slobodanka and Milinko Jakovljević, uncle Risto Tešić, and to all my family and friends from Serbia for their understanding, endless love, patience, and encouragement.

# Curriculum Vitæ

Jelena Tešić

## EDUCATION

| | |
|---|---|
| 1999–2004 | Ph.D. in Electrical and Computer Engineering |
| | University of California, Santa Barbara, California. |
| 1998–1999 | M.S. in Electrical and Computer Engineering |
| | University of California, Santa Barbara, California. |
| 1993–1998 | Dipl. Ing. in Electrical Engineering |
| | University of Belgrade, Belgrade, Serbia & Montenegro. |

## EXPERIENCE

| | |
|---|---|
| 2004-present | Research Staff Member |
| | IBM T.J. Watson Research Center, Hawthorne, New York. |
| 1999-2003 | Research Assistant |
| | University of California, Santa Barbara, California. |
| Summer 2000 | Summer Intern |
| | HP Labs, Palo Alto, California. |
| 1998-1999 | Teaching Assistant |
| | University of California, Santa Barbara, California. |

## Publications


[1] Shawn Newsam, Jelena Tešić, Lei Wang, and B.S. Manjunath, "Issues in Mining Video Datasets," in *SPIE Int. Symp. On Electronic Imaging, Storage and Retrieval Methods and Applications for Multimedia*, San Jose, California, January 2004.

[2] Jelena Tešić, Sitaram Bhagavathy, and B. S. Manjunath, "Issues Concerning Dimensionality and Similarity Search," in *International Symposium on Image and Signal Processing and Analysis (ISPA)*, Rome, Italy, September 2003.

[3] Sitaram Bhagavathy, Jelena Tešić, and B. S. Manjunath, "On the Rayleigh nature of Gabor filter outputs," in *IEEE International Conference on Image Processing (ICIP)*, Barcelona, Spain, September 2003.

[4] Jelena Tešić and B. S. Manjunath, "Nearest Neighbor Search for Relevance Feedback," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Madison, WI, June 2003, pp. 643–648.

[5] Jelena Tešić, Shawn Newsam, and B.S. Manjunath, "Mining Image Datasets using Perceptual Association Rules," in *SIAM Sixth Workshop on Mining Scientific and*

*Engineering Datasets in conjunction with SIAM/SDM*, San Francisco, CA, May 2003.

[6] Jelena Tešić, Shawn Newsam, and B.S. Manjunath, "Scalable Spatial Event Representation," in *IEEE International Conference on Multimedia and Expo (ICME)*, Lausanne, Switzerland, August 2002.

[7] Jelena Tešić, Shawn Newsam, and B.S. Manjunath, "Challenges in Mining Large Image Datasets," in *IPAM Short Program on Mathematical Challenges in Scientific Data Mining*, Los Angeles, CA, January 2002.

[8] G.S. Orton, B.M. Fisher, K.H. Baines, S.T Stewart, A.J Friedson, J.L Ortiz, M. Marinova, M. Ressler, A. Dayal, W. Hoffmann, J. Hora, S. Hinkley, V. Krishnan, M. Mašanović, J. Tešić, A. Tziolas, and K.C. Parija, "Characteristics of the Galileo probe entry site from Earth-based remote sensing observations," *Journal of Geophysical Research*, September 1998.

# Abstract

## Managing Large-scale Multimedia Repositories

by

Jelena Tešić

Capturing and organizing vast volumes of images, such as scientific and medical data, requires new information processing techniques for context of pattern recognition and data mining. In content based retrieval, the main task is the seeking of entries in an image database that are most similar, in some sense, to a given query object. The volume of the data is large, and the feature vectors are, typically, of high dimensionality. In high dimensions, the *curse of dimensionality* is an issue as the search space grows exponentially with the dimensions. In addition, it is impractical to store all the extracted feature vectors from millions of images in main memory. The time spent accessing the feature vectors on hard storage devices overwhelmingly dominates the time complexity of the search. The time complexity problem is further emphasized when the search is to be performed multiple times in an interactive scenario.

One of the main contributions of this dissertation is to enable efficient, effective, and interactive data access. We introduce a modified texture descriptor that has comparable performance but nearly half the dimensionality and less computational expense. Moreover, based on the statistical properties of the texture descriptors, we propose an adaptive for approximate nearest neighbor search indexing approach. In content-based

retrieval systems, exact search and retrieval in the feature space is often wasteful. We present an approximate similarity search method for large feature datasets. It improves similarity retrieval efficiency without compromising on the retrieval quality.

We also address the computation bottleneck of a real-life system interface. We propose a similarity search scheme that exploits correlations between two consecutive nearest neighbor sets and considerably accelerates interactive search, particularly in the context of relevance feedback mechanisms that support distance metric update approach.

In multimedia query processing, the main task is the seeking of entries in a multimedia database that are most similar to a given query object. Since feature descriptors approximately capture information contained in images, they often do not capture visual concepts contained in those images. Semantic analysis of multimedia content is needed. We introduce a framework for learning and summarizing basic semantic concepts in scientific datasets. Moreover, we present a method to detect coarse spatial patterns and visual concepts in image and video datasets. Experiments on a large set of aerial images and video data are presented.

---

Professor B.S. Manjunath
Dissertation Committee Chair

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Vast amounts of images are being created in many areas of science and commerce, necessitating the need for new image/video data management and knowledge discovery tools. While much progress has been made in the area of content-based retrieval, more work is needed in building tools for efficient and effective data access.

The main contribution of this dissertation is to enable such efficient, effective, and interactive data access in large image and video collections. We investigate how to efficiently access the image features, how to perform similarity search in relevance feedback scenarios and how to summarize and characterize information content in images in order to discover underlying patterns. The methodologies we propose involve approaches whose origins lie in several disciplines in addition to classical data management, including optimization, information theory, pattern recognition, signal compression and image processing. The emphasis of this work is on the problems related to image analysis for large-scale image collections.

## 1.1 Motivation

This dissertation work was motivated by a challenge to enable an organized and easily accessible and searchable multimedia database of large collections of images. In many areas of science and commerce, vast amounts of images have been created. Capturing and organizing vast volumes of images requires improved tools and information processing techniques, specifically in the context of pattern recognition and data mining. The unique nature of the media data makes the problem significantly more difficult and interesting with many commercial and scientific applications. Therefore, there is a clear need for a base multimedia infrastructure, simple to use and change, but complex enough to handle large image repositories.

There has been considerable work in the past decade on the use of low level visual features for content based image and video retrieval. The IBM QBIC project [47] played an important role in inspiring effort along this direction. Afterwards, the next generation of content-based image retrieval (CBIR) systems attempted to solve more complex issues, such as: better description of a multimedia object in terms of features, giving more control to the user in query formulation, and better search and retrieval mechanisms.

Netra [75] from the University of California, Santa Barbara incorporated robust automated image segmentation algorithm and allowed object or region-based search and demonstrated that segmentation significantly improves the quality of image retrieval when images contain multiple complex objects. VisualSEEK [116] from the Columbia

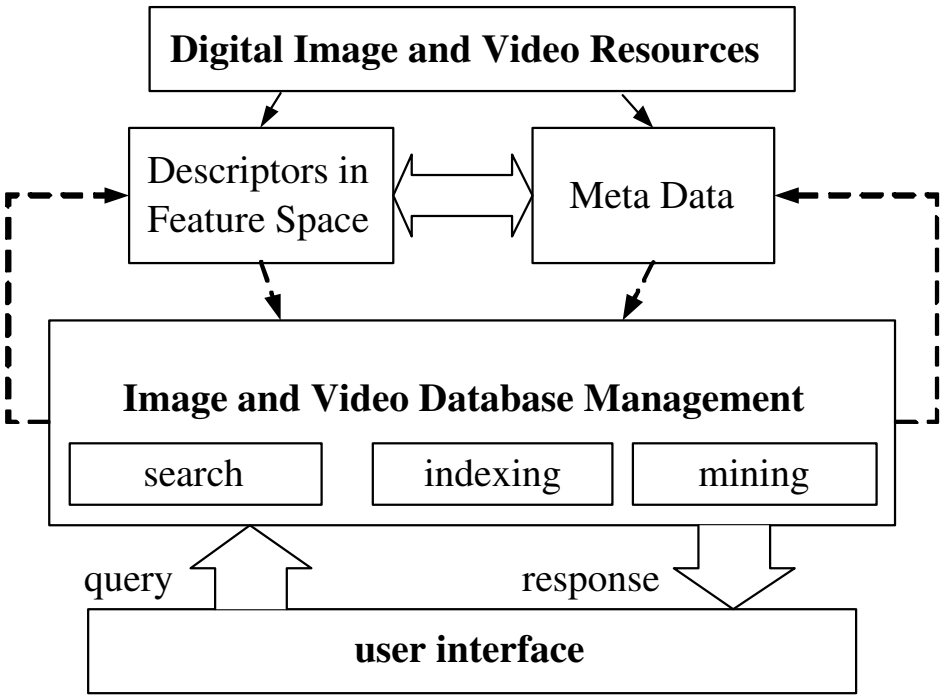



Figure 1.1: Generic Content-based Image Retrieval Framework.

University provided a sketch board so the user could express a query in the form of simple visual concept, and retrieved images contained the closest arraignment of similar regions. Blobworld [27] from the University of California, Berkeley allowed users to compose a query by using objects segmented from images. PicHunter [35] from MIT and MARS [107] from the University of Illinois at Urbana-Champain allowed users to decide which images are the most informative ones to iteratively find target images from a database.

A general framework of a CBIR system is shown in Figure 1.1. Given a collection of image data, relevant feature descriptors are computed to represent the content. Low level visual descriptors are computed from the data, and multiple descriptors may be needed to describe a given image, such as color and texture. The system processes a

query submitted by a user and offers a set of images/videos as a response. The user can refine the query and obtain another answer set from the CBIR system if the offered retrieval set is not the desired answer. As reported in [115] and [128], most of these CBIR systems exist in research form and have not yet been commercially developed. The size of the database used by existing CBIR systems is relatively small, and implemented solutions do not scale well with database size increases [144]. Recently, the primary emphasis in multimedia research is to develop more advanced CBIR systems for scientific and commercial applications, such as Cortina (http://sobel.ece.ucsb.edu/search.html), Vima's Image Search Engine [141], Virage VIR Image Engine [7], and Convera Image RetrievalWare (http://www.convera.com).

Evaluation of CBIR systems largely depends on their effectiveness and efficiency. The effectiveness of a CBIR system depends on the content of the extracted image descriptors. The ISO/MPEG-7 international standard [78] provides a good showcase of the achievements in extracting effective low-level descriptors and metadata. The efficiency of a CBIR system depends on the organization and data storage structure. When a query is presented, the system needs to search and return the query results in timely fashion. Since input/output (I/O) operations for storage devices are slow, the time spent accessing the feature vectors overwhelmingly dominates the time complexity of the search. In order to access data items efficiently, index structures are used. The efficiency of the search procedure depends on how the system utilizes the indexing scheme and how well it scales to the number of database items. For example, to support nearest neighbor queries, index structures typically partition the data or data space and

store information on each partition. However, indexing techniques which provide good results on lowŋdimensional data fail to perform sufficiently well on highŋ-dimensional multimedia features [22].

In this work, we propose solutions and develop tools that will enable easily accessible large-scale CBIR image database systems. We also identify and address some of the challenging issues that are at the fundamentals of visual analysis and database indexing.

## 1.2 Contributions

The combination of proposed solutions and tools in this thesis enables effective and efficient search of large image databases for relevant phenomena. The main contributions of thesis are summarized below.

### 1.2.1 Compression of the Texture Feature Space

Typical image/video descriptors are of high dimensionality (from tens to several hundreds). For example, a 60-dimensional color histogram descriptor is often used to characterize the texture in a given image. The feature space grows exponentially with the dimensions, and the search complexity increases at the same rate. This rapid increase has been termed the *curse of dimensionality* by Bellman [10]. The high dimensionality and computational complexity of typical texture descriptors used in CBIR adversely affect the efficiency of such systems.

In particular, we consider the MPEG-7 texture descriptor that has proven to be quite effective for image and video search [78] and retrieval. In Chapter 3 we propose a new approach to dimensionality reduction of the feature vector. It is based on the MPEG-7 descriptor, cost effective, and removes data redundancies. The modified texture descriptor that has comparable performance, but half the dimensionality and, hence, far less computational expense [20]. Furthermore, it is easy to compute the proposed feature vector using the existing MPEG-7 descriptor. In order to provide a more objective comparison, feature vectors are normalized along each dimension to have the same dynamic range. We propose a novel approach to normalization [121] based on the behavior of the filter statistics along each feature dimension. This technique gives better retrieval performance than existing ones [74].

### 1.2.2 Quadratic Distance Queries for Relevance Feedback

Relevance feedback learning is a popular scheme used in content-based image and video retrieval to support high–level "concept" queries. During the learning process, the algorithm needs to access the high–dimensional descriptors associated with image/video objects. Searching in high dimensions is expensive. The time complexity problem is further exacerbated when the search is to be performed multiple times as in relevance feedback. The impact of using learning on the indexing structure has generally been ignored, and the feasibility of learning algorithms has not been evaluated for handling large amounts of image/video data in relevance feedback scenarios.

We propose in Chapter 4 an efficient algorithm for repetitive searches to compute nearest neighbors in high–dimensional feature spaces [122]. We address scenarios in which a similarity or distance matrix is updated during each iteration of the relevance feedback search, and a new set of nearest neighbors are computed. The proposed scheme exploits correlations between two consecutive nearest neighbor sets and significantly reduces the overall search complexity for general distance metric updates and compression-based data indexing. The nearest neighbor computation in each iteration is quadratic with the number of dimensions and linear with the number of items searched. The proposed scheme enables the use of the user's feedback not only to improve the effectiveness of the similarity retrieval, but also its efficiency in an interactive content based image retrieval system. We show that vector quantization-based indexing structures can support relevance feedback and suggest a modification to an existing nearest neighbor search algorithm to support relevance feedback for the weighted Euclidean and quadratic distance metric. The new scheme significantly reduces the overall search complexity by reducing the number of the data accessed on the storage devices by as much as a factor of 90.

### 1.2.3 Approximation Search Scheme

In the context of image, video and audio data, the extraction of feature vectors from the data objects is itself a heuristic process that attempts to approximately capture relevant information. Exact search and retrieval in such circumstances is often wasteful.

For high feature dimensions, the *curse of dimensionality* is an issue since the traditional database indexing and clustering methods do not scale well beyond 10–20 dimensions. This poses challenging problems for database access. Thus, rather than incur the extremely high cost of an exact result, it is more effective to develop a fast search engine that outputs an approximate set.

So far, the efficiency of query processing was improved by using (1) index structures, and (2) compressed data representatives. In Chapter 5 we propose a new approach to efficiently process queries in multimedia databases: characterize the data based on the relations between the distribution and relation among low-level feature dimensions. This approach scales well with the database size and supports approximate similarity searches. The proposed method improves the efficiency of similarity retrieval without compromising on the retrieval quality. This technique significantly reduces overall search complexity by reducing the number of accessed data on storage devices by a order of magnitude.

### 1.2.4 Multimedia Mining in High Dimensions

Meaningful semantic analysis and knowledge extraction require data representations that are understandable at a conceptual level. Our goal is to obtain a framework for summarizing basic semantic concepts to detect coarse spatial patterns and visual concepts in image and video aerial data and biological imagery.

In Chapter 6.3 we propose the use of visual thesaurus that provides summarized

data information derived from the low-level features of scientific dataset. We introduce a novel data structure termed Spatial Event Cube (SEC) for conceptual representation for complex spatial arrangements of visual thesaurus entries in large multimedia datasets. Visual thesaurus entries and their spatial relationships define a non-traditional space for data mining applications. This space can be used to discover simple spatial relationships in scientific dataset. We present a novel extension of traditional association rule algorithm, the *perceptual association rule algorithm*, to distill the frequent visual patterns in image and video datasets in order to discover interesting patterns.

## 1.3 Organization

This thesis is organized as follows.

- In Chapter 2 we present curse of dimensionality, its effects on large image databases, and an overview of the related work on scalability issues in large datasets, specifically in the areas of high–dimensional indexing, similarity search, and learning.

- In Chapter 3 we introduce MPEG-7 homogeneous texture feature descriptor and propose its modification for more efficient access. We also propose new normalization scheme superior to the current approach.

- In Chapter 4, we propose an efficient algorithm for repetitive searches to compute nearest neighbors in high dimensional feature spaces.

- In Chapter 5, we propose an adaptive indexing scheme that supports approximate similarity search over large image datasets. The method scales well without compromising on the retrieval quality.

- In Chapter 6, we present a scalable visual mining framework that supports event mining and provides efficient pruning for generating higher order candidate itemsets.

- We summarize our research findings and discuss future work and directions in Chapter 7.

# Chapter 2

# Curse of Dimensionality

This chapter discusses the curse of dimensionality and its effects on large image databases. We review previous work related to overcoming the dimensionality curse in large high–dimensional databases, notably nearest neighbor search, clustering and indexing. We focus on techniques which are needed or referenced later in this dissertation.

## 2.1 Introduction

With the rapid advances of large-scale multimedia applications, the amount of data that needs to be processed is growing at a fast rate. A general approach in most modern applications is to generate feature vectors, i.e. object *descriptor* datasets, that represent the original data objects. The size of these datasets extends beyond what can be realized by traditional "focused" solutions in multimedia access and management, where the testbed consists of a couple a thousand images. Datasets consisting of more

than a few hundred thousand images are becoming common. For example, the size of the Cortina database for online content-based image retrieval is over 3 million images and traditional solutions do not easily scale to such large numbers [128].

In addition, the high dimensions of low–level descriptors, such as color histogram, texture and shape, pose a significant challenge on the effective access, summarization and query. The main concerns are data access complexity and "curse of dimensionality." The phrase "curse of dimensionality" was coined by Bellman in 1961 [10]. Bellman used the term to describe the rapid increase in complexity of adaptive control processes, where the number of computations exceeds the available computing power. Today, this term is often used in the pattern recognition and database community to describe a broad variety of mathematical effects that occur in data access and data modeling when the dimensionality of data space is large. Essentially, the structure of high–dimensional spaces can run counter to intuition, which tends to be based on Euclidean spaces of small dimensionality (i.e. 2 or 3 dimensions). Density estimation becomes a challenging task in the high–dimensional space. In statistics, the "dimensionality curse" refers to the fact that the convergence of any estimator to the true value of a smooth function defined on a space of high dimensions is very slow, i.e. a large number of observations is needed to obtain an acceptable estimate. Samples quickly become "lost" in the wealth of the space, while simultaneously, the required sample size increases exponentially with an increasing number of dimensions [113]. Other dimensionality curse effects cause increased complexity, performance degradation and degeneration of traditional nearest neighbor search, clustering and indexing methods.

## 2.2 Nearest Neighbors in High Dimensions

Spatial queries in high–dimensional spaces have been studied extensively [109, 53, 111, 19]. Among them, nearest-neighbor queries are important in many settings, including spatial databases, e.g. find the $k$ closest cities, and multimedia databases e.g. find the $k$ most similar images. A standard technique is to map those data items as points into a high–dimensional feature space. The feature space is usually indexed using a multidimensional index structure. Similarity search then corresponds to a hyperspherical range search, which returns all objects within a threshold level of similarity to the query objects, and a $K$ nearest neighbor search that returns the $K$ most similar objects to the query object.

**Definition 1 (Nearest Neighbor Search)** *Let $\Phi$ be a set of $M$ dimensional points. Define the distance between points $Q$ and $F$ as $d(Q, F)$. The nearest neighbor to the query point $Q$ is a data point $NN(Q) \in \Phi$ such that $NN(Q)$ is closer to $Q$ than any other point in $\Phi$:*

$$NN(Q) = \{F \in \Phi | \forall F' \in \Phi : d(Q, F) \leq d(Q, F')\}$$

**Definition 2 (K Nearest Neighbor Set and Radius)** *The $K$ nearest neighbors to the query point $Q$ in database $\Phi$ are defined as a set $NN_K(Q)$ of at least $K$ data points, $NN_K(Q) \subset \Phi$, such that:*

$$NN_K(Q) = \{F_1, ..., F_K \in \Phi | \forall F' \neq F_i, F' \in \Phi : d(P, F_i) \leq d(P, F')\},$$

*The K-NN radius or K-NN range is then defined as:*

$$max\{d(Q,F), F \in NN_K(Q)\}$$

The volume of $M$-dimensional hyper-cube with edges of length r is $r^M$. Therefore, for a constant edge length, the volume of the hypercube grows exponentially with increasing dimensions and constant edge length. The proportion of this volume and the volume contained between the surface and a smaller hypercube with edge length $r - \epsilon$ is:

$$\frac{r^M - (r-\epsilon)^M}{r^M} = 1 - (1 - \frac{\epsilon}{r})^M \to 1, \quad \text{as} \quad M \to \infty$$

Thus, in high dimensions, most of the volume of the cube is spread around the surface of the cube region. This has an effect on the nearest neighbor search. The radius of the nearest neighbor query and the query point define a hypersphere in high–dimensional space. The largest spherical query that fits into the $M$-dimensional hyper-cube with edges of length $r$ has the radius $r/2$, and it forms hypersphere $\Psi$. Then, the probability that an arbitrary point lies within this sphere is:

$$\frac{Vol(\Psi)}{r^M} = \frac{(\sqrt(\pi)/2)^M}{(M/2)!} \to 0, \quad as \quad M \to \infty.$$

This, if the data dimension $M$ is large, there only is a small probability that any data point will be inside the sphere. Moreover, results in [19] demonstrate that the maximum and minimum distances to a given query point in high–dimensional space are almost the same under a wide range of distance metrics and data distributions. All points converge to the same distance from the query point in high dimensions, and the concept of nearest neighbors becomes meaningless.

However, there are scenarios where nearest neighbor search in high dimensions is meaningful. The first is when the underlying dimensionality of the data is much smaller than the actual dimensionality. The second is when the search space is limited to only a cluster to which the query point belongs. We now present the recent research effort in these two directions to overcome the "dimensionality curse".

## 2.3 Reduction of Dimensionality

Dimensionality reduction algorithms are needed to support scalable object retrieval in high dimensions and large databases and there has been a lot of work done in the area of determining the meaningful dimensions. These approaches can be loosely classified into two categories: (1) distance preserving methods (2) energy preserving methods. We now discuss these two categories and some of their implementations.

### 2.3.1 Distance Preserving Methods

Distance mapping algorithms take the set of objects and inter-object distances and map the objects to a space with lower dimension such that distances among the objects are approximately preserved. In an ideal situation, for any given query one should find the same set of nearest neighbors in the low–dimensional space as in the original high–dimensional space. The lower dimensional representation is found by minimizing a certain cost function.

**Multidimensional Scaling (MDS)** [23] approach adopts the difference in pairwise distances between feature points in the original and low–dimensional space as a cost function. MDS is an iterative method that preserves the global topology of the feature space. In a metric space, points are mapped to a 2D or 3D space so that the distances between pairs of vectors remain in the same order. Computational time is proportional to $O(N^3)$ since the distance between every pair of objects must be computed at every iteration, thus making this method impractical for large datasets. Preserving only the local topology of one point with respect to $K$ closest points, rather than global one, can further reduce the complexity to $O(NK^2)$, see [140]. The performance of the local method [140] is similar to the global MDS, and the complexity is still high for large datasets.

In [43], a significantly faster algorithm called **FastMap**, based loosely on the MDS principles, is presented. FastMap maps vectors to a low–dimensional space while approximately preserving the distances between objects. The axes of the new space are determined by the vectors that are mutually farthest apart. The complexity is linear with number of data in the database, $O(N)$, but its performance is highly data dependant.

### 2.3.2 Energy Preserving Methods

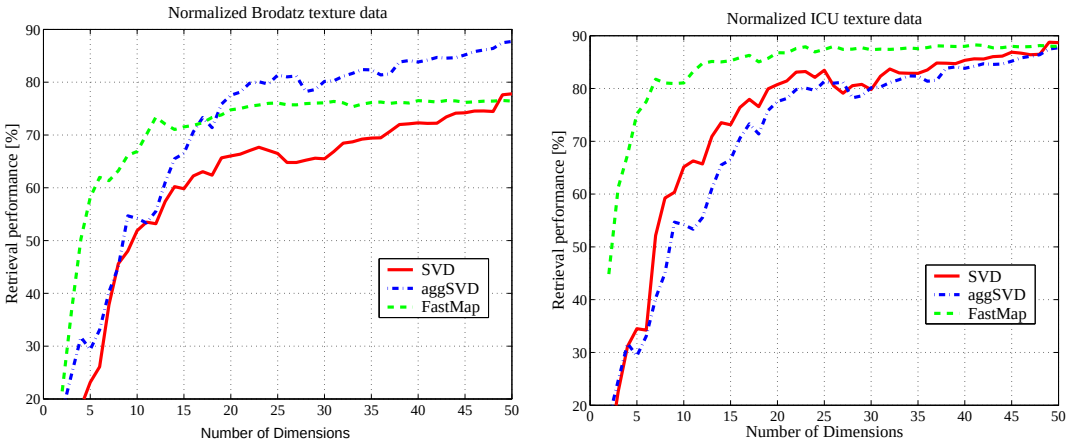
A common approach to dimensionality reduction is to identify the most important features associated with an object so that further processing can be simplified without

compromising the quality of the final results.

**Principal Component Analysis (PCA)** [62] defines a linear transformation of a set of sample points in an $M$-dimensional space such that along the new axes (or principal components, PCs) the sample variances are uncorrelated. The principal axes are found by choosing the origin to be the center of gravity for the sample point set and forming the dispersion matrix. The principal axes and the variance along each of the axes are then given by the eigenvectors and associated eigenvalues of the dispersion matrix. For many datasets, it is sufficient to consider only the first few principal components, which reduces the dimension. If the dispersion matrix is equal to a data covariance matrix, PCA method is called **Singular Value Decomposition (SVD)** [51]. In this case singular vectors are the principal components. The complexity of such transformation of data points from dimension $M$ to dimension $R$ is $O(MNR^2)$, where $N$ is the number of data points. A more pressing problem is that the SVD transformation requires complete data, and computing SVD of very large datasets is computationally complex. **Aggregated SVD** [99] approach reduces linear dependence since the SVD computation is performed on an aggregated data set. The retrieval quality will depend on how well the aggregated set represents the real one.

### 2.3.3 Evaluation

We evaluate the FastMap and SVD dimensionality reduction algorithms on two groups of the datasets. The first group of data collection has class information available



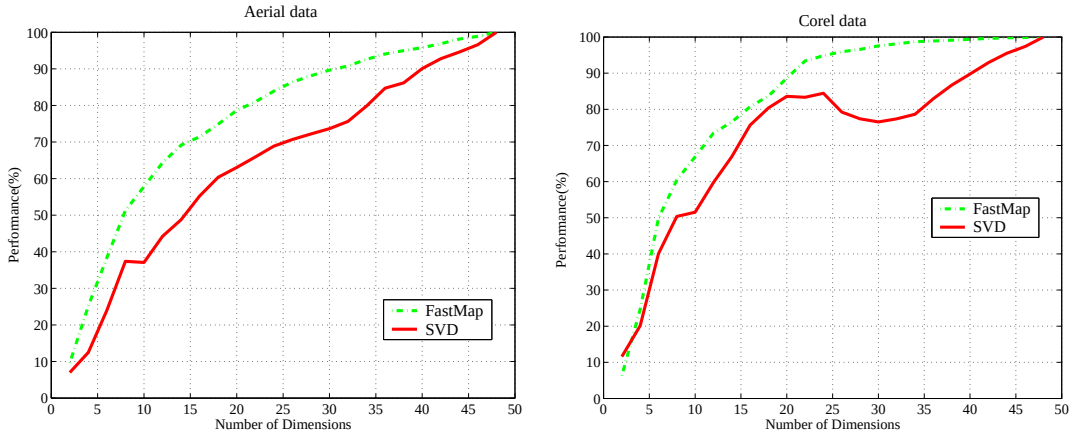

(a) Brodatz album Textures

(b) ICU Data Textures

Figure 2.1: Evaluation of dimensionality reduction algorithms on 50-dimensional MPEG-7 homogeneous texture sets where class information is available for (a) Brodatz album of 1856 images in 116 classes, and (b) MPEG-7 ICU data in 53 classes.

with respect to their texture content. One dataset consists of 50-dimensional MPEG-7 homogenous texture feature vectors similarity for 1856 images from the Brodatz Texture Album [26]. These feature vectors belong to 116 different texture classes, each consisting of 16 vectors. The second dataset consists of 50-dimensional MPEG-7 homogenous texture feature vectors for 832 images from the MPEG-7 ICU texture dataset [101]. These belong to 53 different classes, each consisting of 16 vectors. We measure the retrieval performance based on the class information in low–dimensional space, i.e. how many retrievals out of the first 16 belong to the same class.

The second group of two datasets has no class information available. The first dataset in the second group consists of 34598 48-dimensional MPEG-7 homogeneous texture feature vectors extracted from the $128 \times 128$ tiles of 40 large aerial images




(a) Aerial Image Dataset Textures

(b) Corel Textures

Figure 2.2: Evaluation dimensionality reduction algorithms on 50-dimensional MPEG-7 homogenous texture sets where class information is NOT available (a) 34598 image tiles extracted from 40 Aerial images, and (b) 26697 Corel images.

[74]. The second dataset is formed in a similar way, but the texture features were extracted from the tiles of the Corel Photobook image dataset [40], resulting in 26697 48-dimensional feature vectors. We measure the retrieval performance based on the nearest neighbor information in the full space, i.e. in low–dimensional space how many retrievals out of the first 20 belong to the nearest neighbor set of size 20 in the full dimensional space.

We measure the average query precision over all samples. SVD, aggSVD and FastMap methods are applied to all dataset in order to reduce dimensionality. For every dimension, we evaluate the retrieval precision in the object space of reduced dimensionality. Retrievals are based on the smallest $L_2$ distances from all data points in the reduced space. For the first group, where the class information is available, the

*precision* measures how many images in the retrieved set belong to the image texture class. For the second group, the *precision* measures how many images in the retrieved set are relevant with respect to the relevant image set retrieved in the original feature space.

Experiments show that, for low dimensionality, the best performance is only in the range of 40%–70%, see Figure 2.1 and Figure 2.2. The results in 2.1 also show that, if class information is available, the aggregated SVD approach proves to be as effective as the regular SVD approach. Class centroids capture a good approximated distribution of data. However, if class information is available, we can reduce the search space only to a given class and achieve better performance than by using dimensionality reduction.

The study concludes that FastMap generally performs better low–dimensional space compared to the SVD based approaches. If the reduction algorithm has a good retrieval performance, we can always incorporate an efficient low–dimensional indexing structure to achieve fast query response. As demonstrated in literature [134], the dimensionality is considered low when it is smaller than 6. In conclusion, this dimensionality reduction algorithms do not provide a satisfactory object representation for low level descriptors.

## 2.4 Clustering in High Dimensions

Cluster is defined as a group of the same or similar elements gathered or occurring closely together. If the nearest neighbor search is limited to the cluster of the query point, nearest neighbor query should return the closest points within the cluster. How-

ever, if the cluster of the query point is unknown, the choice of nearest cluster falls under the 'dimensionality curse' problem. As discussed above, recent research findings indicate that, for high–dimensional data, the concept of proximity or clustering may not be meaningful [1]. In addition, high–dimensional data is a challenge because of the inherent sparsity of the points [2]. Therefore, clustering large data sets of high dimensionality presents a serious challenge for clustering algorithms. There are a number of different clustering algorithms that are applicable to very large data sets, and a few that address high–dimensional data. In general, clustering algorithms can be divided into two groups: partitioning and hierarchical.

### 2.4.1 Hierarchical Clustering

In hierarchical clustering, data points are partitioned into successive levels of clusters, forming a hierarchical tree. At the first level all samples are grouped into singleton clusters. As we increase levels, more and more samples are clustered together in a hierarchical manner. This approach is effective, but prohibitively expensive for large datasets since the complexity is quadratic in the number of dataset items, .i.e. $O(N^2)$. Recently, some optimization techniques have been proposed for large high–dimensional datasets. **BIRCH** (Balanced Iterative Reducing and Clustering) [145] approach attempts to cluster the data in the optimal manner when a limited amount of memory is available. It uses a hierarchical cluster feature (CF) tree data structure for storing summary information about clusters of objects. The CPU and I/O costs of the BIRCH

algorithm are linear with the number of database elements, i.e. $O(N)$. BIRCH relies on the existence of the similarity measure between data points, and only performs well on data sets with spherical clusters. **CURE** (Clustering Using Representatives) [52] is an $O(N^2)$ complexity algorithm that can identify clusters of complex shapes in the presence of outliers. CURE uses a fixed number of representative points to define a cluster and handles large data sets through a combination of random sampling and partitioning. Since CURE uses only a random sample of the data set, it manages to achieve good scalability for large data sets and is more efficient than BIRCH.

### 2.4.2 Partition Clustering

Partition clustering algorithms partition the database into K groups. Clusters are typically represented by either the mean of the objects assigned to the cluster [24] or by one representative object of the cluster [65]. Each object is assigned to the closest cluster. **K-Means Clustering** originated from the generalized Lloyd's Algorithm [50], and is an iterative clustering technique that results in K data clusters. The algorithm initializes K centroids by choosing points that are mutually farthest apart. At each iteration, the algorithm recomputes a set of more appropriate partitions of the input vectors and their centroids. K-means is essentially equivalent to vector quantization [50], and its complexity is $O(KN)$. **CLARANS** (Clustering Large Applications based upon RANdomized Search) [90] is an effective practical technique that improves the scalability of K–means for large high–dimensional datasets by using a randomized and

bounded search strategy to cluster data. CLARANS assumes that the objects to be clustered are all stored in main memory. This assumption may not be valid for large databases and thus some type of spatial index structure is required.

**Self Organizing Map (SOM)** [69] is a clustering technique that preserves the topology of the space. SOM uses a number of reference vectors to describe the original space as a regular lattice associated with a high–dimensional weighting vector. Each of the high–dimensional feature vectors is assigned to one of the lattice points based on its proximity to the lattice point under some distance metric. The topological structure of the feature space is preserved in the sense that the neighboring lattice points in the SOM grid are also neighbors in the high–dimensional space. However for those vectors assigned to the same lattice point, their ordinal relationship may not be retained. The density of the reference vectors approximates the density of the input vectors for high–dimensional data. The complexity of the SOM is on the order of $O(MK^2)$, where $M$ is the vector dimension and $K$ is number of vectors. Scalability of SOM depends on the size of training dataset $K$, and its effectiveness on how well the training set represents data distribution in high dimensions.

## 2.5 Indexing in High Dimensions

There has been a considerable amount of work on high–dimensional indexing. In database research, indexing structures are used to prune the search space. Initially, traditional multidimensional data structures such as [53] and K-D tree [109], that were

designed for indexing low$_{\eta}$-dimensional spatial data, were used for indexing high$_{\eta}$-dimensional feature vectors. Recent research activities [22] reported the result that basically none of the querying and indexing techniques which provide good results on low$_{\eta}$-dimensional data perform sufficiently well on high$_{\eta}$-dimensional data for larger queries. As a result of these research efforts, a variety of new index structures and cost models have been proposed. One can broadly classify them into data partitioning (DP) and space partitioning (SP) techniques, depending on how data are grouped.

### 2.5.1 Space Partitioning Indexing Structures

Space partitioning methods divide the data space along a predefined grid regardless of the data distribution. Each node in the tree is defined by a plane through one of the dimensions that partitions the set of points into disjoint sets, each with half the points of the parent node. These children are again partitioned into equal halves, using planes through a different dimension.

A **K-D tree** [11] is a balanced multidimensional binary tree. It can be thought of as a coarse-grained density map of the distribution of data points. The database is split on the dimension with the largest variance. This continues until each node on the tree has a manageable number of objects. These objects can be stored contiguously on the physical media. Each search is first performed on the density map and outputs an estimate of the search times and data volume. Search times are expected to grow logarithmically with the number of data points. **K-D-B tree** [103] combines the

properties of search efficiency of adaptive K-D-tree and the tree balance of B tree [8]. Note that space partitioning is independent of the distance function used to compute the distance among objects in the database or between query objects and databases objects. However, like most of the K-D treeŋbased index structures, K-D-B tree suffers from such problems as no guaranteed utilization. For other space based methods like Quad Trees [46], if the dimensionality exceeds a large value (for example 60 dimensions), the whole dataset must be accessed for even very large datasets [22].

### 2.5.2 Data Partitioning Indexing Structures

Data partitioning indexes divide the data space according to the data distribution [53, 9, 17, 136, 64, 29, 33]. At the data level, the nearby data items are clustered within bounding regions defined by data nodes. At a higher level, nearby bounding regions are recursively clustered within larger bounding regions, thus forming a hierarchical directory structure. The data nodes are organized in the tree so that spatially adjacent points are likely to reside at the same node. Each directory node points to a set of subtrees, and the root node serves as an entry point for query and update processing. The nodes may overlap with each other. These index structures are height-balanced. The lengths of the paths between the root and all data pages, i.e. the height of the index, are identical, but may change after insert or delete operations.

**R-tree**[53] was designed for the management of spatial objects. In the index, objects are represented by the corresponding minimum bounding rectangles. Page regions

are treated like spatially extended, atomic objects, in their parent nodes. Therefore, it is possible that a directory page cannot be split without creating an overlap among the newly created pages. This causes a high degree of regions of overlap in high dimensions and thus low efficiency.

The **R*-tree** [9] is an extension of R-tree that minimizes the overlap between page regions, minimizes the surface of page regions, minimizes the volume covered by internal nodes, and maximizes the storage utilization. The **SS-tree** [136] uses spheres as page regions. For efficiency, the spheres are not minimum bounding spheres. Rather, the centroid point (i.e., the average value in each dimension) is used as the center of the sphere and the minimum radius is chosen such that all objects are included in the sphere. Therefore, the region description comprises the centroid point and the radius. The branches are then pruned using a heuristic method.

The **SR-tree** [64] is an extension of the R*-tree [9] and the SS-tree. The distinctive feature of the SR-tree is the combined utilization of bounding spheres and bounding rectangles. This improves the performance on nearest neighbor queries by reducing both the volume and the diameter of regions compared to R*-tree and SS-tree. **X-tree** [17] extends the R*-tree in two ways: (1) overlap-free split according to a split-history and, (2) supernodes with an enlarged page capacity. The number and the size of supernodes created increase with dimensionality. To operate on low–dimensional spaces efficiently, the X-tree split algorithm also includes a geometric split algorithm.

**Hybrid tree** [29] splits a node using a single dimension, but the indexed subspaces

may overlap. Hybrid tree always chooses the dimension that has the largest extent as the dimension to split. It implicitly eliminates "non-discriminating" dimensions and, therefore, most of the empty space. The tree operations (insertions, deletions and updates) are similar to the R-tree. The indexed subspaces have K-D tree-based organization exploited to achieve faster intra-node search. In **Pyramid Tree** [16], Berchtold et al. proposed a special partitioning strategy (Pyramidŋ-Technique) that divides the data space first into 2D pyramids, and then cuts each pyramid into several slices. They also proposed algorithms for processing range queries on the space partitioned by this strategy. However, the shape of queries used in similarity search is not hypercube, but hypersphere.

It has been shown [134, 22] that the data and space partitioning methods ultimately degenerate to sequential scans through leaf nodes if the number of dimensions in the data space becomes large. In these studies, measures of efficiency for indexing structures used for nearest neighbor search were number of vectors/leaf nodes visited and (I/O) and CPU complexity measured by the elapsed time for NN search. Note that direct sequential scan is faster than the random data access. We can safely assume that a linear scan costs as much as 10% [134] to 15% [19] of a random examination of data pages. Based on reported results [22] for the presented indexing structures, linear scan outperforms data partitioning indexing methods [136, 64, 17] for dimensions larger than 10.

Data partitioning indexing structures tend to have high degrees of overlap between bounding regions in higher dimensions, causing degradation of the performance of query processing in high-ŋdimensional data spaces. In addition to the above drawbacks, these index structures have the well known disadvantages of multidimensional index structures, such as high costs for insert and delete operations, and the effectiveness of many of these indexing structures is highly data dependent and difficult to predict [22].

### 2.5.3 Compression based Indexing

As reported [134], often a simple linear scan of all the database items is cheaper than using an index based search in high dimensions. Quantization-based indexing methods attempt to profit from this by maximizing the efficiency of the linear scan approach, and in the following we present several examples of compression based indexing schemes.

**Vector Approximation File (VA-file)** [134] index is a compression-based architecture where the feature space is quantized by separate quantization of each dimension. Search mechanism is based on sequential searching over a compressed representation of the database items. The compression of the feature space enables more items to be loaded into main memory for faster access. Sequential search over quantized data reduces the search complexity by filtering out a good fraction of the data. Only the remaining fraction of the actual feature vectors is accessed, thus reducing the number of page accesses. The evaluation of VA-file demonstrates that VA-File outperforms most of the tree based index structures in high dimensions [22].

The **Vector Quantization File (VQ)** [126] technique was proposed for limited memory applications. Here, the data is partitioned using the accumulated query history and each partition of data points is separately compressed using a vector quantizer tailored to its distribution. The VQ techniques inherently provide a spectrum of points to choose from. This property is especially crucial for large multimedia databases where I/O time is a bottleneck because it offers the flexibility to trade time for better accuracy.

**Independent Quantization Tree (IQ)** [13] tree claims to be the "best of both worlds" of indexing approaches. It employs a three-level index structure, using minimum bounding rectangles and vector approximations. The first level is a regular (flat) directory consisting of minimum bounding boxes, the second level contains data points in a compressed representation, and the third level contains the actual data. The main technical challenge of the IQ-tree is how to determine an optimal compression rate for each page.

For multimedia databases, compression based methods have certain advantages. First, the construction of the approximation is based on the feature values in each of the dimensions independently. As a result, the compressed representation domain can support query search for different similarity measures. Secondly, the construction of the approximation can be made adaptive to the dimensionality of the data.

## 2.6 Discussion

In this chapter, we introduced the curse of dimensionality and its broad impact on large multimedia datasets management, in particular, data access, retrieval, and summarization.

The three major approaches to overcoming the dimensionality curse are data sampling, and in particular data clustering, and constrained and approximate nearest neighbor search. An increase in dimensionality can often be helpful to mathematical analysis. Simply, we could say that in many cases, there are really "few things that matter" [2]. In addition, careful feature selection and scaling of the inputs fundamentally affects the severity of the problem.

# Chapter 3

# Efficient Access for Gabor Texture Features

There has been considerable work in the past decade on the use of low level visual features for content based image and video retrieval. The focus has been on finding good visual descriptors for similarity search [47, 75, 116]. Typical image/video descriptors are of high dimensionality (from tens to several hundreds of dimensions). Color, texture, shape, and motion descriptors are some of the commonly used low-level visual descriptors for image and video data [78]. For example, a 80-dimensional edge histogram descriptor is often used to characterize the shape in a given image. The feature space grows exponentially with the number of descriptor dimensions, and the search complexity increases at the same rate. The curse of dimensionality is hard to avoid since the high dimensionality and computational complexity of descriptors adversely affect the efficiency of content-based retrieval systems. There has been a considerate amount of work done in the area of determining the meaningful dimensions of feature descriptors by using traditional dimensionality reduction approaches [99, 140, 145].

As demonstrated in Chapter 2, these approaches do not provide a satisfactory representation for low level descriptors, since the precision rate during retrieval is low.

In this chapter, we re-examine a MPEG-7 [78] texture descriptor, and propose a modified descriptor based on how the original texture descriptor was derived. This new descriptor has comparable performance, but with half the dimensionality and less computational expense. We also propose a new normalization scheme that improves similarity retrieval.

## 3.1 Image Texture

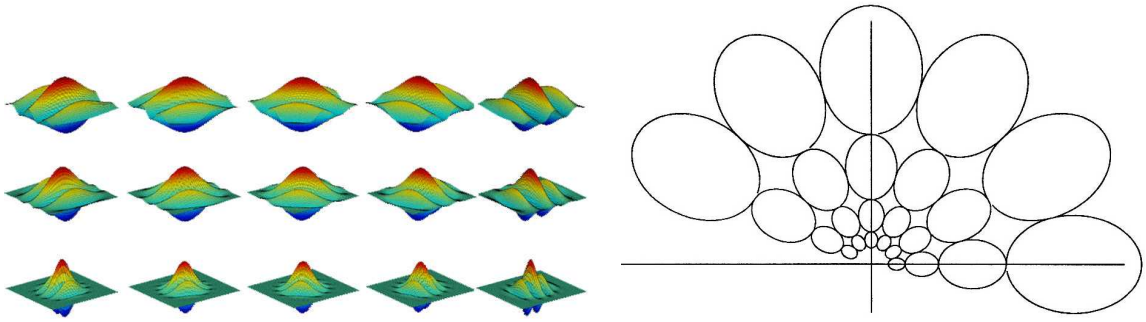
Image texture has emerged as an important visual feature for image/video information retrieval applications in remote sensing, medical diagnosis and quality control. Image texture is useful in a variety of tools, such as segmentation and similarity retrieval, and has been the subject of intense study by many researchers [77, 125]. In a very general sense, image texture is the appearance of spatial distribution of basic patterns. Image texture can be observed in a wide variety of image objects: images of water, grass, clouds, a pattern on a fabric, woven aluminum wire, brick walls, and handwoven rugs, are some examples of image texture. There have been several attempts to model texture using statistical methods [56], as well as random field model-based methods [72, 80], geometrical methods [125], and multiscale filtering techniques [57, 30, 76].

## 3.2 A Homogeneous Texture Descriptor

In a very general sense, texture in an image is the appearance of spatial distribution of basic patterns. Regular spatial distribution of these patterns form a homogeneous texture while a random distribution produces a wide range of non-homogeneous texture patterns. Texture descriptors characterize the texture pattern in an image and are used to compute similarity between two images. Our focus here is on a descriptor that is based on multiresolution Gabor filtering. This descriptor is referred to in the following as the Homogenous Texture Descriptor (HTD). The choice of HTD is motivated by several factors. Daugman [38] proposed the use of Gabor filters in the modelling of the receptive fields of simple cells in the visual cortex of some mammals, and showed that for two–dimensional Gabor functions, the uncertainty relations attain the minimum values [38]. Later, Gabor filter outputs were shown to be well suited for texture segmentation and analysis [80, 125]. In addition, a Gabor-based homogeneous texture descriptor has been adopted by the MPEG-7 standard for its effectiveness and efficiency [78].

### 3.2.1 Gabor filter bank

The Gabor filter can be viewed as a sinusoidal 2D plane of a particular frequency and orientation, modulated by a Gaussian envelope, see Figure 3.1(a). Therefore, a Gabor filter can be considered as an orientation- and scale-tunable edge and line detector. Gabor filters do bear some similarity to Fourier filters. However, Gabor



(a) Spatial Domain (b) Polar Frequency Domain

Figure 3.1: (a) Gabor filters in the spatial domain for K=5 orientations and S=3 scales, and (b) Gabor channel contours in the polar frequency domain; the contours represent half peak magnitude of a Gabor filter response for a=2, K=6 orientations, and S=4 scales.

filters are limited to certain frequency bands by the Gaussian damping terms, and are bandpass filters. A two-dimensional Gabor function is defined as:

$$g(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)+2\pi jWx\right], \tag{3.1}$$

where $W$ is the frequency of the modulated sinusoid and $j=\sqrt{-1}$. Gabor functions form a non-orthogonal basis set for the multi-resolution decomposition of any two–dimensional function $I(x,y)$.

A class of self-similar Gabor functions, also known as Gabor wavelets, is derived from the mother wavelet $g(x,y)$ by appropriate dilations and rotations [76]:

$$\begin{aligned} g_{mn}(x,y) &= a^{-m}g(x',y') \qquad\qquad a>1, \quad m,n\in Z \\ x' &= a^{-m}(x\cos\theta+y\sin\theta) \quad y' = a^{-m}(-x\sin\theta+y\cos\theta), \end{aligned} \tag{3.2}$$

where $m = 1,..,S$, $S$ is the total number of scales. The scale factor $a^{-m}$ in (3.2)

ensures that the energy $\varepsilon = \sum_x \sum_y |g_{mn}(x,y)|^2$ is independent of $m$, i.e., all the filters in the dictionary have the same energy. If $K$ is the total number of orientations then $\theta = n\pi/K$, $n \in [0, 1, 2, \ldots, K-1]$. The non-orthogonality of the Gabor wavelets results in redundant information in the filtered images. Usually, given $K$ and $S$, $a$ is chosen to ensure that the half-peak magnitude supports of the filter responses in the frequency spectrum touch each other, see Figure 3.1(b).

Given an image $I(x,y)$, its Gabor wavelet transform $t_{mn}(x,y)$ is defined by:

$$t_{mn}(x,y) = |g_{mn}(x,y) * I(x,y)|\,, \tag{3.3}$$

where $*$ denotes convolution in spatial domain. This wavelet decomposition allows a compressed presentation of the intensity image data. Given that the total of $S$ scales and $K$ resolutions are used for this multi-resolution decomposition, the number of filtered images $t_{mn}$ is $S \times K$.

## 3.3 Statistics of the filter outputs

Gabor filters detect perceptually significant features at various spatial scales and orientations. The statistics of these micro features characterizes the underlying texture information. Let $A$ be the number of output coefficients, $A = \sum_x \sum_y 1$. The mean $\mu_{mn}$ and the square root of the standard deviation $\sigma_{mn}$ of the magnitude of the transform coefficients $|t_{mn}|$ are used to form one component of the HTD:

$$\mu_{mn} = \tfrac{1}{A} \sum_x \sum_y |t_{mn}(x,y)|\,, \quad \sigma_{mn}^2 = \tfrac{1}{A} \sum_x \sum_y (|t_{mn}(x,y)| - \mu_{mn})^2\ . \tag{3.4}$$

Note that, for Gabor filtering, the window size is pre-determined based on the central frequency of the filter. The window size in MPEG-7 is 64, 128 or 256 pixels wide. Formally, the HTD is a vector formed for $S$ scales and $K$ orientations given by:

$$\vec{f}_{\mu\sigma} = [\mu_{11}, \sigma_{11}, \mu_{01}, \sigma_{01}, ..., \mu_{SK}, \sigma_{SK}]. \tag{3.5}$$

A Gabor-based HTD has been adopted to the MPEG-7 standard [78] for its effectiveness and efficiency [76]. Let $\mu_I$ and $\sigma_I$ denote the mean and the standard deviation of the intensity of image $I(x, y)$:

$$\mu_I = \frac{1}{N}\sum_x\sum_y I(x,y) \qquad \sigma_I^2 = \frac{1}{N}\sum_x\sum_y (I(x,y) - \mu_I)^2, \tag{3.6}$$

$$\vec{f}_{MPEG-7} = [\mu_{11}, \sigma_{11}, \mu_{12}, \sigma_{12}, ..., \mu_{SK}, \sigma_{SK}, \mu_I, \sigma_I]. \tag{3.7}$$

Note that the dimensionality of $f_{\mu\sigma}$ is $2 \times S \times K$ and the dimensionality of $\vec{f}_{MPEG-7}$ is $2 \times S \times K + 2$.

### 3.3.1 Rician model for Gabor filter outputs

Dunn and Higgins [42] provide analytical evidence that the Gabor filter output can be modeled in general was as a Rician random variable. This approach is based on adopting the same principles for texture filtering models as already exist for communication receiver models [58].

As shown in (3.1), the Gabor filter is a Gaussian function modulated by a complex sinusoid. Its frequency response has a Gaussian shape. So the nature of the Gabor filter is a band-pass filter. Consider the 1-D case for simplicity, and let $I(x)$ be uniform

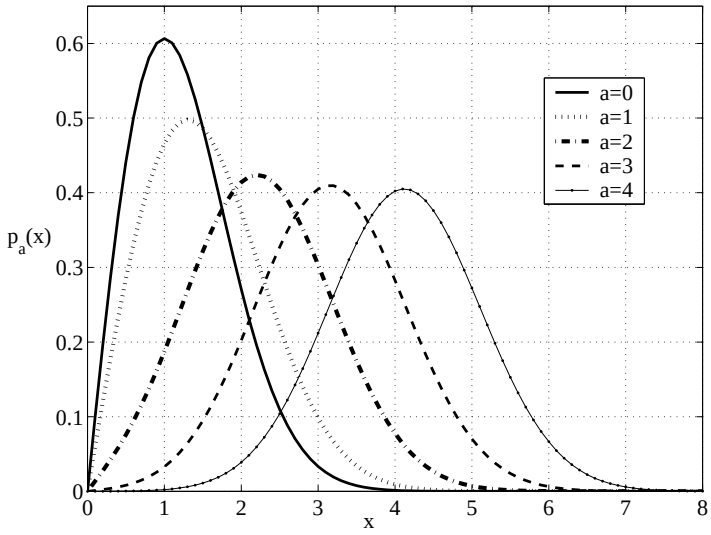



Figure 3.2: The Rician Distribution: $p_a(x) = xe^{-\frac{1}{2}(x^2+a^2)}I_0(ax)$.

texture. Thus, $I(x)$ is a periodic signal and can be represented as a sum of sinusoids [58]. In the frequency domain, the Fourier transform of $I(x)$ is $I(u)$. $I(u)$ consists of a periodic collection of impulses. So, the Fourier transform $t(u)$ of the Gabor filter output $t(x)$, of (3.3) consists of impulses, and thereby $t(x)$ is a sum of sinusoids. The main result is that, if a texture is uniform, Gabor filtering will produce an output that is sinusoidal with amplitude $U$. If the texture is non-uniform, it can be viewed as a homogeneous texture with random perturbations. Perturbations are modeled as white Gaussian noise, and the noise output after Gabor band-pass filtering is narrow band noise $n(x)$. $n(x)$ is a Gaussian zero-mean, wide-sense stationary noise with variance $\sigma^2$. Let $n_I(x)$ and $n_Q(x)$ be the in-phase and quadrature component, respectively, of $n(x)$, and:

$$n(x) = n_I(x)cos(ux) - n_Q(x)sin(ux). \tag{3.8}$$

If we represent the non-uniform texture as a sum of sinusoids and a white Gaussian

noise, the filtering Gabor filter output is:

$$\begin{array}{lll} t(x) = & |U_o cos(ux) + n(x)| & = |n'_I(x)cos(ux) - n_Q(x)sin(ux)| \\ t(x) = & |r(x)cos(ux + \phi(x))| = & r(x) \end{array} . \tag{3.9}$$

Note that $n'_I(x) = U + n_I(Q)$ and that $n_I(x)$ and $n_Q(x)$ have independent, zero mean Gaussian distribution with variance $\gamma^2$ [58]. The magnitude of this narrow-band noise signal is given by:

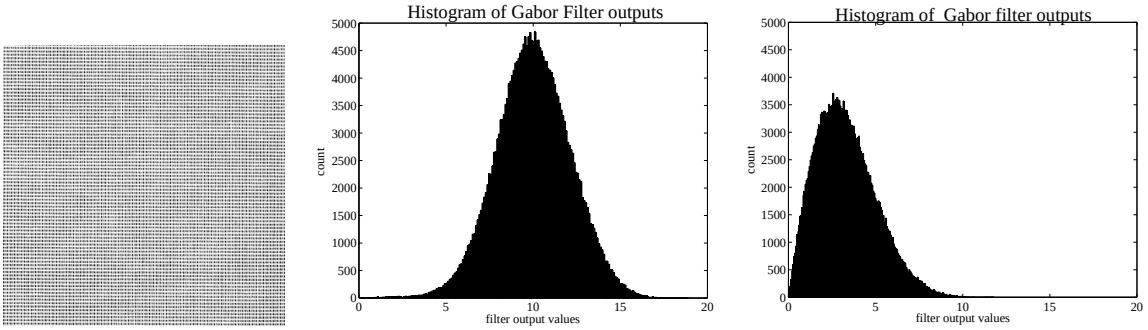
$$r(x) = \sqrt{{n'_I}^2(x) + n_Q^2(x)}. \tag{3.10}$$

The probability distribution of $r(x)$ is Rician, and it was formulated in [100] as

$$p_a(r) = \frac{r}{\gamma^2} \exp\left(-\frac{r^2 + U^2}{2\gamma^2}\right) I_0\left(\frac{Ur}{\gamma^2}\right), \tag{3.11}$$

where $I_0(x)$ is the zero-order modified Bessel function of the first kind. Since $r(x)$ is Rician from (3.11), $t(x)$ is Rician for any $x$. A normalized Rician distribution is shown in Figure 3.2, where $a = \frac{U}{\gamma}$ is the signal-to-noise ratio and $x = \frac{r}{\gamma}$ is the normalized envelope of the signal. The Rician distribution in (3.11) can vary from a Rayleigh distribution for low signal-to noise ratio ($a \approx 0$) to an approximately Gaussian distribution for large $a$ ($a \gg \gamma$), see Fig. 3.2. The Rician model provides considerable insight to the nature of the filter outputs, as we discuss next.

The design of (3.5) for the HTD is driven by the implicit assumption that the filter outputs have a Gaussian-like distributions. If the texture is homogenous, well-defined and periodic for scale $s$ and orientation $k$, the filter output histogram demonstrates a Gaussian-like distribution, as shown in Figure 3.3(b). Therefore, each of these Gaussian-like distributions is well described completely by mean and standard



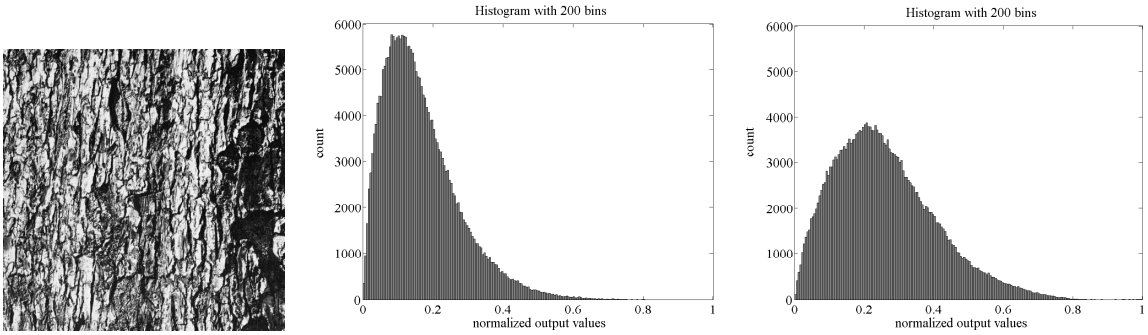

(a) Image 21 (b) Histogram for s=3, k=3 (c) Histogram for s=1, k=2

Figure 3.3: (a) Brodatz Image 21 with regular texture pattern, and histograms of its Gabor filter output values for $g_{s,k}$ (b) s=3, k=3, and (c) s=1, k=2.

deviation. Gaussian distribution approximates the Rician distribution well for high signal-to-noise ratio. This approximation showed to be effective for similarity retrieval experiments over well-defined Brodatz texture dataset [74]. But, even for the well-defined textured image in Figure 3.3(a), the Gaussian model does not describe the output distribution of the majority of Gabor filter well. In the example, over 2/3 of the filter outputs have distributions like the one in Figure 3.3(c). In that case, the Gaussian approximation of the Rician distribution is not a good model. These filter outputs are better described using a Rayleigh approximation of the Rician distribution, see Figure 3.2.

### 3.3.2 Modified texture descriptor

Textures typically do exhibit some underlying structures. However, as discussed for the example of Figure 3.3 in the previous section, *unless* the texture structure is




(a) Image 12 (b) Histogram for s=3, k=3 (c) Histogram for s=2, k=5

Figure 3.4: (a) Brodatz Image 12 with irregular texture pattern, and istograms of its Gabor filter output values for $g_{s,k}$ (b) s=1, k=3, and (c) s=2, k=5.

well-defined and periodic *and* a Gabor filter tuned to that pattern is used, the Gaussian approximation o the Rician distribution of the filter outputs is not accurate. Further, if an image contains complex and/or noisy textures, like the Image 12 from the Brodatz album, shown in Figure 3.4(a), even the outputs of 'tuned' filters exhibit Rayleigh like distribution as shown in Figure 3.4(b). One explanation is that many textures are neither uniform nor homogenous, and contain significant random effects or complex patterns. This results in large perturbations for the model and the signal-to-noise ratio is smaller. In typical experiments, the distributions of filter outputs are usually like the one in Figure 3.4(c).

Over a wide range of textures, the probability that a given texture is uniform and that it has a strong component at a specified center frequency, is small. Thus, the Rayleigh approximation is valid with a higher probability than the Gaussian approximation for the filter outputs that have Rician distributions. This claim is consistent

with the results reported in [135] as well as with experimental results presented in this chapter.

Note that the HTD descriptor in (3.5) was derived from the Gaussian approximation. However, when the output signal-to-noise ratio of the filter is low, the Rician distribution significantly deviates from the Gaussian approximation. Parameter estimation for the Rician distribution is a complex problem that has attracted much of attention in the medical imaging community [112]. Here, we adopt another approximation: Rayleigh distribution is a Rician distribution for $a = 0$ (3.11) as seen in Figure 3.2:

$$p_0(r) = \frac{r}{\gamma^2} \exp\left(-\frac{r^2}{2\gamma^2}\right). \tag{3.12}$$

The Rayleigh distribution has only one parameter, $\gamma$. We can estimate $\gamma$ using the maximum likelihood estimator [118]. If $t(x, y)$ corresponds to the Gabor filter output, and $A$ is the number of output coefficients, the maximum likelihood estimate of $\gamma$ is

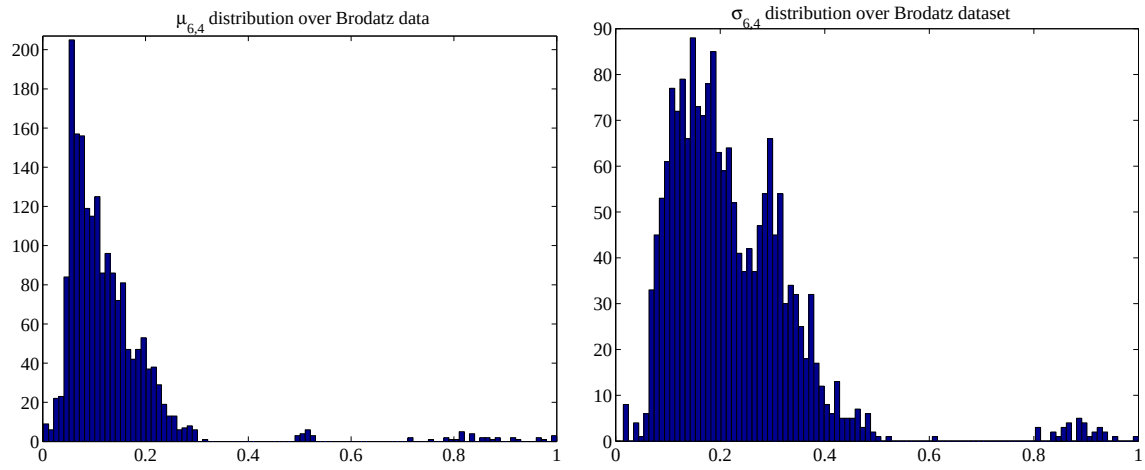
$$\gamma^2 = \frac{1}{2A} \sum_x \sum_y |t(x, y)|^2 \Rightarrow \gamma_{mn}^2 = \frac{1}{2A} \sum_x \sum_y |t_{mn}(x, y)|^2, \tag{3.13}$$

where $\gamma_{mn}$ is the Rayleigh parameter of the output distribution when the Gabor filter $g_{mn}$ is applied. The modified texture descriptor then is $\vec{f_\gamma}$:

$$\vec{f_\gamma} = [\gamma_{00}, \gamma_{01}, ..., \gamma_{s-1k-1}]. \tag{3.14}$$

This descriptor has $S \times K$ dimensions, which is half the number of dimension of $\vec{f_{\mu\sigma}}$ (3.5). Moreover, if we have pre-computed $\vec{f_{\mu\sigma}}$ (or MPEG-7) features, it is easy to compute $\vec{f_\gamma}$ without having to perform the filtering step using (3.4):

$$\gamma_{mn}^2 = \frac{1}{2}\left(\mu_{mn}^2 + \sigma_{mn}^2\right). \tag{3.15}$$




(a) $\mu_{64}$ histogram

(b) $\sigma_{64}$ histogram

Figure 3.5: Histograms with 100 bins of the values along s=6, k =4 of $\vec{f}_{\mu\sigma}$ of (a) $\mu_{6,4}$, and (b) $\sigma_{6,4}$ data.

Thus we can compute the new features from the old ones without having to repeat the computationally expensive filtering step. This observation is significant because many databases already use MPEG-7 texture features for content-based access. A 50% reduction in dimension results in significant savings in the storage of feature vectors and a reduction of the computational complexity for similarity search in large image datasets [83].

## 3.4 Distribution along dimensions

In Section 3.3 we showed that the distribution of Gabor filter outputs $t(x, y)$ can be modeled well by the Rician distribution. However, it was shown that the estimation of Rician parameters is not practical to [112], especially the variance parameter.

Therefore, Gaussian or Rayleigh parameters are used to describe Gabor filter outputs in a feature vector.

We now study the distributions along each dimension of the $\vec{f_\gamma}$ and $\vec{f_{\mu\sigma}}$ parameters of (3.14) over a large image dataset. In order to get a texture information from the image, we filter the $i^{th}$ image from the database with Gabor wavelet $g(m,n)$ and use the statistics of the filter output as the descriptor element of (3.13). This random variable is the square root of the sum of squared filter outputs. Note that the magnitudes $|t(x,y)|^2$ of the filter outputs are shown to have a Rician distribution. Also, from (3.10),

$$|t(x,y)|^2 = {n'_I}^2(x,y) + n_Q^2(x,y). \tag{3.16}$$

Note that $n'_I(x)$ and $n_Q(x)$ have independent, Gaussian distributions with means $U$ and 0 respectively, and variance $\sigma$. Therefore:

$$\gamma = \frac{1}{\sqrt{2A}}Z, \qquad Z = \sqrt{\sum_x \sum_y |t(x,y)|^2} = \sqrt{\sum_x \sum_y ({n'_I}^2(x,y) + n_Q^2(x,y))}, \tag{3.17}$$

where $Z$ is the magnitude of a sum of 2A Gaussian distributed random variables, and its underlying distribution is the *generalized Rice* distribution [4].

If $U_i$ is the mean value of $n'_I$, and $U_o^2 = \sum_i U_i^2$, pdf of $Z$ is [4]:

$$p_Z(z) = \frac{z}{\sigma^2}\left(\frac{z}{U_o}\right)^{A-1} exp(-\frac{z^2+U_o^2}{2\sigma^2}) I_{A-1}\left(\frac{zU_o}{\sigma^2}\right). \tag{3.18}$$

If signal-to-noise ratio is high, generalized Rice distribution can be approximated with a Gaussian (see Appendix A):

$$p_Z(z) \sim \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(z-U_o)^2}{2\sigma^2}}. \tag{3.19}$$

Therefore, for large SNR, $\hat{\gamma}$ has a Gaussian distribution with mean $U_o$ and variance $\sigma$, i.e. $\hat{\gamma} \sim \mathcal{N}(U_o, \sigma)$. As demonstrated Section 3.3.2, low signal-to-noise ratio is more frequent scenario for filtered textures. Thus, the distribution of $Z$ can be approximated by a generalized Rayleigh distribution. General Rayleigh distribution is a $\chi$-distribution (Chi) with scale parameter equal to 1:

$$p_Z(z) = \frac{2z^{2A-1}}{(2\sigma^2)^A \Gamma(A)} e^{-\frac{z^2}{2\sigma^2}} = \frac{1}{\sigma^2} \chi(\frac{z}{\sigma}, 2A). \tag{3.20}$$

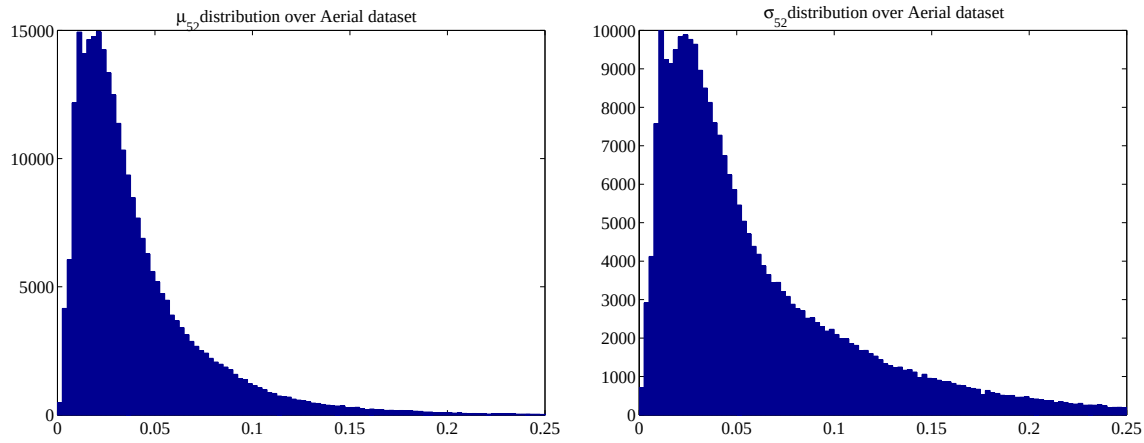
Therefore, for low signal-to-noise ratio $\gamma \sim$ Rayleigh. Over a large dataset, the values along each dimension of both $f_\gamma$ and similarly $f_{\mu\sigma}$ follow a skewed distribution that can be approximated with Rayleigh distribution [135]. This argument is supported by experimental observations demonstrated in Figures 3.5 and 3.6.

We can estimate the Rayleigh distribution parameter $\hat{\gamma}$ using maximum likelihood estimation [118]. If $f_{ij} = \gamma_{mn}$ is a Rayleigh parameter for fixed $m$ and $n$ along one dimension $j = mK + n$ of all the images $i$ from the database, $i = 1, \ldots, N$, the maximum likelihood estimate of the parameter $\gamma_j$ is $\hat{\gamma_j}$:

$$\hat{\gamma_j^2} = \frac{1}{2N} \sum_i |f_{ij}|^2. \tag{3.21}$$

### 3.4.1 Similarity Measure and Normalization

We have that the homogeneous texture in an image can be described with a feature vector whose elements are the statistics of the Gabor filtered image via equations (3.5), (3.7), and (3.12). Similarity between two textures is measured by the distance between




(a) $\mu_{64}$ histogram

(b) $\sigma_{64}$ histogram

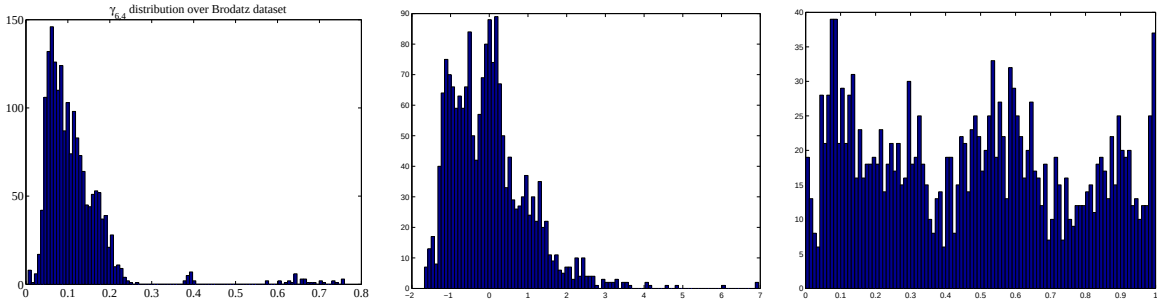
Figure 3.6: Histograms with 400 bins of the values along s=6, k =4 of $\vec{f}_{\mu\sigma}$ of (a) $\mu_{6,4}$, and (b) $\sigma_{6,4}$ for Aerial set of features data.

the corresponding feature vectors. Ideally, the distance computed from these feature vectors should capture the similarity of the underlying textures. Typically, the texture feature characteristics in this space are captured using $L_p$ norms, $p = 1, 2, \ldots, \infty$.

Let $\vec{f}_i = [f_{i1}, f_{i2}, \ldots, f_{iM}]^T$ be the corresponding texture feature vector of the $i^{th}$ image pattern. The $L_p$ distance between two feature vectors $\vec{f}_i$ and $\vec{f}_j$ is thus defined as:

$$d(i,j) = \sqrt[p]{\sum_{k=0}^{M}(f_{ik} - f_{jk})^p}. \tag{3.22}$$

Detailed experimental evaluations of $L_p$ distance measures suggest that the $L_2$ (Euclidean) or $L_1$ distance measures capture the similarity best [110]. However, if we measure the distance between two raw vectors using the $L_2$ or $L_1$ norm, those feature components that have a large dynamic range are likely to dominate the measure. In order to provide more objective comparison, we normalize the feature vectors to obtain




(a) $\vec{f_\gamma}$ (b) $\vec{f}_\gamma^{(SN)}$ (c) $\vec{f}_\gamma^{(SN)}$

Figure 3.7: Histograms with 100 bins for a Brodatz album data of values along of $\vec{f_\gamma}$ of $\gamma_{6,2}$: (a)$\vec{f_\gamma}$, (b) $\vec{f}_\gamma^{(SN)}$, and (c) $\vec{f}_\gamma^{(RE)}$.

better results [74]. The descriptors are normalized separately so that each dimension has the same dynamic range. Since we observed two different approximations of the distributions of parameters $f_\gamma$ and $f_{\mu\sigma}$ along each dimension in Section 3.4, we propose a new normalization Rayleigh Equalization (R.E.) method and compare it with existing Standard Normalization (S.N) approach.

**Standard Normalization** [74] assumes the Gaussian approximation of component distributions along each dimensions. Note that Gaussian cumulative distribution function (CDF) does not have a closed form. Therefore, we scale the values along each of the dimensions to have zero mean and unit variance. The dynamic range is thus the same, and the components should contribute to the overall distance measure equally. The component–wise average and standard deviation of the $N$ feature vectors along dimension $j$ are:

$$\bar{\mu_j} = \frac{1}{N}\sum_{i=0}^{N} f_{ij} \quad \text{and} \quad \bar{\sigma_j} = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(f_{ij} - \bar{\mu_j})^2} \rightarrow f_{ij}^{(SN)} = \frac{f_{ij} - \mu_j}{\sigma_j}. \tag{3.23}$$

**Rayleigh Equalization** adopts the Rayleigh approximation of component distribution along each dimensions as shown in Section 3.4. Rayleigh equalization forces the distribution along dimensions to be uniform on the segment $[0, 1]$ using the Rayleigh CDF:

$$f_{ij}^{(RE)} = 1 - e^{-\frac{f_{ij}^2}{2\hat{\gamma}_j^2}}, \tag{3.24}$$

where $\hat{\gamma}_j$ is estimated using (3.21). Rayleigh equalization enforces a more uniform data distribution, see Figures 3.7. This enables the use of uniform space partitioning instead of data partitioning in high–dimensional space, thus resulting in lower indexing complexity and overhead.

### 3.4.2 Evaluation

Although research on image retrieval has been actively pursued for more than a decade, less work has been done on how to effectively evaluate such retrieval techniques and systems. A typical procedure for evaluation is to (1) set up an image testbed; (2) define a query set and a ground truth for each query image; (3) use these metrics to compare the effectiveness of different approaches, and (4) measure how many images are relevant in the retrieved set (precision) and how many of the relevant images in the collection were retrieved (recall). Thus, typical performance evaluation of retrieval approaches are highly dependent on the nature of the selected image dataset and query set selected. There is no known metric which is independent of any query set while accurately depicting the performance of the image retrieval system.

With the above in mind, we compare the similarity retrieval performance of $\vec{f}_{\mu\sigma}$ and $\vec{f}_{\gamma}$ on the Brodatz texture dataset [26]. The dataset consists of 1856 images (16 subimages from each of 116 texture classes). Since we consider 5 scales and 6 orientations, the dimensionality of $\vec{f}_{\mu\sigma}$ is 60 and $\vec{f}_{\gamma}$ is 30. Each texture descriptor $\vec{f}$ from both sets is used in turn as the query. Retrievals are based on the smallest $L_2$ distances from $f$ in the descriptor space. Let $T(\vec{f})$ be the retrieved set with cardinality $T$, and $C(\vec{f})$ be the collection of images relevant to $\vec{f}$. The *precision* measures how many images are relevant in the retrieved set as:

$$P(\vec{f}) = \frac{\left|C(\vec{f}) \cap T(\vec{f})\right|}{\left|T(\vec{f})\right|}, \tag{3.25}$$

and the *recall* measures how many of relevant images we retrieved:

$$R(\vec{f}) = \frac{\left|C(\vec{f}) \cap T(\vec{f})\right|}{\left|C(\vec{f})\right|}. \tag{3.26}$$

where $|\cdot|$ denotes cardinality of the set.

Figures 3.8 and 3.9 show the precision vs. recall curves for each descriptor and normalization method for the $L_1$ and $L_2$ distances, respectively. The curves are plotted by averaging precision and recall over all $t$, for different size of retrieved set, ranging from $T = 1$ (high precision, small recall) to $T = 48$ (small precision and high recall). While the dimensionality of $\vec{f}_{\gamma}$ is smaller by 50%, the drop in precision (equivalently, the increase in error rate) is around 5% on average for $L_1$ and 6.6% for $L_2$ over 48 different values of $T$.

Use of Rayleigh equalization method increases the retrieval precision for both the $L_1$ and $L_2$ distance measures, and $\vec{f}_{\mu\sigma}$ and $\vec{f}_{\gamma}$ datasets, and it is superior to standard

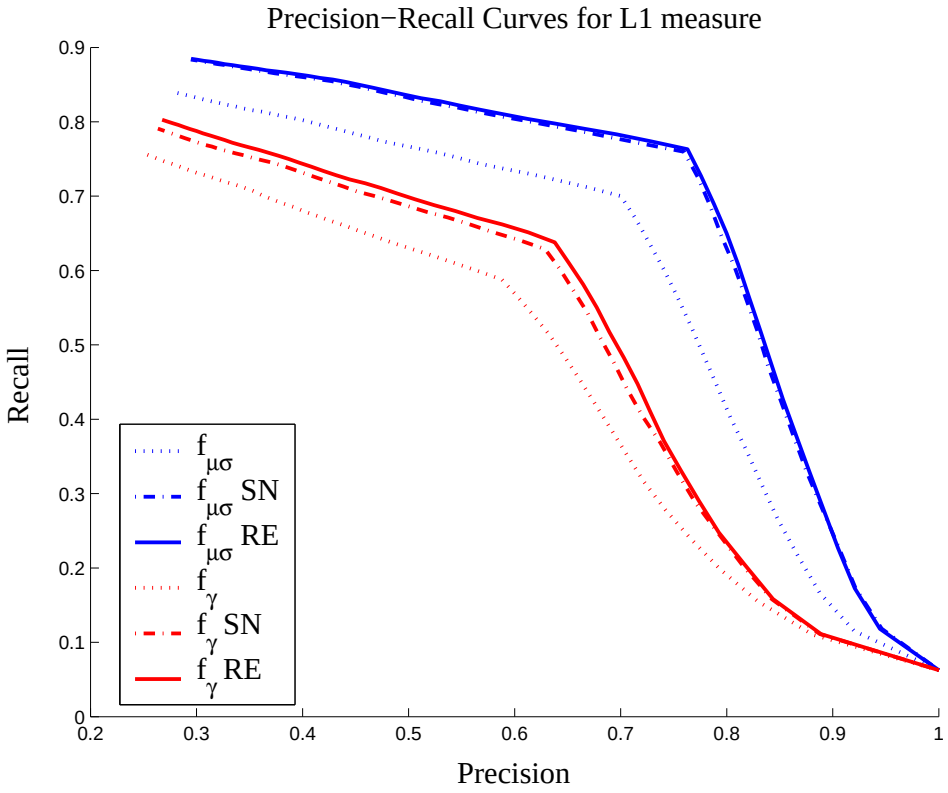



Figure 3.8: Precision vs. Recall curves for $L_1$ distance measure over the Brodatz album.

normalization method. This confirms our claim that the Rayleigh distribution assumption along feature dimensions is valid with high probability when we consider a wide range of textures.

## 3.5 Discussion

When texture images are processed through Gabor filters, the filter outputs have a strong tendency to follow a Rayleigh distribution. Based on this, we modified the MPEG-7 homogeneous texture descriptor resulting in lower dimensionality and com-

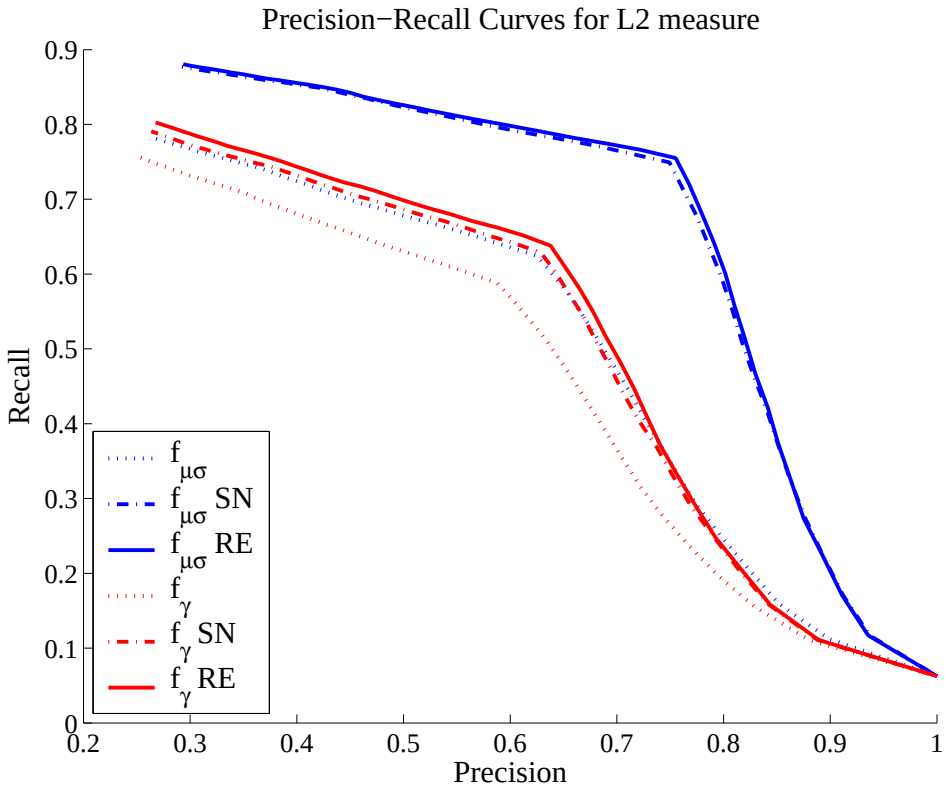



Figure 3.9: Precision vs. Recall curves for $L_2$ distance measure over the Brodatz album.

putational complexity. This benefits content-based retrieval systems by significantly reducing storage, computational expense, indexing overhead, and retrieval time. We support this approach by demonstrating that the new descriptor performs comparably with the MPEG-7 descriptor.

Another observed phenomenon is that the values along each dimension of both descriptors follow a generalized Rice distribution that can be modeled well by a generalized Rayleigh pdf. Exploiting this behavior, we proposed a new normalization for the Homogenous Texture descriptor databases. We demonstrated that the new normalization scheme improves similarity retrieval performance.

# Chapter 4

# Quadratic Distance Queries for Relevance Feedback

In this chapter, we present an efficient approach to relevance feedback search in high-dimensional feature space. Relevance feedback learning is a popular scheme used in content based image and video retrieval to support high-level "concept" queries. This work addresses those scenarios in which a similarity or distance matrix is updated during each iteration of the relevance feedback search, and a new set of nearest neighbors are computed. Repetitive nearest neighbor computation in high-dimensional feature spaces is expensive, particularly when the number of items in the data set is large (hundreds of thousands). In this context, we present a scheme that exploits correlations between two consecutive nearest neighbor sets and significantly reduces the overall search complexity. We show that vector quantization–based indexing structures can support relevance feedback and suggest a modification to an existing nearest neighbor search algorithm to support relevance feedback for the weighted Euclidean and quadratic distance metric. Detailed experimental results indicate that only a small

fraction of the actual feature vectors are accessed under the new framework, thus significantly reducing search complexity.

## 4.1 Introduction

Similarity search in content based retrieval is based on searching for nearest neighbors (NN) in a given feature space. For example, to index a collection of images, one first computes various feature descriptors, such as those for color, texture and shape. To search the image collection for images similar in color to a given query image, a nearest neighbor search is performed in the corresponding color feature space and the closest neighbors are then retrieved and presented. This is a typical scenario for similarity search in content based retrieval. Thus, the first step in creating a database is to compute relevant feature descriptors to represent the content. Color, texture, shape and motion descriptors are some of the commonly used low-level visual descriptors for image and video data [78]. While the low level features are quite effective in "similarity" retrieval, there exists a significant gap between these features and the associated visual semantics.

In this context, relevance feedback was introduced to facilitate interactive learning for refining the retrieval results [104, 102]. In relevance feedback, the user identifies a set of retrieval examples relevant to a given query image. The feedback from the user, which often includes identifying a set of positive and negative examples, is then used to compute a new set of retrievals [61, 81, 98, 105, 107, 85]. The user's feedback can be

used in several different ways to improve the retrieval performance. For example, the similarity metric can be modified based on the positive and negative examples. The distance between two feature vectors is typically calculated as:

$$d^2(Q, F, W) = (Q - F)^T W (Q - F), \tag{4.1}$$

where $Q$ is a query vector, $F$ is a database feature vector, and $W$ is a positive semi-definite weight matrix [5]. During each iteration, the weight matrix is updated based on the user's feedback. Using the updated weight matrix, the next set of nearest neighbors is then computed.

Recently, kernel based approaches have emerged as an alternative to relevance feedback using weight matrix updates [147]. A kernel-based system is capable of handling non-linear distributions and "learns" the user's search aims through a sequence of user interactions [146, 119]. However, the computational complexity of learning mechanisms cannot be neglected for large datasets and high dimensions. For this reason, the weight matrix update for relevance feedback is still preferred over kernel-based methods.

While most similarity-based retrievals require nearest neighbor computations in a feature space, it is a well known fact that nearest neighbor computations over a large number of dataset items is expensive [14, 21, 22]. This is further aggravated by the dimensionality curse, as described in Chapter 2. Typical image descriptors are in high dimensions and it is necessary to perform this search repetitively when using relevance feedback. This can limit the overall effectiveness of using relevance feedback for similarity retrieval. Thus, there is a growing awareness that existing relevance

feedback systems cannot process relatively complex queries on large image databases [67].

Recent research has focused on indexing structures to support high-dimensional feature spaces. However, as described in Section 2.5, most of the hierarchical structures are not always reliable and can often be outperformed by a simple linear search over the entire feature space. As an alternative, a sequential search technique over a compressed representation of the database items, termed Vector Approximation File (VA-file) and described briefly in Section 4.2, was introduced in [134]. Using that approach, the feature space is quantized and each feature vector in the database is encoded using its compressed representation. Search complexity is significantly reduces since the computations are carried out in a quantized feature space and only a fraction of the actual feature vectors are accessed.

In the following sections, we present an algorithm for efficient repetitive searching of high-dimensional feature spaces derived from a combination of the VA-file technique and weight matrix update method for relevance feedback. The basic idea is to constrain the search space for the exact nearest neighbor search at iteration $t+1$ using the set of nearest neighbors from the current iteration $t$, and we derive constraint conditions for achieving this. Further, the proposed algorithm takes advantage of the quantization approach to indexing and allows efficient computation of nearest neighbors when the underlying quadratic distance metric is changing. This work expands and generalizes our earlier work on this problem [122, 121, 138] to include a general distance metric

and significantly improved filtering bounds. Detailed experimental results on real image feature datasets show performance improvements in I/O access of up to nearly two full orders of magnitude compared to the standard VA-file approach to nearest neighbor search.

## 4.2 Vector Approximation File (VA-file)

The Vector Approximation File, VA-File, was introduced in [134] as an efficient index structure for nearest neighbors. The key element of VA-File [134] is to first construct a compressed domain representation of the feature space. This compression by quantization enables more items to be loaded into the main memory for fast access. Given a query feature vector, it is possible to efficiently filter out in the quantized space those feature vectors that can not be in the top $K$ nearest neighbor set for that query.

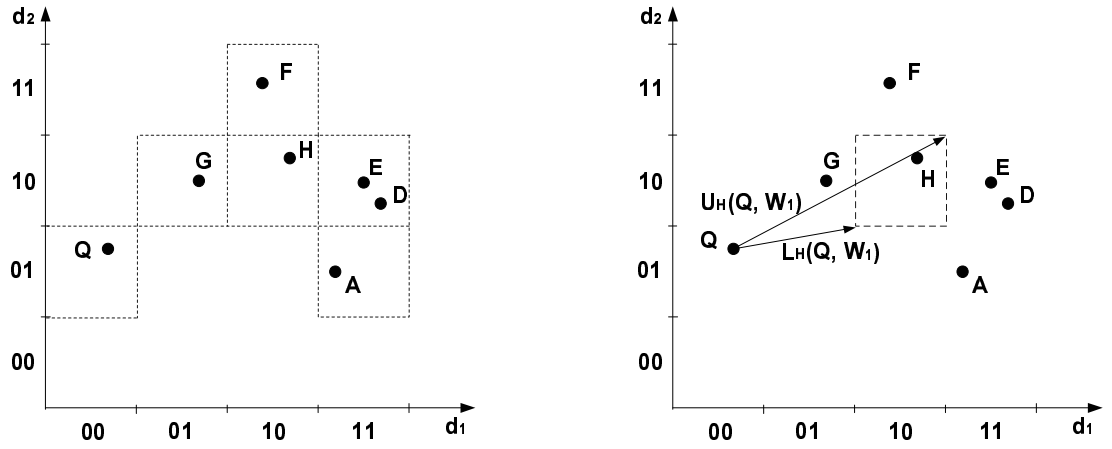
### 4.2.1 Construction of Approximations

Consider a database $\Phi$ of $N$ elements, $\Phi = \{F_i \mid i \in [1, N]\}$, where $F_i$ is an $M$-dimensional feature vector:

$$F_i = [f_{i1}, f_{i2}, \ldots, f_{iM}]^T. \tag{4.2}$$

Let $Q = [q_1, q_2, \ldots, q_M]^T$ be a query object from the database $\Phi$, $Q \in \Phi$. Define the quadratic distance metric $d(Q, F_i, W)$ between query $Q$ and database object $F_i$ as:

$$d(Q, F_i, W_t) = \sqrt{(Q - F_i)^T W_t (Q - F_i)}. \tag{4.3}$$




(a) VA-file Construction

(b) Bound Computation

Figure 4.1: a) Construction of VA-file approximations where $B_1 = B_2 = 2$, and b) computation of upper and lower bounds on $d(Q, H, W_1)$ for Euclidean distance.

$W_t$ is a symmetric, real, positive definite weight matrix.

Each of the feature vector dimensions is partitioned into non overlapping segments, as shown in Figure 4.1(a). Generally, the number of segments is $2^{B_j}$, $j \in [1, M]$, where $B_j$ is the number of bits allocated to dimension $j$. Then, the total number of bits allocated for each high-dimensional cell is $\sum_{j=1}^{M} B_j$. For a feature vector $F_i$, the approximation $C(F_i)$ is an index to the cell containing $F_i$. If $F_i$ is in partition $l$, $l \in [0, 1, 2, \ldots, 2^{B_j} - 1]$, along the $j^{th}$ dimension, the boundary points that determine the $l^{th}$ partition are $b_{lj}$ and $b_{l+1,j}$, and $b_{l,j} \leq f_{i,j} \leq b_{l+1,j}$.

Consider Figure 4.1(a) where $B_1 = B_2 = 2$. The approximation cell $C(G)$ for point $G$ is coded as "0110". "01" and "10" are the indices on dimension $d_1$ and $d_2$, respectively. Note that each approximation cell may contain a number of feature vectors. For example, $C(D)$ and $C(E)$ have the same approximation "1110".

### 4.2.2 Nearest Neighbor (NN) Search

Approximation–based nearest neighbor search can be considered as a two phase filtering process [134]:

**Phase I** - Approximation level filtering: In this phase, the set of all vector approximations is scanned sequentially and lower and upper bounds on the distances of each object in the database to the query object are computed, as shown in Figure 4.1(b). During the scan, a buffer is used to keep track of $\rho$, the $K^{th}$ largest upper bound found from the scanned approximations. If an approximation is encountered such that its lower bound is larger than $\rho$, the corresponding feature vector can be skipped since at least $K$ better candidates exist. Otherwise, the approximation will be selected as a candidate and its upper bound will be used to update the buffer, if necessary. The resulting set of candidate objects at this stage is $N_1(Q, W)$, and the cardinality of the set is $|N_1(Q, W)|$.

**Phase II** - Data level filtering: In this phase, the $K$ nearest neighbors are found. The actual feature vectors, whose approximations belong to a candidate set, $N_1(Q, W)$, are accessed. The feature vectors are visited in increasing order of their lower bounds and the exact distances to the query vector are computed using (4.3). If a lower bound is reached that is larger than the $K^{th}$ actual nearest neighbor distance encountered so far, there is no need to visit the remaining candidates. Let $N_2(Q, W)$ be the set of objects visited before the lower bound threshold is encountered. The $K$ nearest neighbors are found by sorting the $|N_2(Q, W)|$ distances.

Phase II finds the $K$ nearest neighbors from the feature vectors contained by approximations filtered in Phase I. In database searches, the disk/page access is an expensive process. The number of candidates from Phase I filtering determines the cost of disk access/page access. Our focus in this chapter is on improving the Phase I filtering for use with relevance feedback.

## 4.3 Relevance Feedback

Low level descriptors often fail to capture the underlying semantics of the data. To overcome these, researchers have focused on automatic query expansion to help the user re-formulate the query in a feedback session. Relevance feedback has long been suggested as a solution for query modification in word and document retrieval. For example, Rocchio [104] introduced a classification algorithm that uses vectors of numeric weights to represent the data, i.e. a vector space model. The probabilistic model proposed by Robertson and Sparck–Jones [102] modifies these individual weights based on the distribution of the terms in relevant and non-relevant document sets.

In the context of content-based image retrieval, relevance feedback has attracted considerable attention. In a typical scenario, given a set of retrievals for an image query, the user may identify some relevant and some non-relevant examples. Based on this the similarity metric is modified to recompute the next set of retrievals. The hope is that this modification to the similarity metric will help provide better matches to a given query and meet the user's expectations.

Kernel–based approaches [147] tend to learn the user's concepts better through a series of feedback sessions (20-30), and on smaller sample sets. However, for large datasets and smaller numbers of sessions, kernel-based concepts fail to provide fast response time and significantly better retrievals, leaving only similarity metric update as an acceptable and scalable solution for real–time content-based image retrieval systems.

At iteration $t$, let $W_t$ be the weight matrix used, and $R_t$ be the set of $K$ nearest neighbors to the query object $Q$, using (4.3) to compute the distances. Also for iteration $t$, define the $k^{th}$ positive example vector ($k = \{1, \ldots, K'\}$) as

$$X_k^{(t)} = [x_{k1}^{(t)}, x_{k2}^{(t)}, \ldots, x_{kM}^{(t)}]^T, \tag{4.4}$$

where $X_k^{(t)} \in R_t$ and $K'$ is the number of relevant objects identified by the user. These $K'$ examples are used to update the weight matrix $W_t$ to $W_{t+1}$. We consider an optimized learning technique that merges two existing well –known updating schemes:

**MARS** [105] restricts $W_t$ to be a diagonal matrix. The weight matrix is updated using the standard deviation $\sigma_m$ of $x_{km}^{(t)}$ ($k = \{1, \ldots, K'\}$, $m = \{1, \ldots, M\}$). The weight matrix is normalized after every update and the result is given by:

$$(W_{t+1})_m = \frac{(\prod_{i=1}^{M} \sigma_i^2)^{\frac{1}{M}}}{\sigma_m^2}. \tag{4.5}$$

**MindReader** [61] updates the full weight distance matrix $W_t$, by minimizing the distances between the query and all positive feedback examples. In this scheme, the user picks $K'$ positive examples and assigns a degree of relevance $\pi_k^{(t)}$ to the $k^{th}$ positive example $X_k^{(t)}$. The optimal solution for $W_t$ is equivalent to the Mahalanobis distance

assuming the positive examples are Gaussian:

$$W_{t+1} = det(C_t)^{\frac{1}{M}}(C_t)^{-1}. \tag{4.6}$$

The elements of the covariance matrix $C_t$ are defined as:

$$(C_t)_{ij} = \frac{\sum_{k=1}^{K'} \pi_k^{(t)}(x_{ki}^{(t)} - q_i)(x_{kj}^{(t)} - q_j)}{\sum_{k=1}^{K'} \pi_k^{(t)}}. \tag{4.7}$$

For $K' > M$, matrices $C_t$ and $W_t$ are symmetric, real and positive definite. $C_t$ can be factorized as:

$$C_t = (P_t')^T \Lambda_t' P_t', \quad {P_t'}^T P_t' = I, \quad \Lambda_t' = diag(\lambda_1', ..., \lambda_M') \tag{4.8}$$

and $W_t$ can be factorized in the same manner:

$$W_t = P_t^T \quad \Lambda_t P_t, \quad P_t^T P_t = I, \Lambda_t = diag(\lambda_1, ..., \lambda_M). \tag{4.9}$$

Note that (see (4.6)) $P_t = P_t'$ and:

$$\lambda_i = \frac{(\prod_{i=1}^{M} \lambda_i')^{\frac{1}{M}}}{\lambda_i'}. \tag{4.10}$$

Mars and MindReader formulate the weight matrix query point update as an optimization problem: Minimize the sum of distances subject to the constraint that $||W|| = 1$. Both updates are subjected to the constraint that the sum of distances between query vector and positive examples is minimized when going from $W_t$ to $W_{t+1}$. Results reported in [61] and [106] show that the distance matrix converges to its optimum value after only a couple of iterations.

The full matrix update of MindReader approach captures quite well the dependencies among feature dimensions, thus reducing redundancies in high-dimensional feature

space and allowing more false candidates to be filtered out. The downside of the full matrix update approach is that the inverse covariance matrix $(C_t)^{-1}$ exists only if the number of positive examples, $K'$, is equal to or larger than the number of feature dimensions, $M$. In case $K' \leq M$, MARS approach is used.

Note that the number of candidates taken into account for relevance feedback does not necessarily have to come from the user. In a large interactive system, the user does not see all the millions of images. The system lets the user pick representatives and refine the query throughout the iterations. Based on what the user picks, the system can re-use items from the previous iteration, including same class representatives (in a multimedia framework environment) or the items from the same cluster (if cluster information is available) for the new updates.

With this brief introduction to relevance feedback, we can now formulate the nearest neighbor search problem as follows: *Given $R_t$, $W_t$, and $K'$, the weight matrix $W_{t+1}$ is derived from $W_t$ using some update scheme, compute the next set of $K$ nearest neighbors $R_{t+1}$ using $W_{t+1}$ and using a minimum number of computations.*

## 4.4 Bound Computation

The nearest neighbor (NN) filtering process in the Vector Approximation approach of Phase I, as described in 4.2 uses information on the lower and upper bounds of the distance between a query point $Q$ and a feature vector $F_i$. Given a query $Q$ and a feature vector $F_i$, the lower and upper bounds on the distance $d(Q, F_i, W^t)$ are defined

as $L_i(Q, W_t)$ and $U_i(Q, W_t)$ such that the following inequality holds:

$$L_i(Q, W_t) \leq d(Q, F_i, W_t) \leq U_i(Q, W_t). \tag{4.11}$$

### 4.4.1 Weighted Euclidean Distance

The computation of the lower and upper bounds on distance $d(Q, F_i, W_t)$ for the VA-file index is straightforward for a diagonal $W_t$. Bounds are constructed based on a hyper rectangular approximation. If $W_t$ is a diagonal matrix with non-negative entries, i.e.. $W_t = \Lambda_t = diag(\lambda_1, ..., \lambda_M)$, then:

$$L_i^2(Q, W_t) = [l_{i1}, l_{i2}, \ldots, l_{iM}]^T \Lambda_t [l_{i1}, l_{i2} \ldots, l_{iM}] \tag{4.12}$$

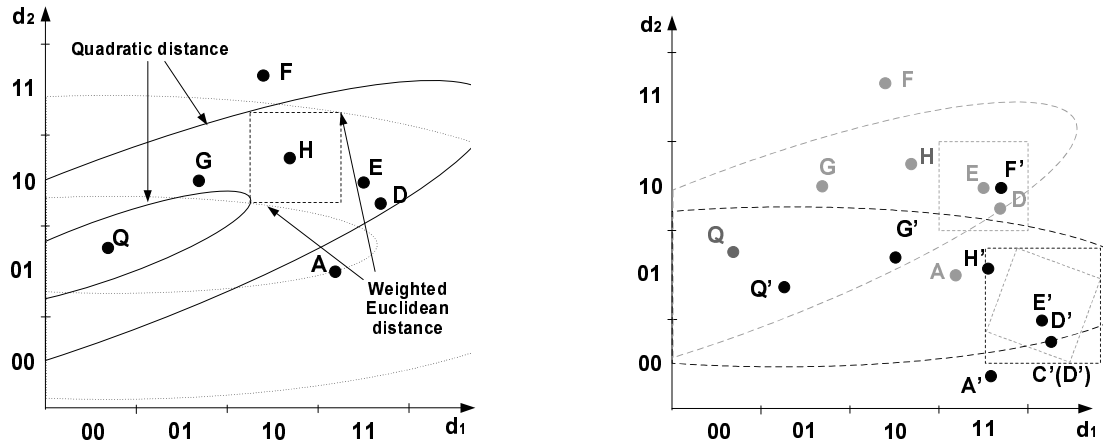
$$U_i^2(Q, W_t) = [u_{i1}, u_{i2} \ldots, u_{iM}]^T \Lambda_t [u_{i1}, u_{i2} \ldots, u_{iM}],$$

where:

$$l_{ij} = \left\{ \begin{array}{ll} q_j - b_{l+1,j} & q_j > b_{l+1,j} \\ 0 & q_j \in [b_{l,j}, b_{l+1,j}] \\ b_{l,j} - q_j & q_j < b_{l,j} \end{array} \right\}, \tag{4.13}$$

and

$$u_{ij} = \left\{ \begin{array}{ll} q_j - b_{l,j} & q_j > b_{l+1,j} \\ max(q_j - b_{l,j},\ b_{l+1,j} - q_j) & q_j \in [b_{l,j}, b_{l+1,j}] \\ b_{l+1,j} - q_j & q_j < b_{l,j} \end{array} \right\}. \tag{4.14}$$




(a) Bound Computation

(b) Rotational Mapping

Figure 4.2: a) Bound computation for (1) weighted Euclidean and (2) quadratic distance, and b) Rotational mapping of feature space and approximation cells:

$$D \rightarrow D' \ : \ D' = PD.$$

### 4.4.2 General Quadratic Distance

For the case of a general quadratic distance metric, nearest neighbor query becomes an ellipsoid query. Points $F_i$ that have the same distance $d(Q, F_i, W_t)$ from a query point $Q$ form an ellipsoid centered around query point $Q$. Lower and upper bound computations in the cases of weighted Euclidean distance and quadratic distance are illustrated in Figure 4.2(a).

It is computationally expensive to determine whether a general ellipsoid intersects a cell in the original feature space. For the quadratic metric, exact distance computation between the query object $Q$ and a rectangle $C(F_i)$ requires a numerically extensive quadratic programming approach, which would undermine the advantages of using any vector approximation indexing structure. Our proposed solution to this problem is to

approximate the upper and lower bound on a cell based on approximation techniques for ellipsoid queries [5]. Assuming that distance matrix $W_t$ is real, symmetric, and positive definite, we can applied the factorized $W_t$ from (4.9) to get the distance as

$$d^2(Q, F_i, W_t) = (P_t(Q - F_i))^T \Lambda_t (P_t(Q - F_i)). \tag{4.15}$$

Define a rotational mapping for matrix $P_t$ such that $Q \rightarrow Q' : \ Q' = P_t Q$. All quadratic distances in the original space transform to weighted Euclidean distances in the mapped space, i.e.

$$d^2(Q, F_i, W_t) = (Q' - F_i')^T \Lambda_t (Q' - F_i'). \tag{4.16}$$

Cell $C(D)$ that approximates feature point $D$ is rotated into a hyper parallelogram $C(D')$ in the mapped space, as illustrated in Figure 4.2(b). The parallelogram $C(D')$ can be approximated with bounding hyper rectangular cell $C'(D')$. The weight matrix in the mapped space is $\Lambda_t$, and the quadratic distance becomes a weighted Euclidean distance. $L_i(Q, W_t)$ and $U_i(Q, W_t)$ are approximated in the mapped space with $L_i(Q', \Lambda_t)$ and $U_i(Q', \Lambda_t)$. $L_i(Q', \Lambda_t)$ and $U_i(Q', \Lambda_t)$ are determined as in Section 4.4.1.

Conservative bounds on rectangular approximations introduced in [5, 108] allow us to avoid computing the exact distance between query object $Q$ and every approximation cell $C(F_i)$. However, for restrictive bounds, the distance computation stays quadratic with the number of feature dimensions in (4.2), while the approximation $C(F_i)$ need only specify the bounding rectangles position in the mapped space. Note that the size of approximated bounding rectangle depends only on the cell size in the original

space, and the rotation matrix $P_t$, which is computed prior to a new search. The computational complexity of the distance metric is further reduced since the weighted Euclidean distance has the same computational complexity as the Euclidean distance.

## 4.5 Adaptive Nearest Neighbor Search for Relevance Feedback

In database searches, the disk/page access is an expensive process, directly proportional to the number of approximations determined in Phase I filtering, as described in Section 4.2. Phase I filtering determines a subset of approximations, $N_1(Q, W_t)$, from which the $K$ nearest neighbors can be retrieved,. Let $N_1^{opt}(Q, W_t)$ be the minimal set of approximations that contain $K$ nearest neighbors. The best case result from Phase I filtering is to exactly identify this subset $N_1(Q, W_t) = N_1^{opt}(Q, W_t)$. However, Phase I filtering generally introduces some false candidates. The number of false candidates depends on how firm the filtering bounds are throughout the sequential scan. Also, the approximation lower bound can be much smaller than the real lower bound and can thus introduce many false candidates. The smaller the candidate set is, the better the indexing and search performances. Our objective is to improve Phase I filtering in the context of relevance feedback. Let $\rho$ be the $K^{th}$ largest upper bound encountered so far during a sequential scan of approximations. In the standard approach of Section 4.2, the approximation $C(F_i)$ is included in $N_1(Q, W_t)$ only if $L_i(Q, W_t) < \rho$, and the

value of $\rho$ is updated if $U_i(Q, W_t) < \rho$. Thus, only the $K^{th}$ largest upper bound from the scanned approximations is available and used for filtering.

In investigating Phase I filtering in the context of relevance feedback, we begin by exploring the relationship between $R_t$ and $R_{t+1}$, i.e., the $K$ nearest neighbors in two consecutive iterations. At iteration $t-1$ , $R_{t-1}$ contains the $K$ nearest neighbors of query $Q$ computed using the weight matrix $W_{t-1}$. Our objective is to compute the next set of $K$ nearest neighbors $R_t$ for weight matrix $W_t$.

Let $R_t = \{F_k^{(t)}\}$ be the set of $K$ nearest neighbors of query $Q$ at iteration $t$ with weight matrix $W_t$:

$$R_t = \{F_k^{(t)} |\ \forall i \neq k, d(Q, F_k^{(t)}, W_t) < d(Q, F_i^{(t)}, W_t), k \in [1, K]\}. \tag{4.17}$$

At iteration $t$, define $r_t^u(Q)$ as:

$$r_t^u(Q) = max_k\{d(Q, F_k^{(t-1)}, W_t)\}, \quad F_k^{(t-1)} \in R_{t-1}. \tag{4.18}$$

Define $r_t(Q)$ as the maximum distance between $Q$ and the items in $R_t$:

$$r_t(Q) = max_k\{d(Q, F_k^{(t)}, W_t)\}, \quad F_k^{(t)} \in R_t. \tag{4.19}$$

**Lemma 1** When $W_{t-1}$ is updated to $W_t$, then the upper bound on $r_t(Q)$ is given by:

$$r_t(Q) \leq r_t^u(Q). \tag{4.20}$$

**Proof** Equation (4.18) states that there are $K$ distance values $d(Q, F_k^{(t-1)}, W_t)$ in the subset $R_{t-1}$ of the whole database $\Phi$ that are smaller or equal to $r_t^u(Q)$. From (4.19),

it follows that there are $K$ distance values $d(Q, F_k^{(t)}, W_t)$ over the whole database $\Phi$ smaller or equal to the value of $r_t(Q)$. If $R_{t-1} = R_t$, equality in 4.20 holds. If $R_{t-1} \neq R_t$ , there exists at least one feature vector in $R_{t-1}$ that does not belong to a $K$-NN set of $R_t$ i.e. $r_t(Q) < r_t^u(Q)$.

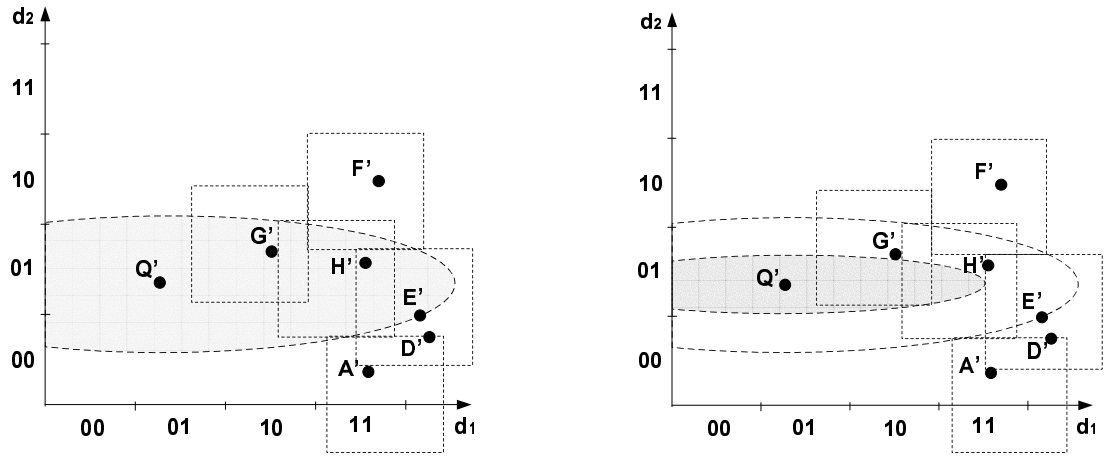
In summary, the maximum of the distances between the query $Q$ and objects in $R_t$ computed using $W_t$ can not be larger than the maximum distance between the query $Q$ and the objects in $R_{t-1}$ computed using $W_t$.

**Corollary 1** For approximation $C(F_i)$ to be a qualified one in $N_1^{opt}(Q, W_t)$, its lower bound $L_i(Q, W_t)$ must satisfy:

$$L_i(Q, W_t) < r_t^u(Q). \tag{4.21}$$

Let $R_1 = \{E, H\}$ be a user's answer set for a query $Q$ in a the feature space illustrated in Figure 4.3(a). When $W_1$ is updated to $W_2$, $r_2^u(Q) = d(Q, E, W_2)$. The Lemma states that the maximum of the distances between the query $Q$ and objects in $R_2$ computed using $W_2$, $r_2)(Q)$ can not be larger than $r_2^u(Q)$. The answer set offered to a user will be limited to points inside radius $r_2^u(Q) = d(Q, E, W_2)$, marked as the shaded area in Figure 4.3(a).

As we discussed above, false candidates can be introduced if Phase I filtering bound is larger than the actual $K^{th}$ smallest lower bound over $\Phi$. This is also important when approximating cell bounds using quadratic distance metric with spatial transformations. A spatial transformation, as demonstrated in Figure 4.2(b), can introduce false




(a) $r_t^u$ filter bound

(b) $l_t^u$ filter bound

Figure 4.3: Adaptive search space: (a) Illustration of using $r_t^u$ to limit the search space in Phase I adaptive filtering, and (b) Illustration of using $l_t^u$ to limit the search space in Phase I adaptive filtering.

candidates in Phase I filtering, since the approximation rectangle in the mapped space is larger. Therefore, we further reduce the search space by establishing an upper bound on the $K^{th}$ largest minimum lower bound encountered during the database scan. Define $l_t^u(Q)$ as:

$$l_t^u(Q) = max\{L_k^{(t-1)}(Q, W_t)\}, \quad F_k^{(t-1)} \in R_{t-1}. \tag{4.22}$$

Define $l_t(Q)$ as the maximum lower bound away from $Q$ of the items in $R_t$:

$$l_t(Q) = max\{L_k(Q, W_t)^{(t)}\}, \quad F_k^{(t)} \in R_t. \tag{4.23}$$

Define $\epsilon$ as the diagonal of an index hypercube using $S$ bits per dimension and $M$ dimensions, thus

$$\epsilon = \frac{\sqrt{M}}{2^S}. \tag{4.24}$$

**Lemma 2** When $W_{t-1}$ is updated to $W_t$, then the upper bound on $l_t(Q)$ is given by

$$l_t(Q) \leq l_t^u(Q) + \epsilon. \tag{4.25}$$

**Proof** Equation (4.22) states that there are $K$ lower bounds $L_k(Q, W_t)$ over the subset $R_{t-1}$ of the database $\Phi$ that are smaller or equal to $l_t^u$, $k \in [1, K]$. From (4.23), it follows that there are $K$ lower bounds $L_k(Q, W_t)$ over the database $\Phi$ smaller or equal to the value of $r_t$. If $R_{t-1} = R_t$, then equality in 4.25 holds. If $R_{t-1} \neq R_t$, there exists at least one feature vector in $R_{t-1}$ that does not belong to a $K$-NN set of $R_t$, i.e., its lower bound is larger than $l_t^u(Q)$ and $l_t(Q) < l_t^u(Q) + \epsilon$.

This lemma states that the filtering bound on maximum lower bound of the approximations encountered during the scan cannot be larger than the maximum lower bound computed using $W_t$ for the query $Q$ and the objects in $R_{t-1}$ .

**Corollary 2** For approximation $C(F_i)$ to be a qualified one in $N_1^{opt}(Q, W_t)$, its lower bound $L_i(Q, W_t)$ must satisfy:

$$L_i(Q, W_t) < l_t^u(Q) + \epsilon. \tag{4.26}$$

Based on the approximate nearest neighbor definition in [6], given a query point $Q$ and an positive error bound $\epsilon$, $\epsilon$-NN is a neighbor of the query point within a factor of $(1 + \epsilon)$ of the distance to the true nearest neighbor of $Q$. Lets $N_1^{app}(Q, W_t)$ be the $\epsilon$-NN set of query point $Q$, where $\epsilon$ is defined in 4.24.

**Corollary 3** For approximation $C(F_i)$ to be a qualified one in $N_1^{app}(Q, W_t)$, its lower bound $L_i(Q, W_t)$ must satisfy:

$$L_i(Q, W_t) < l_t^u(Q). \tag{4.27}$$

Note that, $r_t^u(Q) \leq l_t^u(Q) + \epsilon$. However if we use only $l_t^u(Q)$ as a Phase I filtering bound for the lower bound on distance between the query vector and database objects, the upper bound computation for each approximation encountered is entirely avoided. There are possible scenarios where $l_t^u(Q)$ bound can filter out some qualified candidates. Experiments show that, for high dimension, the accuracy of adaptive filtering that uses $l_t^u(Q)$ is the same as the accuracy of the exact nearest neighbor search, see4.6. In other words, the use of $l_t^u(Q)$ statistically guarantees that the exact $K$ nearest neighbors are retrieved, since the probability of the miss is very small in high dimensions [59].

A benefit of pre-computed bounds is their property of selection, as illustrated in Figure 4.3 where $R_{t-1} = \{E, H\}$. The user identified points $E$ and $H$ as the ones similar to the query point $Q$. Using the feedback, the weight matrix is updated from $W_1$ to $W_2$. Under the new distance metric, $r_t^u = d(Q, E, W_t)$ and the search space is restricted to the shaded area in Figure 4.3(a). Also, $l_t^u = L_E(Q, W_t)$ and the search space is further restricted, as shown in Figure 4.3(b), where shaded area marks the search space for $L_i(Q, W_t)$ limited by the ellipsoid defined by $l_u^t = L_E(Q, W_t)$.

### 4.5.1 An adaptive K-NN search algorithm

We will now outline a new $K$-NN search method that improves upon the VA-file index for repetitive searches. For $t = 1$, the $K\text{-}NN$ search Phase I filtering reduces to standard Phase I filtering [134]. Note that in the standard approach a buffer is used to keep track of the value of $\rho$, the $K^{th}$ largest upper bound found so far during a scan. In practice, this buffer is used only when user's feedback or accumulated query history is not available.

In the presence of relevance feedback, the buffer update is avoided and the new filtering bound is defined by $r_t^u(Q)$ or $l_t^u(Q)$. The data filtering (Phase II) step results in a set of $K$ nearest neighbors, $R_t$. Then, a new iteration is started with $t = t + 1$. The user identifies positive examples in $R_{t-1}$ and the information is used to update $W_{t-1}$ to $W_t$. Note that the $W_t$ update ($P_t$ and $\Lambda_t$ are also computed in the process) and $r_t^u(Q)/l_t^u(Q)$ computation are done before starting the next phase of the Phase I filtering. The proposed changes to the Phase I filtering are shown in Algorithm 1 when $l_t^u(Q)$ is used as a filtering bound.

### 4.5.2 Advantages of the Proposed Method

In the standard approach the umber of false candidates resulting from Phase I filtering depend on the convergence rate of $\rho$ to its final value. Using firmer filtering bounds than in the standard approach reduces the number of false candidates collected during the sequential scan of the approximations [122] and increases search

**Algorithm 1** Adaptive $K$-$NN$ search

$\boldsymbol{t = 1}$:

Standard VA-file Phase I filtering (Section 4.2);

**while** $t < T$ **do**

$t = t + 1$;

$W_t \rightarrow W_{t+1}$; (Section 4.3)

**if** $W_t = W_{t+1}$ **then**

Return $R_{t-1}$;

BREAK;

**end if**

$l_t^u(Q) = max\{L_{i^{t-1}}(Q, W_t)\}, F_i^{t-1} \in R_{t-1}$;

**Adaptive Phase I**:

**for all** $i$ **do**

Compute $L_i(Q, W_t)$; (Section 4.4.2)

**if** $L_i(Q, W_t) \leq l_t^u(Q)$ **then**

Insert $C(F_i)$ into $N_1^{(l)}(Q, W_t)$;

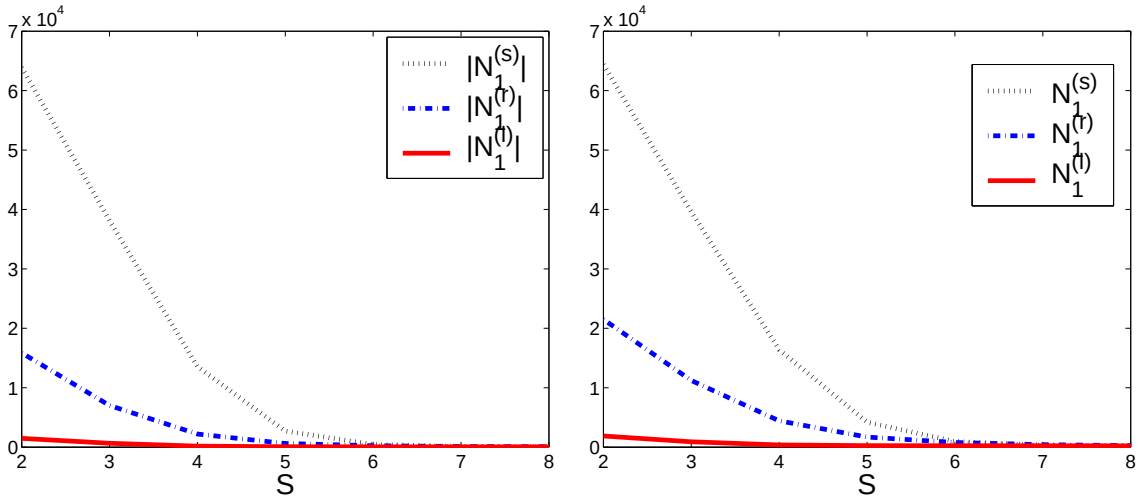
**end if**

**end for**

Return: $N_1(Q, W_t)$;

**Phase II:** Return $R_t$

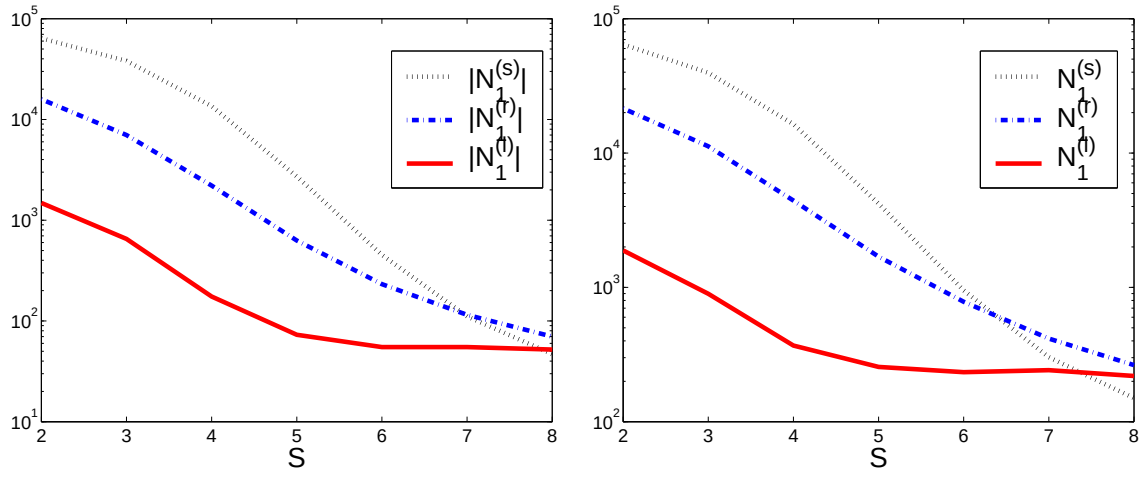
**end while**




(a) $M = 30$ (b) $M = 60$

Figure 4.4: Average number of cells selected in Phase I (4.29) from the whole database of 90774 vectors for weighted Euclidean distance, and $K = 20$ nearest neighbor search: $N_1^{(s)}$ using standard approach, $N_1^{(r)}$ using adaptive approach with $r_t^u$ as the bound, and $N_1^{(l)}$ using adaptive approach with $l_t^u$ as the bound, see (4.29).

efficiency. Note that both $r_t^u(Q)$ and $l_t^u(Q)$ are computed before Phase I filtering. Since $l_t^u(Q, W_t) \leq r_t^u(Q, W_t)$, fewer candidates need to be examined in Phase I filtering when $l_t^u(Q)$ is used as a filtering bound. Also, keeping the standard $\rho$ updated requires an upper bound computation for every candidate [134], and the expensive calculation of upper bounds and the $K^{th}$ smallest upper bound can be avoided using the proposed approach.




(a) $M = 30$ (b) $M = 60$

Figure 4.5: Logarithmic (base 10) scale of average percentage of cells selected in Phase I (4.29) from the whole database for weighted Euclidean distance, and $K = 20$ nearest neighbor search: $N_1^{(s)}$ using standard approach, $N_1^r$ using adaptive approach with $r_t^u$ as a bound, and $N_1^l$ using adaptive approach with $l_t^u$ as a bound, see (4.29).

## 4.6 Experiments

We compare the standard VA-file approach for computing the $K$ nearest neighbors to our proposed adaptive method for different bit resolutions $S$ and the various filtering bounds described in Section 4.5. We demonstrate the efficiency of the proposed approach on a dataset of $N$ texture feature vectors. These vectors are computed for $N = 90774$ region tiles of an aerial image dataset (image data size is 1.4Gb). 60-dimensional feature vectors are formed using the first- and second-order moments of the Gabor filter outputs, see (3.5), and 30-dimensional feature vectors are formed from the Rayleigh coefficients of the Gabor filter outputs, see (3.14).

The approximations are constructed using the standard VA–file index, as described in Section 4.2.1. Experiments are carried out for different numbers of bits, $S \in [2, 3, 4, 5, 6, 7, 8]$ assigned to every dimension, where the same number of bits $S$ is assigned to each of the $M$ uniformly partitioned feature dimensions. A larger value of $S$ corresponds to constructing the approximation at a finer resolution. For each query, $K$ nearest neighbors are retrieved during each iteration. The user's feedback is based on texture relevance only. For a specific query, the user selects $K'$ relevant nearest neighbors to update the distance metric before every iteration (Section 4.3).

Queries $Q_i$ are selected from the dataset to cover both dense cluster representatives and outliers in the feature space. For a given resolution $S$ and query vector $Q_i$, let the number of candidates from the Phase I standard filtering approach be $|N_1^{(s)}(Q_i)|$ for the standard approach. Let the corresponding values for the proposed adaptive methods be $|N_1^{(r)}(Q_i)|$ with filtering bound $r_t^u(Q)$, and $|N_1^{(l)}(Q_i)|$ with filtering bound $l_t^u(Q)$.

Define the average Phase I selectivity bound as

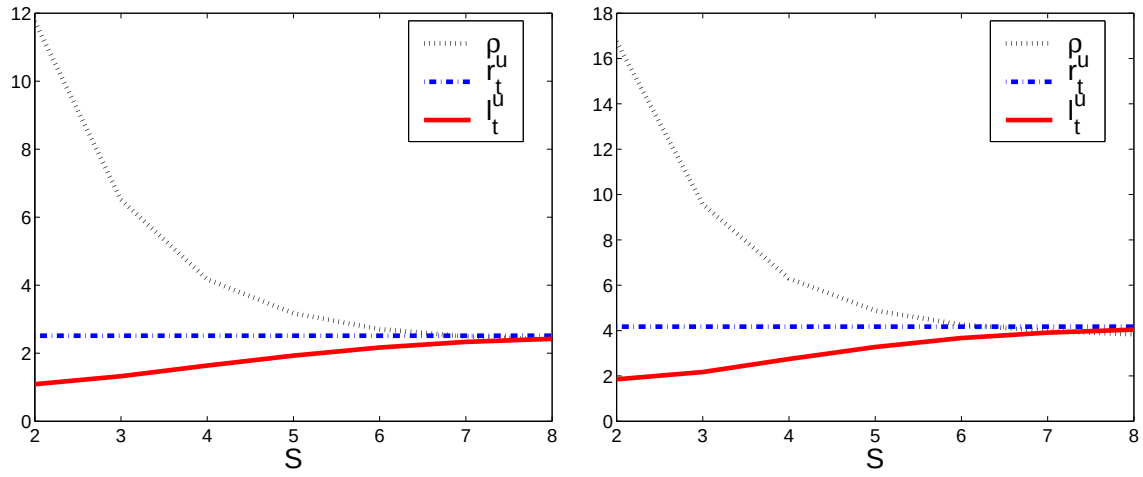
$$\rho = \frac{1}{I}\sum_{i=1}^{I}\rho(Q_i) \quad r_t^{(u)} = \frac{1}{I}\sum_{i=1}^{I} r_t^{(u)}(Q_i), \quad l_t^{(u)} = \frac{1}{I}\sum_{i=1}^{I} l_t^{(u)}(Q_i), \tag{4.28}$$

and the number of Phase I candidates over the example queries as

$$N_1^{(s)} = \frac{1}{I}\sum_{i=1}^{I}|N_1^{(s)}(Q_i)|, \quad N_1^{(r)} = \frac{1}{I}\sum_{i=1}^{I}|N_1^{(r)}(Q_i)|, \quad N_1^{(l)} = \frac{1}{I}\sum_{i=1}^{I}|N_1^{(l)}(Q_i)|, \tag{4.29}$$

and the corresponding effectiveness measures,

$$\alpha^{(r)} = \frac{1}{I}\sum_{i=1}^{I}\frac{|N_1^{(s)}(Q_i)|}{|N_1^{(r)}(Q_i)|} \quad \alpha^{(l)} = \frac{1}{I}\sum_{i=1}^{I}\frac{|N_1^{(s)}(Q_i)|}{|N_1^{(l)}(Q_i)|} \quad \gamma = \frac{1}{I}\sum_{i=1}^{I}\frac{|N_1^{(r)}(Q_i)|}{|N_1^{(l)}(Q_i)|}. \tag{4.30}$$



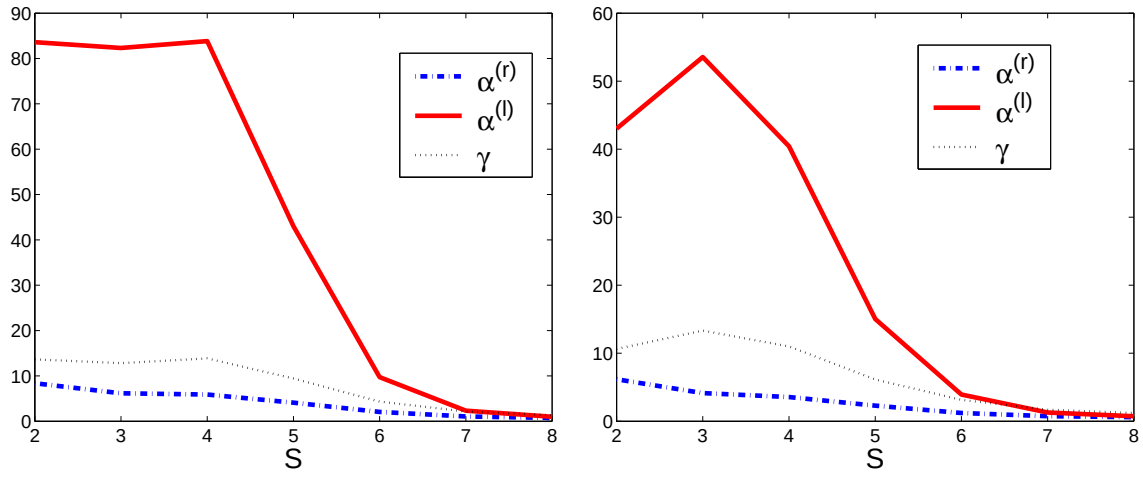

(a) $M = 30$ (b) $M = 60$

Figure 4.6: Phase I selectivity bound distances for weighted Euclidean distance and $K = 20$ nearest neighbors: $\rho$ for standard filtering, and $r_t^u$ and $l_t^u$ for the adaptive approach.

### 4.6.1 Weighted Euclidean Metric

For the Weighted Euclidean distance, we average over $I = 20$ query vectors for $K = 20$ nearest neighbors and $K' = 15$ relevant features for the MARS weight matrix update (4.5). The number of candidates resulting from the standard and adaptive Phase I filtering (with different filtering bounds) is shown in Figure 4.4. Figure 4.5 shows that $l_t^u(Q)$ restricts the search space more than $r_t^u(Q)$ even for finer resolutions. When $S = 8$, $\rho$ converges to a value that is smaller than $r_t^u(Q)$, but very close to $l_t^u(Q)$. The filter bounds over all resolutions are shown in Figure 4.6.

In Figure 4.7, the average gain of the proposed method as reported by the effectiveness measures of (4.30) is not monotone over $S$, since the results are strongly correlated



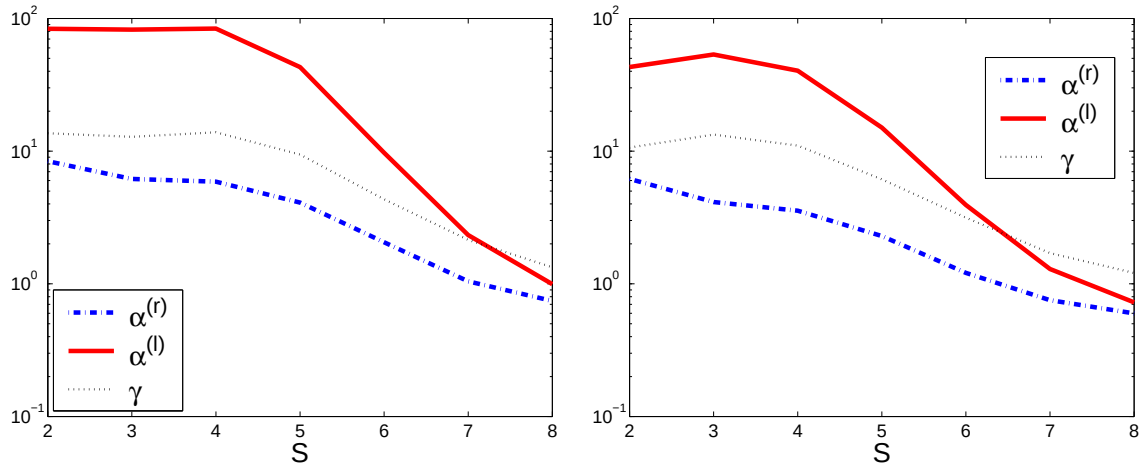

(a) $M = 30$ (b) $M = 60$

Figure 4.7: Adaptive gain (4.30) for weighted Euclidean distance: $\alpha^{(r)}$ for adaptive Phase I search using $r_t^u$, $\alpha^{(l)}$ for adaptive Phase I search using $l_t^u$, and $\gamma$ for standard VA-file search.

with the density of the region around the query image. However, the average minimum gain of the proposed adaptive filtering is still significant at every resolution $S$. At coarser resolution, the gain of the firmer filtering bound $\gamma$ is over 10. Note that $\alpha^{(l)}$ is around 10 times larger than $\alpha^{(r)}$ for $S = 2, 3, 4$. At finer resolutions, $l_t^u(Q)$ and $\rho$ are comparable filtering bounds as shown by the gain $\alpha^{(l)}$ being close to 1 in Figure 4.8.

### 4.6.2 Quadratic Metric

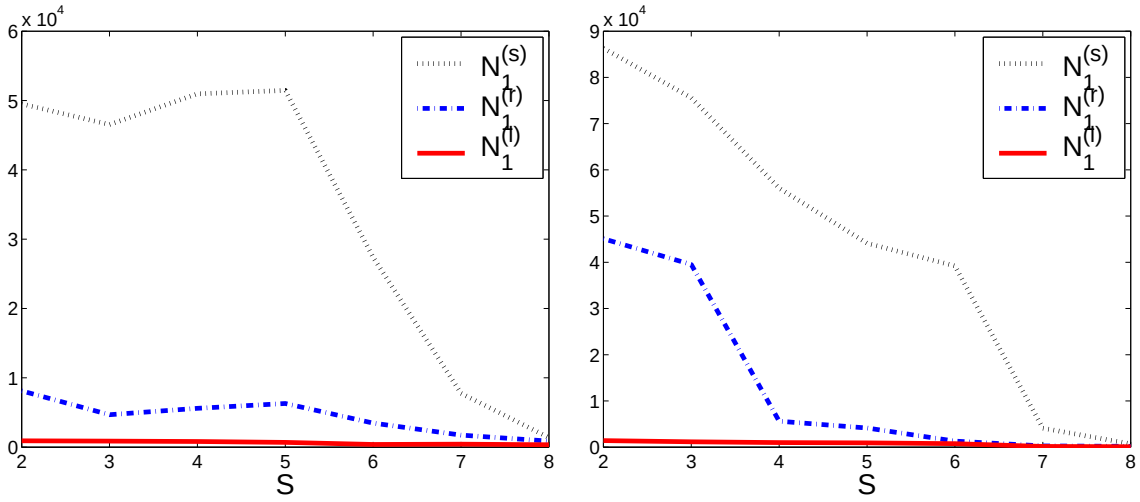
For the quadratic distance, we average over $I = 20$ query vectors, for $K = M + 10$ nearest neighbors and $K' = M + 5$ relevant features using the MindReader weight matrix update of (4.6). Since $l_t^u(Q)$ is the $K^{th}$ smallest approximate upper bound




(a) $M = 30$ (b) $M = 60$

Figure 4.8: Logarithmic (base 10) scale adaptive gain (4.30) for weighted Euclidean distance: $\alpha^{(r)}$ for adaptive Phase I search using $r_t^u$, $\alpha^{(l)}$ for adaptive Phase I search using $l_t^u$, and $\gamma$ for standard VA-file search.

under the new weight matrix, $l_t^u(Q)$ restricts the search space significantly more than $r_t^u(Q)$, as shown in Figure 4.11, and it further reduces the number of candidates as shown in Figure 4.9. In Figure 4.10 the standard approach candidate set for $S = 2$ includes all approximations. $l_t^u$ restricts the search significantly more than $r_t^u$ using the quadratic distance metric and at all resolutions. When $S = 8$, $\rho$ converges to a value that is close to $r_t^u(Q)$, but still larger than $l_t^u(Q)$. These results show that the achieved efficiency gain is significant as demonstrated in Figure 4.12. For finer approximations $S = 8$ where $l_t^u$ proves to be a better estimate of the upper bound on approximations than $\rho$, as the gain $\alpha^{(l)}$ is larger than 1 for $S = 8$, as shown in Figure 4.12.



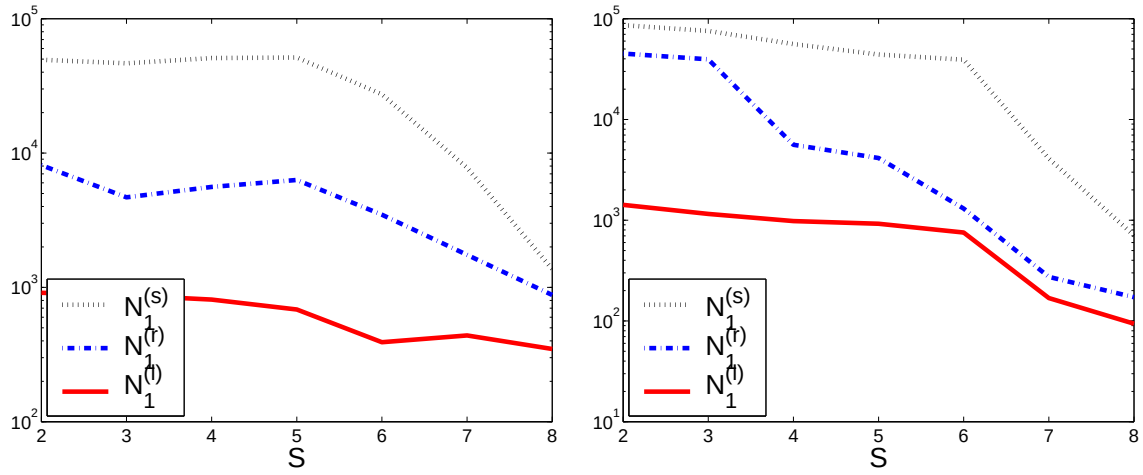

(a) $K = 40$, $K' = 35$, $M = 30$

(b) $K = 70$, $K' = 65$, $M = 60$

Figure 4.9: Average number of cells selected in Phase I for quadratic Euclidean distance: $N_1^{(s)}$ for standard approach, $N_1^r$ and $N_1^l$ for adaptive approach with, respectively, $r_t^u$ and $l_t^u$ as a filtering bounds, see (4.29).

## 4.7 Discussion

In the standard VA-file approach, the number of false candidates is very high for coarse approximations. The large size of the hyper rectangles causes the filter bound $\rho$ to increase significantly. Therefore, a tighter filtering bound such as $l_t^u(Q)$ results in a significant improvement in false candidate filtering. The parameter $l_t^u(Q)$ plays a more significant role for coarser approximations. Figures 4.6 and 4.11 show that $l_t^u(Q)$ is smaller with respect to $r_t^u(Q)$ and $\rho$ for lower values of $S$. For the quadratic distance metric, the approximation bound computation in high-dimensional space introduces additional false candidates during approximation level search. However, $l_t^u$ is significantly smaller than $r_t^u$ for $S = 2, 3, 4$, and allows a smaller number of false candidates

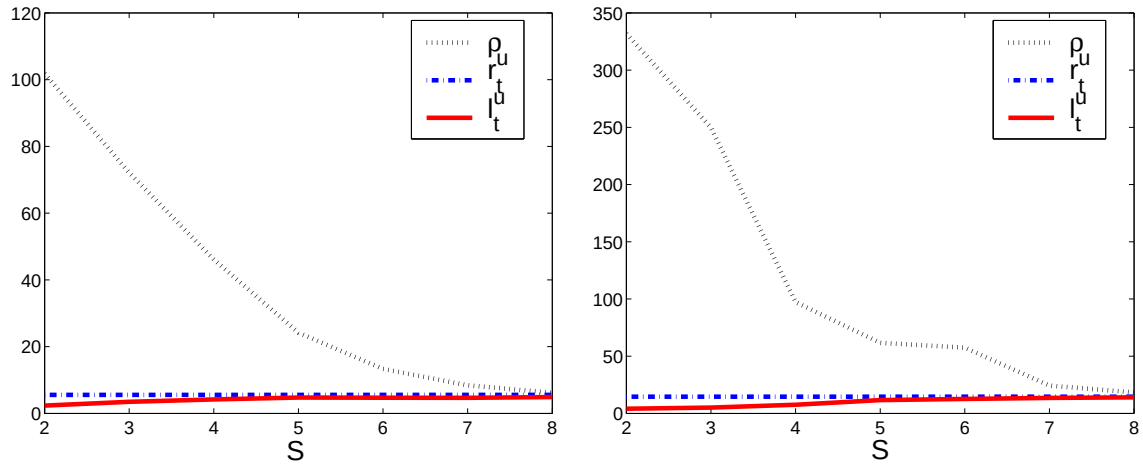



(a) $K = 40$, $K' = 35$, $M = 30$ (b) $K = 70$, $K' = 65$, $M = 60$

Figure 4.10: Logarithmic (base 10) scale average percentage of cells selected in Phase I for quadratic Euclidean distance: $N_1^{(s)}$ for standard approach, $N_1^r$ and $N_1^l$ for adaptive approach with $r_t^u$ and $l_t^u$ as a filtering bounds. is used for the y-axis.

in Phase I filtering process. Besides the irrelevant data points that are pruned by using $l_t^u$, more vectors are pruned than by $r_t^u$ at the expense of allowing some false dismissals. Note that there were no false dismissals in this experiments. We contribute that to large number of nearest neighbors retrieved. Therefore, in the presence of relevance feedback and using $l_t^u(Q)$ as a filtering bound we can either save memory for approximation storage or reduce the number of required disc accesses for the same resolution. Experimental results lead us to conclude that $l_t^u$ is a good estimate of $\rho$ in the relevance feedback scenario.

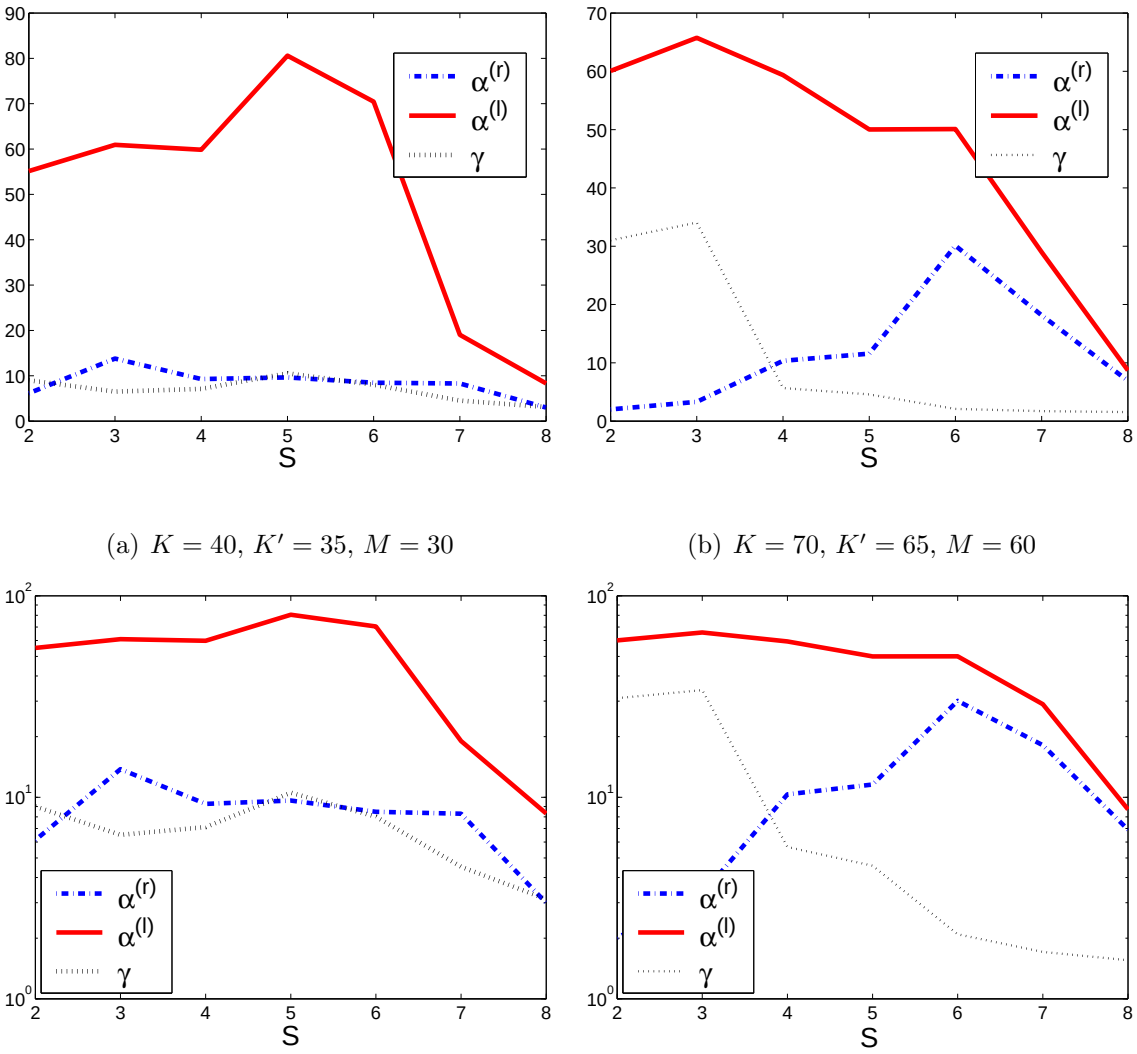
We presented an adaptive framework that supports efficient retrieval in iterative scenraios when the similarity metric is quadratic. The weight matrix of the feature




(a) $K = 40$, $K' = 35$, $M = 30$ (b) $K = 70$, $K' = 65$, $M = 60$

Figure 4.11: Phase I selectivity bounds 4.28 for quadratic Euclidean distance: $\rho$ for standard filtering, and $r_t^u$ and $l_t^u$ for adaptive one.

space is modified at every iteration based on the user's input, and a new set of nearest neighbors is computed. Nearest neighbor computation at each iteration is quadratic with the number of dimensions and linear with number of items. The proposed scheme enables the use of the user's feedback not only to improve the effectiveness of the similarity retrieval, but also its efficiency in an interactive content based image retrieval system. The proposed approach uses rectangular approximations for nearest neighbor search under quadratic distance metric and exploits correlations between two consecutive nearest neighbor sets. We believe that the proposed similarity search approach is one of the first attempts to address complexity issue of nearest neighbor update in relevance feedback and in high-dimensional feature spaces, while simultaneously reducing the overall search complexity. Future research directions could include exploring non-linear mapping of the feature spaces and efficient approximate search schemes.




(a) $K = 40$, $K' = 35$, $M = 30$

(b) $K = 70$, $K' = 65$, $M = 60$

(c) $K = 40$, $K' = 35$, $M = 30$

(d) $K = 70$, $K' = 65$, $M = 60$

Figure 4.12: Adaptive gain and logarithmic (base 10) scale adaptive gain (4.30) for quadratic Euclidean distance: $\alpha^{(r)}$ for adaptive Phase I search using $r_t^u$, $\alpha^{(l)}$ for adaptive Phase I search using $l_t^u$, and $\gamma$ for standard VA-file search.

# Chapter 5

# Adaptive Approximation Search

This chapter focuses on approximate nearest neighbor searches. One obvious motivation for the approximate nearest neighbor searches is to simplify the computations in high–dimensional feature spaces. An additional motivation in the context of multimedia feature vectors is that in many cases these searches are for retrieving items that are perceptually similar, and the feature vectors are only an approximate abstraction of the original data. As such, looking for precise nearest neighbors has little meaning. There have been some recent efforts within the multimedia community to address this problem of approximate nearest neighbor search [6, 60, 70, 32, 126]. In this chapter, we present a novel approach to approximate search and indexing of multimedia feature spaces. In this chapter, we propose an efficient search method that takes advantage of the apriori knowledge of feature vector distribution. The design of the index structure adapts to data's distribution and can support different similarity metrics. Thus, we avoid the computationally complex step of pre learning the data clusters [70], decorrelating the data [126] or learning data distribution along independent dimensions

[6]. For this reason, the proposed method can scale easily to large databases. We use MPEG-7 texture descriptor as an example. This approach can be generalized and applied to the other MPEG-7 feature vectors.

## 5.1 Introduction

In content based retrieval, the main task is seeking entries in a multimedia database that are most similar to a given query object. This problem is central to a wide range of applications in audio/visual databases. Given a collection of data, the first step in creating a database is to compute relevant feature descriptors to represent the content. The degree of similarity between two objects is then quantified by a distance measure operating on the extracted feature descriptor vectors. The exact nearest neighbors to a given query vector are determined based on that distance measure. In typical multimedia applications, the data quantity and feature vector dimensionality can be very demanding, thus reducing the overall system efficiency.

Computing exact nearest neighbors in high dimensions over large datasets is a very difficult task as we have seen in previous Chapters. Run-time, a measurement of the time required for a typical nearest neighbor search algorithm to run, does not scale well as a function of the input size. Note that selection and derivation of image features and the use of a distance metric in multimedia applications are rather heuristic, essentially perceived similarity in images and videos. In such circumstances, exact search and retrieval is often wasteful and expensive, since the perception of retrieving the "exact"

nearest neighbor set is subjective. It has been shown [60, 133] that by computing nearest neighbors approximately, it is possible to achieve significantly faster run times often with a relatively small actual errors.

Approximate nearest neighbor (ANN) searches are of particular interest in the case of large media databases where feature descriptors represent the data only approximately. Traditional ANN searching allows the user to select a maximum error bound $\epsilon$ [6], thus providing a tradeoff between accuracy and running time. Consider the set $\Psi$ of $N$ data points in Euclidean space $\mathbb{R}$. Given a query point $Q$ and a positive error bound $\epsilon$, a point $P$ is a $(1+\epsilon)$-approximate nearest neighbor if

$$d(P,Q) \leq (1+\epsilon)d(P',Q) \tag{5.1}$$

where $P'$ is the true nearest neighbor of $Q$ under distance metric $d(P,Q)$.

ANN search algorithms have lower processing cost at the expense on the accuracy of results. However, as reported in [32], ANN algorithms are still largely influenced by the dimensionality curse, and become impractical for larger dimensions. A probabilistic approach to error measurement of ANN relaxes the previous condition (5.1) in terms that it guarantees that the bound $\epsilon$ will not be exceeded in more than $\delta$ cases i.e.:

$$Pr\{|d(P,Q) \geq (1+\epsilon)d(P',Q)|\} \leq \delta \tag{5.2}$$

The probably approximately correct (PAC) nearest neighbor algorithm was proposed by Ciaccia and Patella in [32], essentially relaxing the requirements on error bound by adopting the well known PAC learning schema. This technique adds a second level of

approximation, where more objects can be pruned by allowing points to exceed the $(1+\epsilon)$-approximate nearest neighbor distance with some fixed probability $\delta$.

Compression-based approaches to indexing have proven to be the most adequate for large high–dimensional datasets as described in Chapter 2.5.3. ANN search based on the VA-file technique was proposed in [133]. The resulting retrieval set is created using only Phase I filtering. The upper bound on the distance in Phase I is tighter, thus reducing the number of visited vectors. The bound is an approximate one, guarantees a $(1+\epsilon)$ nearest neighbors if the dataset distribution is assumed to be uniform, and allows some false negatives. However, results reported in [133] demonstrate that this method is still sensitive to data distribution.

This can be explained with the VA-file structure, where approximations are constructed by using equally spaced grids on each dimension, as shown in Figure 5.1. A large portion of the database is accessed in the first filtering phase if the number of bits per dimension is small, or if approximation cells are overpopulated. As a result, the assumption that data is uniformly distributed is important for effective indexing. Real multimedia datasets are often highly skewed [45, 139, 13, 126]. Assuming the uniformity of the data distribution is too strong an assumption for real high–dimensional data.

Consider the distribution of the last 2 dimensions of the texture feature vector from the Aerial image collection, as shown in Figure 5.1. This Aerial collection consists of 40 large aerial photos divided into $64 \times 64$ tiles. The total number of tiles in the

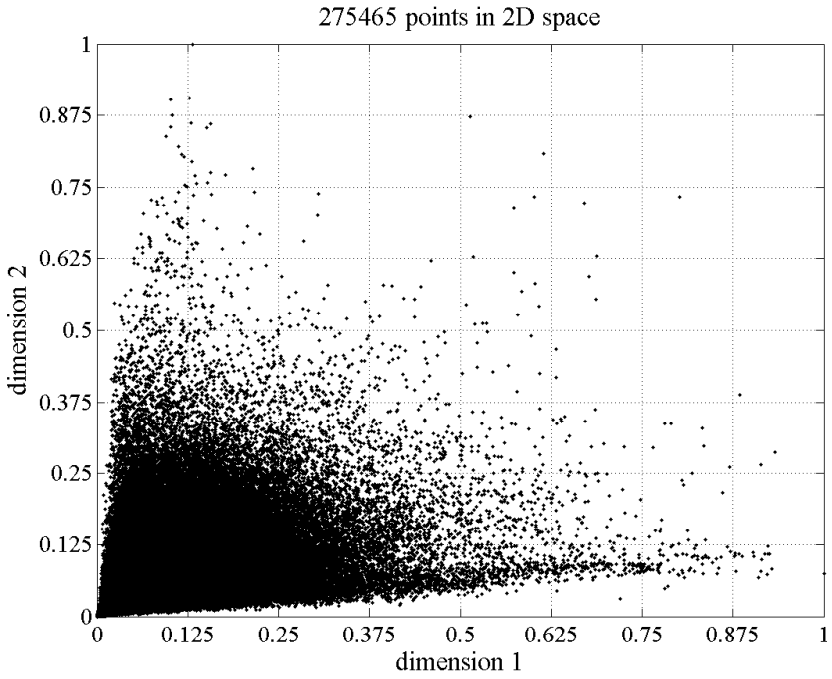



Figure 5.1: Two dimensional distribution of aerial data points.

collection is 275,465. For each image, the method described in Chapter 3.2 is used to extract a 60 dimensional texture feature descriptor $f_{\mu\sigma}$. The data along each dimension follows a Rayleigh-like distribution described in Chapter 3.4, and is not uniformly distributed. Consider the use of VA-file index of Chapter 4.2 for the data, and coding each dimension using $B = 3$ bits. In this case, 219292 out of 275465 2D vectors, or 79.6% of the database, are indexed by the same index, since they belong to the [0.125, 0.125] interval of the quantized code, see Figure 5.1. This example illustrates how a non-uniform data distribution can significantly degrade the indexing performance. Poor selectivity in Phase I filtering and a high processing cost for Phase II filtering for the large datasets make even sequential scan over the compressed data more efficient.

## 5.2 Previous work

Compression-based indexing structures in high dimensions are sensitive to the data distribution [13]. Therefore, Ferhatosmanoglu et al. [45] proposed to adapt the construction of the approximation to the statistical properties of data using transform coding [50]. The authors reduce the dimension of the data using an SVD-based algorithm to find principal components. Afterwards, they cluster the data in low-dimensional space using the K-means algorithm. NN search is then approximated on the closest clusters in lower dimensions. Converting feature vectors to a low-dimensional space reduces the complexity but it has negative effects on the retrieval performance, as described in Chapter 2.3. The experiments conducted on a much smaller dataset concluded that the dimensionality reduction algorithms like SVD do not provide a satisfactory object representation for low level descriptors in dimensions smaller than 10. The alternative is to investigate methods that can facilitate efficient search in high–dimensional feature spaces directly. In [126], the authors use a vector quantization technique to achieve better clustering results, and propose an approximate search only over a cluster into which the query point falls.

These proposed methods of [45] and [126] have similar drawback: their search efficiency depends on accurately estimating the principal components and data distribution, respectively, from only a fraction of the dataset. As reported in Chapter 2.3 and [19], estimation of data distribution based on data sampling is not valid for actual large high–dimensional datasets due to the "dimensionality curse". The complexity of

the distribution estimation grows exponentially with the dimension. Furthermore, if the density function is to be estimated based on a set of high–dimensional samples, the number of samples required for accurate density function estimation also grows exponentially. With a fixed number of training samples, the dimension for which accurate estimation is possible is severely limited to a small number, 6 at most [10]. Therefore, it is not possible to accurately estimate data distribution or the principal components from data sample for high–dimensional spaces.

In [139], the authors proposed an adaptive indexing structure that supports approximate search, where they estimate the marginal distribution of feature vectors on each dimension using Expectation Minimization algorithm. Then, each of the data dimensions is partitioned using optimal vector quantization [50], such that each bin contains and approximately equal number of objects. Their NN search is an exact one, but the authors do not address the complexity increase in exact search resulting from the variable-size compression scheme, model parameter initialization and estimation accuracy. In conclusion, the attempt to improve the efficiency of query processing in multimedia databases has tended towards improving either the index structures [45, 13], or the compressed data representations [139, 126]. In both tracts, authors assumed that there is no prior knowledge available on the indexed dataset.

However, it has been shown [10, 6] that a priori information can help with the curse of dimensionality. In general, data distribution can be irregular or not well modeled by some well defined function if the extraction of the data is unknown. If we have some

a priori information on what is the data distribution model, it is possible to overcome the dimensionality curse in general. In this chapter, we propose an adaptive indexing structure that relies on the a priori data distribution model. Our proposed method benefits from a characteristic of multimedia feature descriptors that form the high–dimensional database. This method enables efficient ANN search over large multimedia databases without compromising retrieval quality.

## 5.3 Indexing Performance and Data Distribution: HTD Example

The ISO/MPEG-7 international standard [78] provides a detailed explanation of how low-level descriptors and metadata are extracted from the images. These low–level descriptors form high–dimensional feature spaces. Based on the extraction method of MPEG-7 homogenous texture descriptor, as described in Chapter 3.2, we modeled the distribution of the texture feature dataset along each dimension as generalized Rice distribution. In this section we use the MPEG-7 homogeneous texture descriptor distribution as an example to demonstrate the creation of adaptive index structures. The same paradigm, however, can be applied to the other MPEG-7 feature vectors, such as the the edge histogram descriptor (EHD) and the scalable color descriptor (SCD). Scalable Color descriptor can be interpreted as a Haar transform-based encoding scheme applied across values of a color histogram in the HSV color space. Edge

histogram represents local edge distribution in an image. In general, both features are constructed using statistics from the band-pass filter outputs, and their distribution along one dimension can be modeled using parametric distribution function.

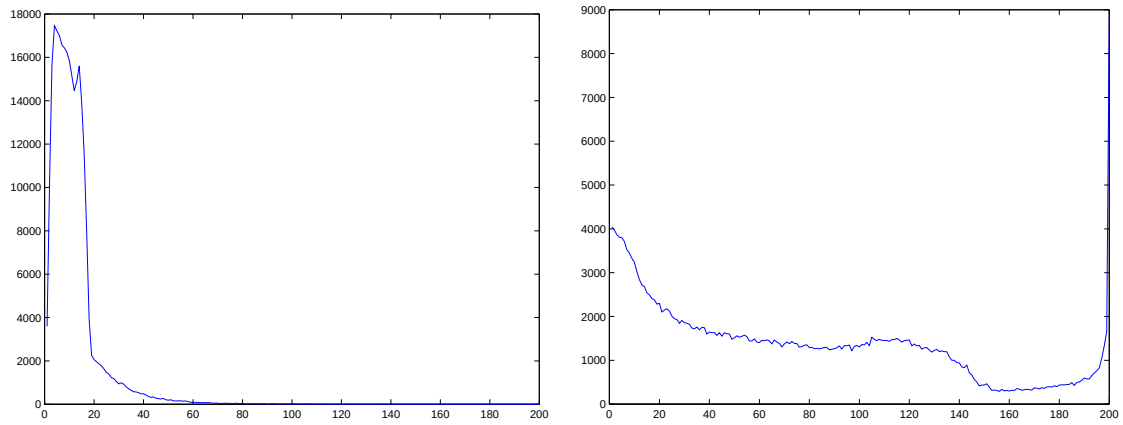
In Chapter 3.4, we showed that the homogeneous texture feature data distribution along each dimension $j$ can be approximated well by a generalized Rayleigh distribution. In that case, Rayleigh parameter $\gamma_j$ is estimated in (3.21):

$$\hat{\gamma_j^2} = \frac{1}{2N} \sum_i |f_{ij}|^2. \tag{5.3}$$

Moreover, we proposed Rayleigh equalization approach in Chapter 3.4.1 to normalize the data along each dimension using parameter $\hat{\gamma_j}$. Rayleigh Equalization gives a better precision/recall results than if we use only Gaussian normalization method, and it forces the uniform distribution of feature along each dimension.

A typical texture feature distribution along one dimension is shown in Figure 5.2. This Aerial dataset consists of over 250K feature vectors. The figure shows histograms with 200 bins for the raw MPEG-7 texture feature descriptor. Figure 5.2(a) shows the Gaussian normalized histogram and Figure 5.2(a) shows the rayleigh normalized histogram. In the case of uniform distribution, if each of the data dimensions is partitioned into equal bins, each bin should contain approximately an equal number of objects.

If the features had uniform distribution, expected number of vectors in a histogram bin would be 1377. In Figure 5.2(a), median number of feature vector in a bin is 6, and maximum value is 17466. Over 80% of the space is underpopulated, and we have




(a) $\gamma_{52}^{(SN)}$ histogram

(b) $\gamma_{52}^{(RE)}$ histogram

Figure 5.2: Histograms with 200 bins of the texture feature distribution along one dimension for 275465 Aerial feature set when the (a) Gaussian Normalization method, and (b) Rayleigh Equalization method is applied.

15 overpopulated bins with over 10000 elements. Variance is 1377. in this scenario, any indexing scheme is going to be very inefficient due to the uneven bin population.

However, in Figure 5.2(b), median value is 1339 median value and it is very close to the median value of data. Maximum value is 8845, and it is a lonely population peak on the interval border. Variance is 992. Maximum value of the rest of the bins is 4046. Using equalization, we have populated most of the cells with similar number of feature vectors. We observe this behavior in each of the feature dimensions. Data variance is reduced by 50% (on average) and, therefore, equalization reduces the number of overpopulated and underpopulated cells.

Indexing structure is going to benefit from this uniform-like distribution. The key element to taking advantage of indexing structures built for uniformly distributed data,

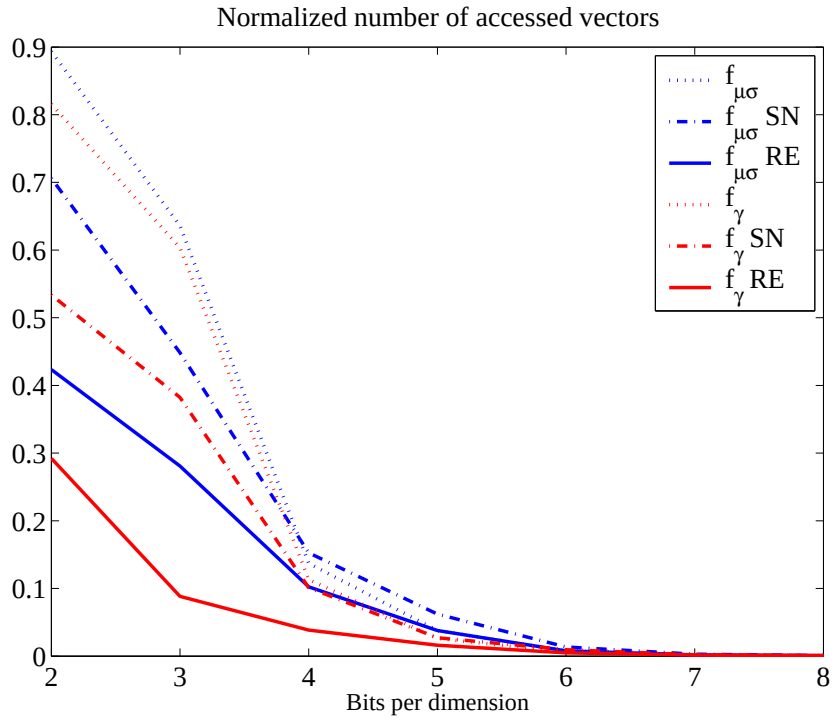



Figure 5.3: Normalized number of the feature vectors accessed after VA filtering phase for $K = 20$ for standard HTD $f_\mu\sigma$, modified feature vector $f_\gamma$, and Gaussian normalization and Rayleigh equalization.

like VA-files, is to create bins in the high–dimensional space that are approximately equally populated. To evaluate this claim, we will now consider the number of feature vectors visited in a nearest-neighbor search (NN-search) using the standard VA-File (Chapter 4.2) over the different normalization methods of the same dataset. Experiments are conducted similar to that described in Chapter 4.6. We tested the approach using 2,3,4,5,6,7 and 8 bits for each dimension to construct the approximation. For each approximation, we consider the queries to be all image items in the database. We normalized the Homogenous Texture Descriptor dataset $\{f\}$ using Gaussian normalization $\{f^{SN}\}$ and Rayleigh Equalization $\{f^{RE}\}$, as described in Chapter 3.4.1.

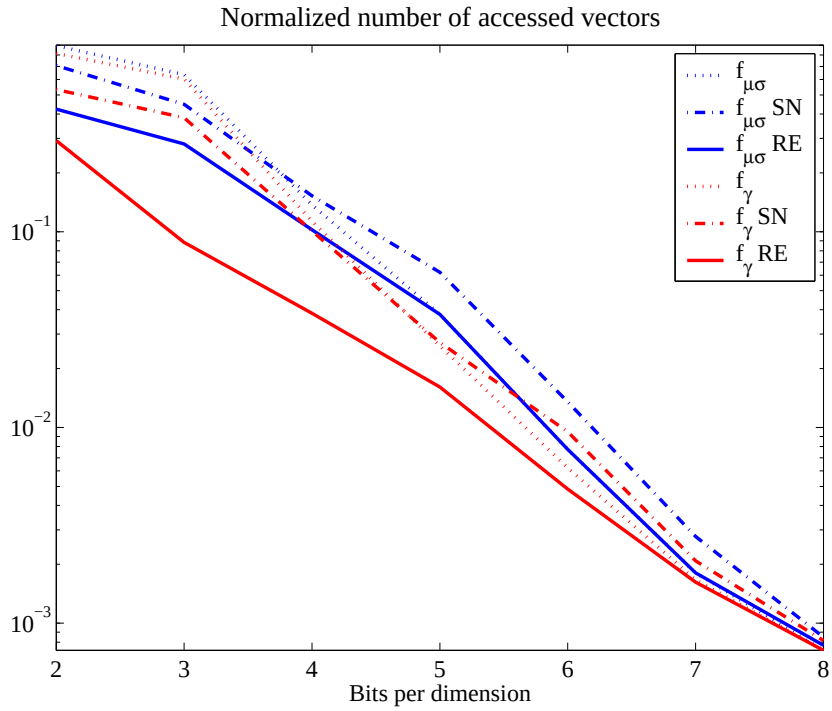



Figure 5.4: Log scale of normalized number of the feature vectors accessed after VA filtering phase for $K = 20$ for standard HTD $f_\mu\sigma$, modified feature vector $f_\gamma$, and Gaussian normalization and Rayleigh equalization.

Experiments were conducted using $\{f\}$, $\{f^{SN}\}$, and $\{f^{RE}\}$ datasets for HTD database $\{f_{\mu\sigma}\}$ and $\{f_\gamma\}$. Initially, around 10% of total number of image objects are used to estimate parameter $\gamma$ for Rayleigh equalization for each database. We compare the number of index candidates from VA Phase I filtering for $K = 20$.

The results over a database of 100,000 texture feature vectors are shown in Figure 5.3. The figure shows that the proposed equalization improves the efficiency of NN search for uniform indexing approach. The effectiveness of the proposed method is significant for lower resolutions, as shown in Figure 5.4. The number of candidates is reduced up to 3 times just by using Rayleigh Equalization of the modified feature

vector $\{f_\gamma\}$ instead of standard HTD $\{f_{\mu\sigma}\}$.

In conclusion, the indexing performance is improved, since the constructed adaptive approximation reduces the possibility of having overpopulated or underpopulated cells. The feature vector distributions along each dimension are easy to compute and offer significant overall performance improvement. This examples demonstrates how the performance of compression-based indexing is sensitive to data distribution, and how it can be improved. By adapting the VA file index to the marginal distribution of the data, indexing efficiency can be improved. Therefore, we propose an adaptive indexing structure that relies on the a priori data distribution model.

### 5.3.1 Adaptive VA-file Indexing

Adaptive indexing structure benefits from a characteristic of multimedia feature descriptors that form the high–dimensional database. When there are correlations between dimensions, index techniques tend to benefit [13]. Based on the explored characteristics of the feature dataset, we propose a 2-tier VA-file based indexing structure using existing dependency relations in MPEG-7 HTD feature datasets, as explained in Chapter 3 (see 3.3.2 and 3.4. The Algorithm 2 presents the steps for constructing the index.

In the first step, we incorporate the prior knowledge of relations among texture feature space dimensions to make the search dataset more compact. Since we have

**Algorithm 2** Construction of Adaptive VA-file index

**for all** $\hat{f}_{\mu\sigma} \in$ HTD database **do**

1. Compute $\hat{f}_{\gamma}$ from $\hat{f}_{\mu\sigma}$ as in (5.4)
2. Compute $\hat{\gamma_j}, \forall j$ as in (5.5)
3. Equalize all $\hat{f}_{\gamma}$ to $\hat{f}_{\gamma}^{RE}$ as in (5.6)
4. Construct and store an adaptive index $C_a(f)$ of $\hat{f}_{\mu\sigma}$ as an VA-file index $C(f)$ of $\hat{f}_{\gamma}^{RE}$

**end for**

pre-computed MPEG-7 feature $\vec{f}_{\mu\sigma}$, it is easy to compute $\vec{f}_{\gamma}$ as:

$$f_{\gamma,ij}^2 = \frac{1}{2}\left(f_{\mu,ij}^2 + f_{\sigma,ij}^2\right). \tag{5.4}$$

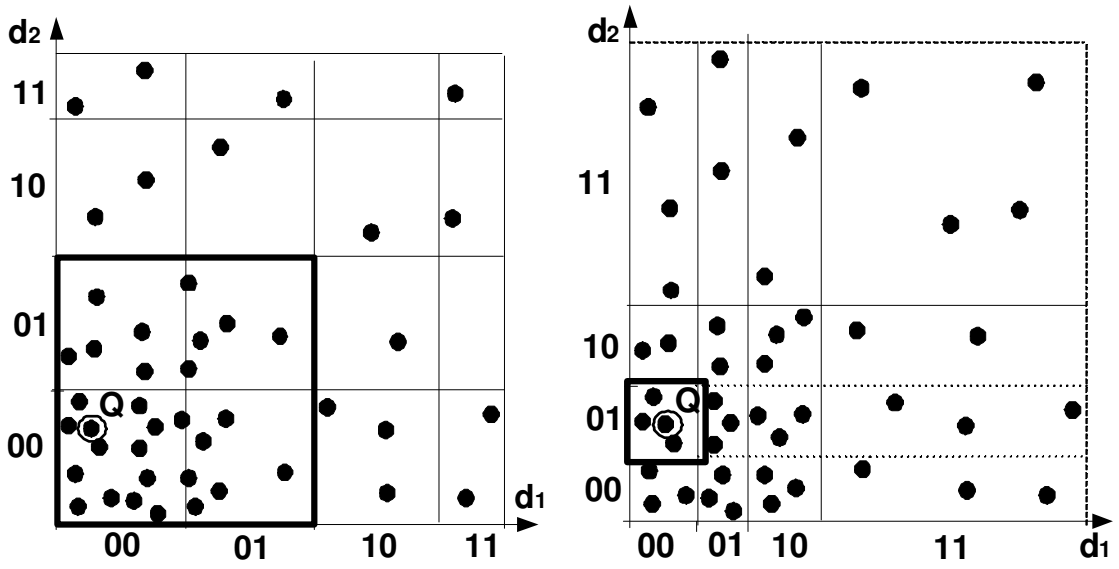
Next, we estimate the Rayleigh parameters $\hat{\gamma_j}$ along each dimension from a fraction N of $\vec{f}_{\gamma}$ vectors:

$$\hat{\gamma_j^2} = \frac{1}{2N}\sum_i |f_{\gamma,ij}|^2. \tag{5.5}$$

In the third step, we equalize all $\hat{f}_{\gamma}$ to $\hat{f}_{\gamma}^{RE}$ using $\hat{\gamma_j}$ as in:

$$f_{\gamma,ij}^{(RE)} = 1 - e^{-\frac{f_{\gamma,ij}^2}{2\hat{\gamma_j^2}}}. \tag{5.6}$$

In the last step, we construct the index $C_a(f)$ of $\hat{f}_{\mu\sigma}$ as an VA file index $C(f)$ of $\hat{f}_{\gamma}^{RE}$ as described in Chapter 4.2. In this manner, we have populated VA cells with similar number of items using a priori data distribution, and we did not change the format of data in the database. Note that the first three steps were intermediate, and results are not stored. What is stored is an adaptive VA-file index for the original HTD dataset.




(a) VA-File search

(b) Approximate Adaptive Search

Figure 5.5: Black Rectangle marks Phase I candidates for (a) traditional VA file search, and (b) Approximate adaptive search.

Adaptive index constructed in this way reduces the possibility of having densely populated cells, i.e. the number of vectors indexed by each cell is approximately the same, and the distribution within the cells is uniform. This, approach reduces the number of Phase I candidates significantly. Advantages of the proposed method are illustrated using two–dimensional example in Figure 5.5. In Figure 5.5(b) the adapted index of the query vector is computed according to the proposed scheme we are searching over a modified VA-file index, adapted to texture feature distribution.

## 5.4 Approximate Search over Adaptive Index

The cost of nearest neighbor search over VA-file is linear with number $N_1$ of candidate vectors and the number $N_2$ of actual visited feature vectors [133]. The search becomes very expensive for smaller number of bits $S$ assigned to each cell dimension. Note that $S$ has an upper limit due to the dataset size and memory size. Underlying assumption is that the index fits into main memory, and the need to balance size of the index with the cost of nearest neighbor search is imperative in large high–dimensional datasets.

If the index is constructed as in Algorithm 2, the distribution of index over a database is more uniform. That results in a smaller number of candidates in Phase I search, see Figure 5.4. However, nearest neighbor search over adaptive VA-file index is not an exact one. Phase I filtering over adaptive VA-index may introduce some false negatives, and nearest neighbor search is an approximate one.

**Lemma 1** If the $d_{\mu\sigma}(Q, F)$ is the Euclidean distance between query vector $Q$ and database vector $F$ in the original HTD $\vec{f}_{\mu}\sigma$ database, and $d_{\gamma}(Q, F)$ is the distance between query point $Q$ and database vector $F$ in the $\vec{f}_{\gamma}$ database, where $\gamma$ coefficients are computed according to (5.4), the following inequality holds:

$$d_{\mu\sigma}(Q, F)^2 \geq 2d_{\gamma}(Q, F)^2 \tag{5.7}$$

**Proof** Equation (5.7) states that the distance in the HTD space is lower bounded by the distance in the compact space. $d_{\mu\sigma}(Q,F)$ is computed as $d_{\mu\sigma}(Q,F)^2 = \sum_{j=0}^{N}(Q_{\mu,j} - F_{\mu,j})^2 + (Q_{\sigma,j} - F_{\sigma,j})^2$. Similarly, $d_{\gamma}(Q,F)^2 = \sum_{j=0}^{N}(Q_{\gamma,j} - F_{\gamma,j})^2$, where $F_{\gamma,j}^2 = \frac{1}{2}\left(F_{\mu,j}^2 + F_{\sigma,j}^2\right)$. according to (5.4). Therefore, we need to prove that

$$\sum_{j=0}^{N}(Q_{\mu,j} - F_{\mu,j})^2 + (Q_{\sigma,j} - F_{\sigma,j})^2 \geq 2\sum_{j=0}^{N}(Q_{\gamma,j} - F_{\gamma,j})^2 \tag{5.8}$$

This holds true if for $\forall j$, the following inequality holds:

$$(Q_{\mu,j} - F_{\mu,j})^2 + (Q_{\sigma,j} - F_{\sigma,j})^2 \geq 2(Q_{\gamma,j} - F_{\gamma,j})^2. \tag{5.9}$$

Since $Q_{\gamma,j}$ and $F_{\gamma,j}$ are computed using (5.4), this inequality can be expressed as:

$$(Q_{\mu,j} - F_{\mu,j})^2 + (Q_{\sigma,j} - F_{\sigma,j})^2 \geq (\sqrt{Q_{\mu,j}^2 + Q_{\sigma,j}^2} - \sqrt{F_{\mu,j}^2 + F_{\sigma,j}^2})^2. \tag{5.10}$$

Since the left side of (5.10) can be expressed as: $Q_{\mu,j}^2 + F_{\mu,j}^2 - 2Q_{\mu,j}F_{\mu,j} + Q_{\sigma,j}^2 + F_{\sigma,j}^2 - 2Q_{\sigma,j}F_{\sigma,j}$, and the right side as: $Q_{\mu,j}^2 + Q_{\sigma,j}^2 + F_{\mu,j}^2 + F_{\sigma,j}^2 - 2\sqrt{(Q_{\mu,j}^2 + Q_{\sigma,j}^2)(F_{\mu,j}^2 + F_{\sigma,j}^2)}$,, (5.10) can be rewritten as:

$$\sqrt{(Q_{\mu,j}^2 + Q_{\sigma,j}^2)(F_{\mu,j}^2 + F_{\sigma,j}^2)} \geq Q_{\mu,j}F_{\mu,j} + Q_{\sigma,j}F_{\sigma,j}. \tag{5.11}$$

Both sides of inequality (5.11) are positive values. Therefore, this inequality holds if:

$$(Q_{\mu,j}^2 + Q_{\sigma,j}^2)(F_{\mu,j}^2 + F_{\gamma,j}^2) \geq (Q_{\mu,j}F_{\mu,j} + Q_{\sigma,j}F_{\sigma,j})^2, \tag{5.12}$$

$$Q_{\mu,j}^2F_{\mu,j}^2 + Q_{\sigma,j}^2F_{\mu,j}^2 + Q_{\sigma,j}^2F_{\sigma,j}^2 \geq Q_{\mu,j}^2F_{\mu,j}^2 + Q_{\sigma,j}^2F_{\sigma,j}^2 + 2Q_{\mu,j}Q_{\sigma,j}F_{\mu,j}F_{\sigma,j}. \tag{5.13}$$

It follows that the inequality (5.7) is true if

$$(Q_{\mu,j}F_{\sigma,j} + Q_{\sigma,j}F_{\mu,j})^2) \geq 0. \tag{5.14}$$

Since (5.14) is a true statement for all $Q$ and $F$, Lemma 1 is proved.

**Corollary 1** For adaptive index $C_a(F)$ to be a qualified one in $N_1$, it is necessary and sufficient that its lower bound $L_\gamma(Q, F)$ satisfies

$$L_\gamma(Q, F) < \rho_{\mu\sigma}. \tag{5.15}$$

The nearest neighbor (NN) filtering process in the Phase I, as described in Chapter 4.2 uses information on the lower bounds $L(Q, F)$ and upper bounds $U(Q, F)$ of the distance between a query point $Q$ and a feature vector $F$. For standard VA-file index $C(F)$ to be a qualified one, inequality $L_{\mu\sigma}(Q, F) < \rho_{\mu\sigma}$ should hold. $\rho_{\mu\sigma}$ is the $K^{th}$ largest upper bound of the standard VA-file indices in $\{\vec{f}_{\mu\sigma}\}$ database. Adaptive index $C_a(F)$ is constructed over the compact $\{\vec{f}_\gamma\}$ database. There, from (4.11), the following inequalities hold, :

$$L_{\mu\sigma}(Q, F)^2 \leq d_{\mu\sigma}(Q, F)^2 \leq U_{\mu\sigma}(Q, F)^2. \tag{5.16}$$

From (5.7) it follows that

$$2L_\gamma(Q, F)^2 \leq 2d_\gamma(Q, F)^2 \leq d_{\mu\sigma}(Q, F)^2 \leq U_{\mu\sigma}(Q, F)^2. \tag{5.17}$$

Therefore, $\rho_{\mu\sigma}$ is an upper bound for $L_\gamma(Q, F)$.

Since $\rho_{\mu\sigma}$ is computed in the original space, it is not possible to compute it over adaptive $C_a(F)$ indices in the main memory. In $\{\vec{f}_\gamma\}$ database, from (4.11) it follows

that:

$$L_\gamma(Q,F)^2 \leq d_\gamma(Q,F)^2 \leq U_\gamma(Q,F)^2 \tag{5.18}$$

Define $\rho_\gamma$ as the $K^{th}$ largest upper bound of the standard VA-file indices in $\{\vec{f_\gamma}\}$ database. If we use only $\rho_\gamma$ as a Phase I filtering bound for the lower bound on distance between the query vector and adaptive VA-file index, the computations of a Phase I candidates is reduced to using only the index structure in the main memory. Therefore, if an VA-file index is encountered such that its lower bound is larger than $\rho_\gamma$, the corresponding feature vector can be skipped since at least $K$ better candidates exist. This bound is not guaranteed to include all the nearest neighbor search over adaptive VA-file index, since $\rho_\gamma$ is a tighter bound than $\rho_{\mu\sigma}$. Therefore, it is possible that $\rho_\gamma$ can filter out some exact nearest neighbors. Experiments show that, for high dimensions and large dataset, the accuracy of adaptive filtering that uses $\rho_\gamma$ is high, see Section 5.5.

In Chapter 5.3 we have shown that if the data distribution is skewed and the compression rate is large (small $S$) the Phase I of VA-file filtering still results in a large number of candidate items to access for during the Phase II search. To overcome this problem, all $\hat{f}_\gamma$ are equalized to $\hat{f}_\gamma^{RE}$ using mapping from (3.24): $f_j \rightarrow f_j^{(RE)}$ : $f_j^{(RE)} = 1 - e^{-\frac{f_j^2}{2\hat{\gamma}_j^2}}$. This is a monotonically increasing function that maps one feature dimension to [0,1] interval. The topology of the space stays the same i.e. if $A_j < B_j \rightarrow A_j^{(RE)} < B_j^{(RE)}$. Since the adaptive indexing enforces more even spread of features over the index structures, Phase I filtering over equalized dataset is more restrictive. First, the filtering using equalized index does not have a negative influence on a search

performance over $\rho_\gamma$. Second, adaptive VA cells are populated with similar number of items using a priori data distribution, and false positives are "pushed" away from the query, and therefore filtered out in Phase I.

In Phase II, the $K$ nearest neighbors are found in the original feature space. The actual feature vectors, whose indices belong to a candidate set, $N_1$, are accessed. The feature vectors are visited in increasing order of the computed lower bounds in the Phase I filtering. From (5.7) we have that $L_\gamma(Q,F) \leq d_\gamma(Q,F) \leq d_{\mu\sigma}(Q,F)\frac{1}{\sqrt{2}}$. In this scenario, scaled $L_\gamma(Q,F)$ can be used as computed lower bounds in the Phase II filtering. However, in approximate adaptive Search, the Phase I filtering is modified. Since lower bound is computed in the $\{\vec{f}_\gamma^{(RE)}\}$, it is not a measure for the filtering bound in the original feature space. Therefore, we introduce another level of approximations for approximate search over $C_a(F)$ index. Lower bounds in the $\{\vec{f}_\gamma\}$ database are estimated from computed lower bound $L_\gamma^{(RE)}(Q,F)$ in $\{\vec{f}_\gamma^{(RE)}\}$ database, based on the mapping (5.7), as:

$$L_\gamma(Q,F)^2 \sim \sum_{j=0}^{N} 2\gamma_j^2 L_{\gamma,j}^{(RE)}(Q,F). \tag{5.19}$$

Approximate adaptive nearest neighbor search algorithm is: **Phase I** In this phase, the set of all adaptive VA-files $C_a(F)$ is scanned sequentially and lower $L_\gamma^{(RE)}(Q,F)$ and upper bounds $U_\gamma^{(RE)}(Q,F)$ on the distances of each object in the database $F$ to the query object $Q$ are computed. During the scan, a buffer is used to keep track of $\rho_\gamma$, the $K^{th}$ largest upper bound found from the scanned indices. If an index $C_a(F)$ is encountered such that its lower bound is smaller than $\rho_\gamma$, the index $C_a(F)$ will

be selected as a candidate and its upper bound will be used to update the buffer, if necessary. Then, the lower bound $L_\gamma(Q, F)$ is estimated as in (5.19) and stored. Note that $\gamma_j$ are computed during the adaptive indexing phase. The resulting set of candidate objects at this stage is $N_1$, and the cardinality of the set is $|N_1|$.

**Phase II** - In this phase, the $K$ nearest neighbors are found. The actual feature vectors, whose indices belong to a candidate set, $N_1$, are accessed. The feature vectors are visited in increasing order of their stored lower bounds and the exact distances to the query vector are computed using (4.3). If a lower bound is reached that is larger than the $K^{th}$ actual nearest neighbor distance encountered so far, there is no need to visit the remaining candidates. $N_2$ is the set of objects visited before the lower bound threshold is encountered. The $K$ nearest neighbors are found by sorting the $|N_2|$ distances.

## 5.5 Evaluation

In Section 5.4, we developed an adaptive indexing scheme to demonstrate its usage for realistic applications. The objective of the proposed method evaluation is to present the relation between the granularity and the approximation search quality, and the one between the granularity and the search performance on a real dataset.

The irrelevant vectors in the result are called false positives and the relevant vectors that are not in the answer set are called false negatives. Let $T(\vec{f})$ be the retrieved set with cardinality $T$, and $C(\vec{f})$ be the collection of images relevant to $\vec{f}$. The *precision*

measures the number of relevant feature vectors in the answer set as:

$$P(\vec{f}) = \frac{\left|C(\vec{f}) \cap T(\vec{f})\right|}{\left|T(\vec{f})\right|}. \tag{5.20}$$

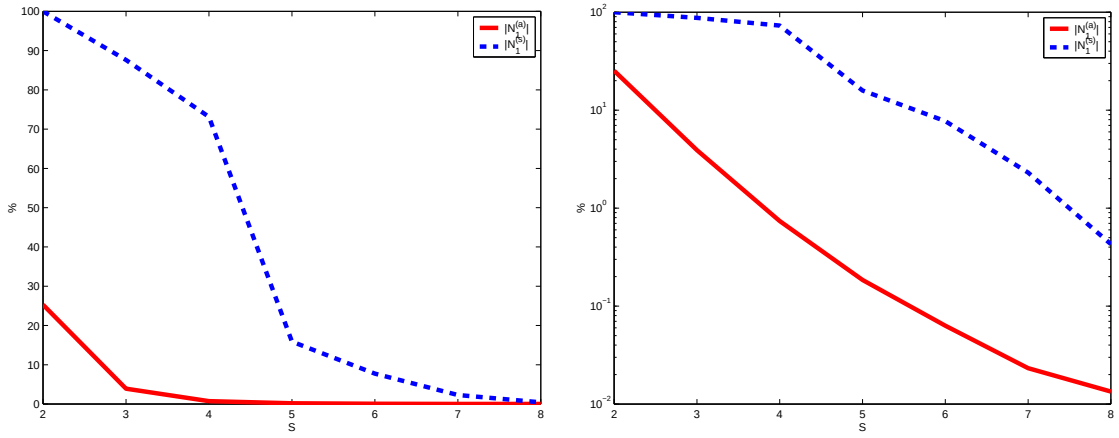
$1-P(\vec{f})$ is the number of false positives. *Recall* measures the percentage of the retrieved data in the exact answer set:

$$R(\vec{f}) = \frac{\left|C(\vec{f}) \cap T(\vec{f})\right|}{\left|C(\vec{f})\right|}. \tag{5.21}$$

$1 - R(\vec{f})$ is the number of false negatives. $|\cdot|$ denotes cardinality of the set. Initially, around 10% of total number of image objects are used to initialize the $\gamma$ parameter. Based on the estimated parametric distribution, dataset is equalized. The VA-file index is constructed over $\{\vec{f}_{\gamma}^{(RE)}\}$ database. The approach that uses the standard VA-File is also implemented for comparison purpose. The number of candidates from the first phase filtering $N1$ is used to evaluate the performance.

### 5.5.1 Experiments

Our preliminary evaluation is performed on an Aerial image database of 275,465 images. For each image, the texture descriptor HTD as introduced in Chapter 3 is adopted to compute a 60 dimensional feature vector to characterize image's texture feature. We have used $S \in [2, 3, 4, 5, 6, 7, 8]$ bits for each dimension to construct the index. For each $S$, $K = 20$ nearest neighbors search is performed on the whole database. Note that the adapted index has 30 dimensions in comparison to the 60 dimensions of the VA-file index of MPEG-7 texture feature vectors. Therefore, the index is two



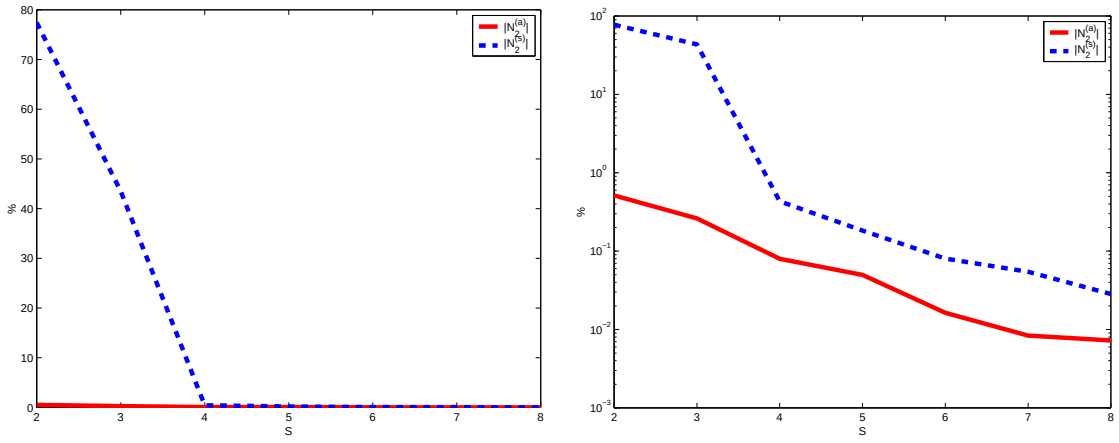

(a) $N_1$ (b) $\log(N_1)$

Figure 5.6: The number of Phase I candidates for traditional VA-file search $N_1^{(s)}$, and approximate adaptive VA-file search $N_1^{(a)}$.

times smaller, resulting in significant savings on size of the index.

We first evaluate the proposed approximate adaptive search on a real dataset using different index granularities. The number of candidates remaining after the Phase I and Phase II filtering for the exact search over VA-File (dotted line), and adaptive VA-file (dashed line), and the proposed approximate adaptive search (full line) are shown in Figure 5.6 and Figure 5.7 respectively.

The number of vector visited by traditional VA-file is up to 100 times ($S = 4, 5, 6, 7$) more than the proposed adaptive method. For $S = 2$, only 25% is selected using adaptive method, comparing to accessing the whole database using the standard method. However, in order to fit the whole index of large database into the main memory, higher compression rate is required. For $S = 2, 3$ savings are over 150 times in the Phase II, as shown in Figure 5.7(a). Since the underlying data distribution is assumed to be



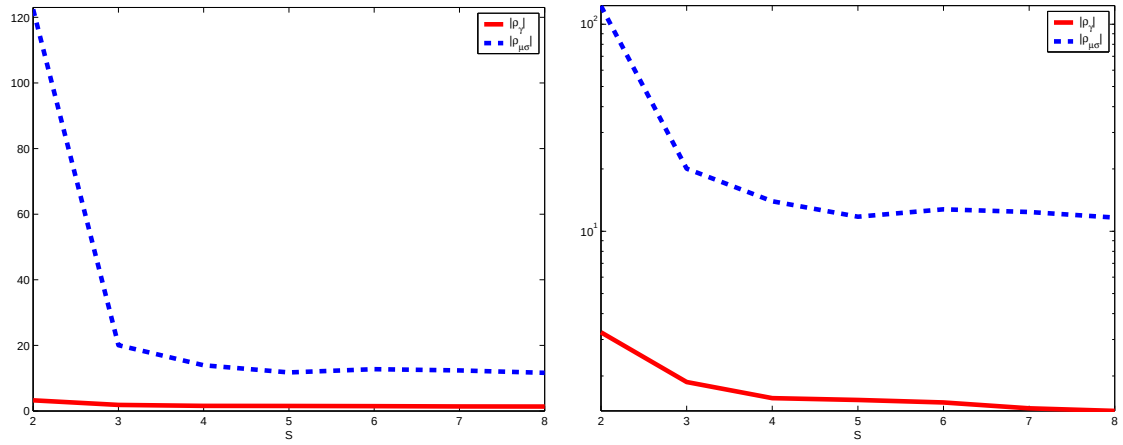

(a) $N_2$ (b) $\log(N_2)$

Figure 5.7: The number of Phase II candidates for traditional VA-file search $N_2^{(s)}$, and approximate adaptive VA-file search $N_2^{(a)}$.

uniform, the number of actually accessed feature vectors in the Phase II is also proportionally small. Adaptive VA-file indexing proved to be a great solution for overcoming computational bottleneck for any granularity $S$, both in Phase I and Phase II.

Figure 5.8 shows the relationship between the standard upper bound $\rho_{\mu\sigma}$ from (5.15) and the approximate upper bound of estimated $\rho_\gamma$ from (5.18). $\rho_{\mu\sigma}$ is scaled down by $\sqrt{2}$ to be comparable to $\rho_\gamma$ as a Phase I filtering bound. Estimated $\rho_\gamma$ sets a tighter upper bound, thus significantly reducing the number of false negatives. Since this is an approximate bound, some false positives may occur.

Second evaluation is between the granularity and the approximation search quality. Figure 5.9 shows the precision vs. recall curves for each method for Euclidean distance. The curves are plotted by averaging precision and recall over number of bits $S$ used per dimension. Size of the retrieved set is $K = 20$, and the retrieved set varies from




(a) $\rho$ (b) $\log(\rho)$

Figure 5.8: The filtering bound for Phase I candidates $\rho_{\mu\sigma}$ for traditional VA-file search, and $\rho_\gamma$ for approximate adaptive VA-file search. $\rho_{\mu\sigma}$ is scaled down by $\sqrt{2}$ to be comparable to $\rho_\gamma$) as a Phase I filtering bound.

5 (high precision, low recall) to 100 (low precision, high retrieval). Note that the underlying assumption in this approach is that the adaptive VA-file index cells are equally populated. This assumption is supported by preprocessing of texture features using Rayleigh Equalization model. Second assumption is that the number of feature vectors with the same adapted index is significantly larger that the $K$-NN we are trying to recover using this method. This assumption is important because it lowers down the number of false negatives. The resulting retrieval set is created using only Phase I filtering.

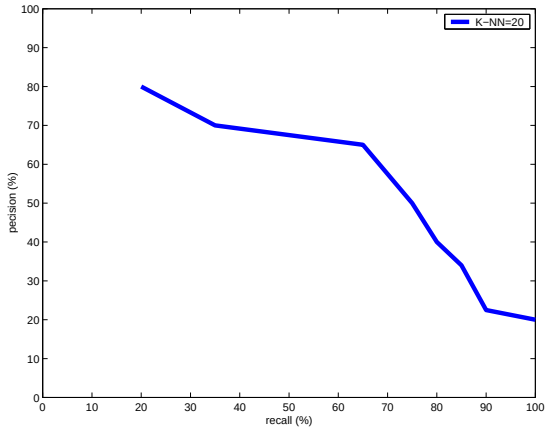
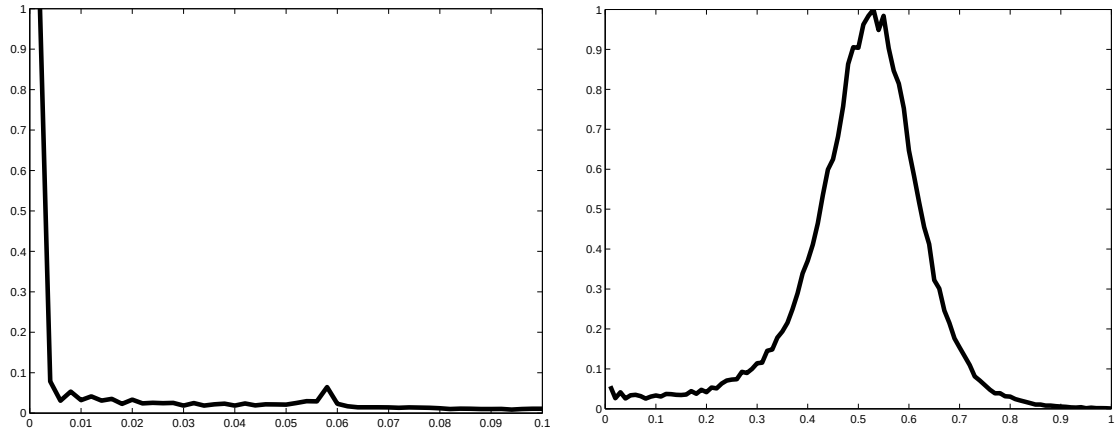



Figure 5.9: Precision–Recall curve for the approximate adaptive search for different size of the retrieval set.

## 5.6 Discussion

We have presented a novel adaptive indexing scheme to index high–dimensional image/video objects. The design of the index structure adapts to the data distribution and can supports various distance metrics. Data distribution is assumed to have a parametric form and parameters are easily derived from the data sample. By using a parametric estimation of a dataset, we were able to reduce the complexity of nearest neighbor search over large high–dimensional datasets. Rayleigh approximation and equalization of data proved to be a very good regularization approach for large MPEG-7 texture feature datasets. Overall, we have demonstrated that a search mechanism can benefit from the characteristic of multimedia feature descriptors. We proposed an adaptive indexing structure that enables efficient ANN search over large multimedia databases without significantly compromising retrieval quality.

As we mentioned before, the same framework can be applied to the other MPEG-




(a) SCD (b) EHD

Figure 5.10: Typical distribution along one dimension of (a) scalable color descriptor, and (b) edge histogram descriptor over an online image dataset.

7 feature vectors, like the the edge histogram descriptor (EHD) and THE scalable color descriptor (SCD). Scalable Color descriptor is formed from the outputs of the Haar function wavelets over a color histogram in the HSV color space. The distribution along any feature dimension is observed to follow an exponential distribution, as shown in Figure 5.10(a). Similarly, the edge histogram descriptor along any one dimension is the relative frequency of occurrence of one of the 5 types of edges in the corresponding sub-image: vertical, horizontal, 45-degree diagonal, 135-degree diagonal, and non-directional edges. The histogram value is obtained using A specific filter for each of these edges [78]. These filters are simple band-pass filters. The distribution along a feature dimension is observed to follow a Gaussian distribution similar, as shown in Figure 5.10(b). Thus, the proposed scheme can be extended to both the EHD and SCD using parametric models similar to the HTD.

# Chapter 6

# Multimedia Mining in High Dimensions

This chapter describes a framework for applying traditional data mining techniques to the non-traditional domain of image datasets for the purpose of knowledge discovery. We introduce a novel data structure termed Spatial Event Cube (SEC) for conceptual representation of complex spatial arrangements of image features in large multimedia datasets. A primary contribution of this chapter is the derivation of image equivalents for the traditional association rule components, namely the items, the itemsets, and the rules.

## 6.1 Introduction

Humans can instantly answer the question "Is this highway going through a desert?" just by looking at an aerial photograph of a region. This query, essentially formulated as a high–level concept, cannot be answered by most existing intelligent image analysis

systems. Existing image representations based on low-level features fail to capture perceptual events. Meaningful semantic analysis and knowledge extraction require data representations that are more understandable at a conceptual level. Compounding the urgency for new representations is the rapid rate at which multimedia data is being acquired. The value of these sizable datasets extends beyond what can be realized by traditional "focused" computer vision solutions, such as face detection, object tracking, and segmentation.

We present new methods of analysis based on data mining techniques to discover the aforementioned implicit patterns, relationships and other knowledge that are not readily observable. Data mining techniques have been used for some time to discover implicit knowledge in transaction databases. In particular, methods are available for determining the interesting associations among itemsets over large numbers of transactions, such as among the products that are most frequently purchased together, useful in market basket analysis. Achieving similar success with multimedia datasets is a challenge not only due to the size and complexity of image and video data, but also due to the nature of image content descriptors that often do not capture well the underlying semantics. We propose a framework whose information representation allows meaningful data summarization for efficient image dataset understanding at a coarse level. The framework is scalable with respect to dataset size and dimension, multi-feature representation, thus allowing fast data processing.

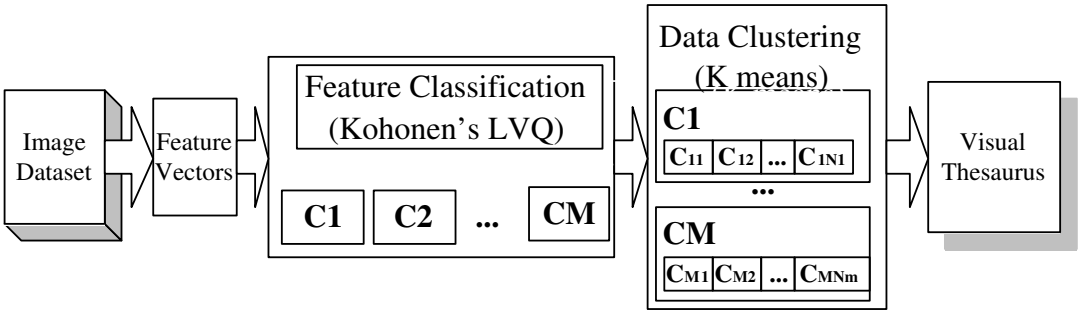



Figure 6.1: The construction of Visual Thesaurus

## 6.2 Visual Texture Thesaurus

Limited success towards the description of image content has been achieved by systems that use low-level visual features, such as texture and color descriptors, to represent the images. However, alone these systems fail to support high–level perceptual interaction.

A concept of visual thesaurus was introduced as one of the first attempts to organize the data information derived from the low-level features and to assign a semantic meaning to a cluster of those features, [73]. The visual thesaurus is constructed to label the video frame tiles in a perceptually meaningful way by clustering the high–dimensional texture feature vectors using supervised and unsupervised learning techniques. The training set required by the supervised learning stage is manually chosen with the help of domain experts. The construction of visual thesaurus is illustrated in 6.1.

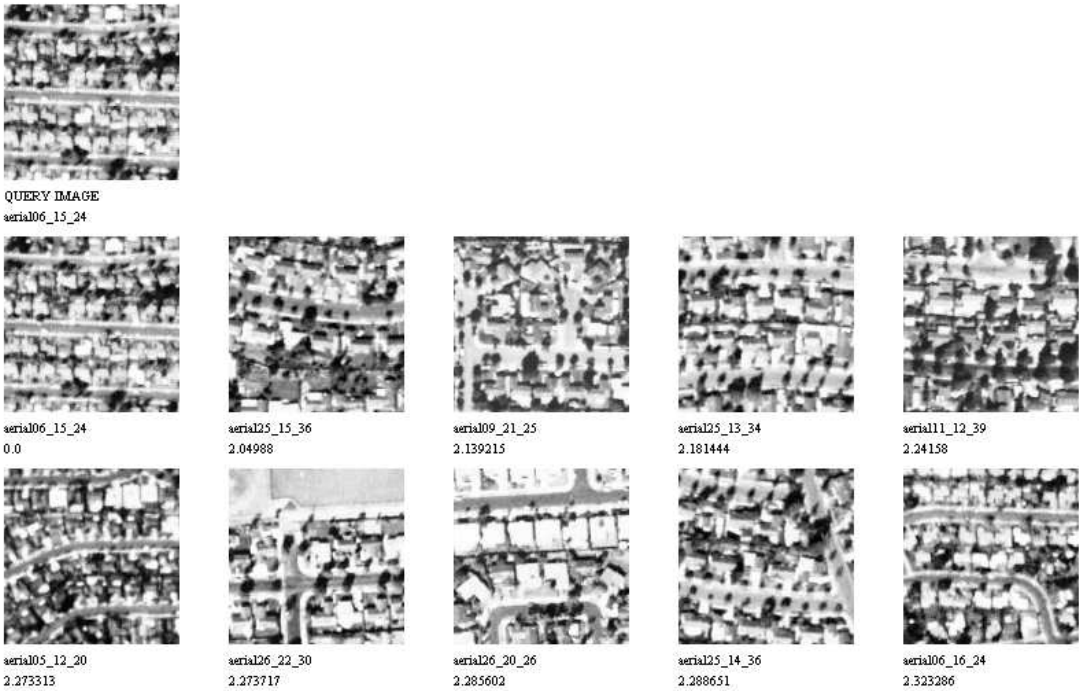



Figure 6.2: Effectiveness of the MPEG-7 HTD over an aerial image dataset for nearest neighbor search.

### 6.2.1 Image Features

The first step in constructing a visual thesaurus is feature extraction. Feature extraction is localized by partitioning an image into tiles. Regular partitioning is a simple alternative to segmentation that allows straight-forward feature extraction and provides a simple spatial layout. After partitioning, an MPEG-7 [78] compliant 60-dimensional HTD is extracted for each tile. Other features can be similarly extracted from the tiles. The texture feature vector is composed as described in 3.2, and the similarity is measured using Euclidean distance.

The MPEG-7 HTD effectively captures visual similarity, as illustrated Figure 6.2.

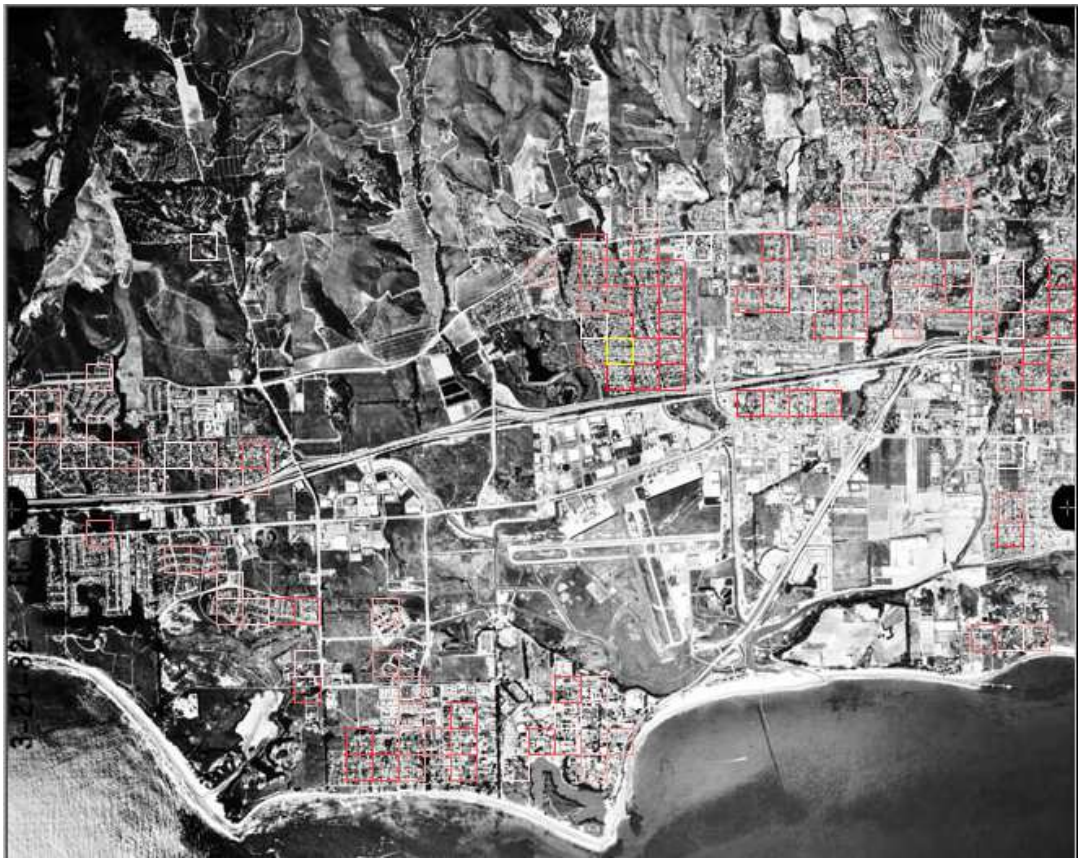


Figure 6.3: Effectiveness of the MPEG-7 HTD over an aerial image dataset for range search over similar tiles in one image.

Given a texture feature vector of a tile as a query, the nearest neighbor search in the texture space retrieves visually similar tiles. Moreover, Figure 6.3 illustrates that range search over an image that contains query tile (yellow square) retrieves all the similar regions within that image.

### 6.2.2 Feature Classification

The second step in constructing a visual thesaurus is feature classification. Conceptually, visually similar tiles are assigned the same class label by partitioning the high–dimensional feature space using a combination of supervised and unsupervised learning techniques.

Learning Vector Quantization (LVQ) [117] is a supervised learning algorithm used in the classification process. An LVQ network consists of an array of labeled *weight vectors*, (usually called the winning vector or neuron), where each label corresponds to a class. When training, each labeled vector in the training sample is submitted to the learning network. The closest weight vector to the input sample is computed, and updated in such a way that it gets closer to the input vector if both belong to the same class, and is moved away if they belong to different classes. LVQ supervised learning is efficient if the initial weight vectors are close to their final values. In a typical CBIR scenario, feature vectors are high–dimensional, and the training set is very small compared to the size of the database. Initializing LVQ has proved to be a challenging task in this scenario since it is difficult to apply in high–dimensional feature space due to "dimensionality curse" as described in Section 2.4. Kohonen proposed the use of Self Organizing Maps (SOM) for vector initialization [69] in this scenario. A set of training tiles is used to configure SOMs. This approach was shown to be effective on large sets of texture features in [74]. A SOM converts complex, nonlinear statistical relationships between high–dimensional data items into simple geometric relationships on a low-

dimensional display, while preserving the topological layout of the feature space [117]. The output nodes of the SOM are labeled using the training set and a majority-vote principle [18]. The labels are first manually assigned to a training set so that adjacent class numbers correspond to visually similar classes. The resulting clusters are assigned class labels using a majority-vote rule and the SOM result is used to initialize an LVQ algorithm. The supervised learning stage of the feature classification is summarized in the Algorithm 3.

**Algorithm 3** Feature Classification

SOM summarizes input training feature space;

label SOM output using training set;

$t = 1$.

**while** $(t \leq T)$ **do**

Fine–tune class boundaries using LVQ;

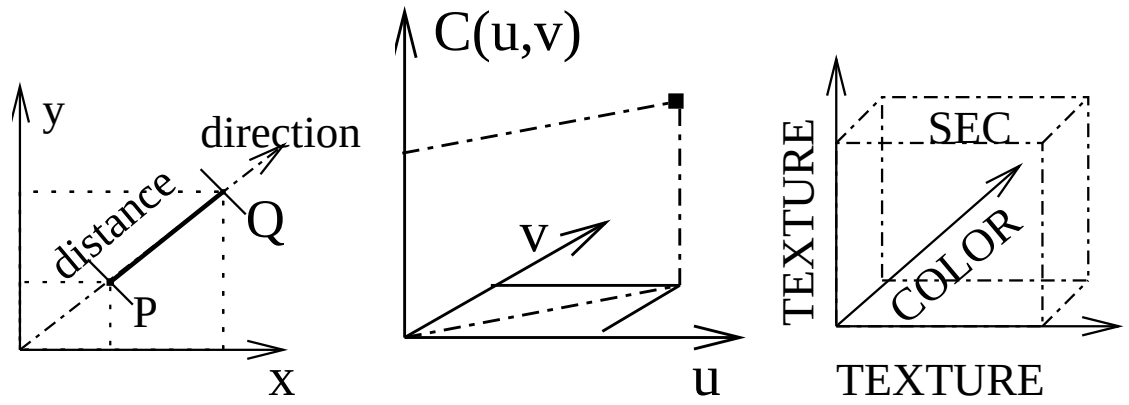
Re-assign labels using majority-vote approach;

$t = t + 1$

**end while**

### 6.2.3 Thesaurus Entries

High–dimensional feature spaces are usually very sparse as described in Section 2.4. Feature classification enforces space partitioning that frequently results in visually dissimilar features belonging to the same class. Therefore, data partitioning via the Generalized Lloyd Algorithm [50] is used to further split the classes into more con-




(a) $P\rho Q$ (b) $C_\rho(u, v)$ (c) Multimodal SEC

Figure 6.4: (a) Illustration of binary relation $\rho$, (b) construction of a Spatial Event Cube entry, and (c) Example of Multimodal Spatial Event Cube

sistent clusters. The number of clusters within each class should be proportional to the number of items within that class, enforcing similar cluster size. A representative codeword is selected for each cluster to form the visual thesaurus entry. The remaining cluster features are synonyms of the codeword and receive the same codeword label. This demonstrates a key feature of the visual thesaurus: the final codeword labeling represents a finer and, therefore, more consistent partitioning of the high–dimension feature space than the manually chosen training set.

## 6.3 Spatial Event Cubes

The motivation for building a spatial event data structure is to discover interesting spatial patterns in extended image datasets. Towards this end, we introduce Spatial Event Cube (SEC), a novel data representation obtained by applying a spatial pred-

icates to image features labeled using the visual thesaurus. The visual thesaurus is used to label the image regions based solely on their distribution in the feature space. Knowledge of the spatial arrangement of the regions is incorporated through SEC, a scalable data structure that tabulates the region pairs that satisfy a given binary spatial predicate [123].

Spatial Event Cubes are a scalable approach to mining spatial events in large image datasets based on the spatial co-occurrence of perceptually classified image features. We define the image raster space $\mathbb{R}$, for an image partitioned into $M \times N$ tiles, as

$$\mathbb{R} = \{(x, y) |\ x \in [1, M],\ y \in [1, N]\}.$$

Spatial relationships between coordinates in an image can be defined as a binary predicate $\rho$, $\rho : \mathbb{R} \times \mathbb{R} \rightarrow \{0, 1\}$, or $P \rho Q \in \{0, 1\}$, where $P, Q \in \mathbb{R}$. If we define $\rho$ as: $P \rho Q = 1$ only if distance between $P$ and $Q$ is $d$ in a direction $\phi$ from the point $P$, figure 6.4(a) shows an example of such a binary relation $\rho$.

Let the set $T$ of thesaurus entries $u_i$ be $T = \{u_i | u_i \text{ is a codeword}\}$. Let $\tau$ be a function that maps image coordinates to thesaurus entries, $\tau(P) = u$, where $P \in \mathbb{R}$ and $u \in T$; Then, a face of a Spatial Event Cube is the co-occurrence matrix $C_\rho(u, v)$ of thesaurus entries $(u, v) \in T$ of all points whose spatial relationship satisfies $\rho$:

$$C_\rho(u, v) = \|(P, Q) |\ (P \rho Q) \wedge (\tau(P) = u). \wedge (\tau(Q) = v)\|$$

Figure 6.4(b) shows the structure of SEC. The co-occurrences are computed over all the images in the dataset. Note that it is the relation $\rho$ that determines the particular spatial arrangement tabulated by the SEC. The choice of $\rho$ is application dependent

and can include spatial relationships such as adjacency, orientation, and distance, or combinations thereof. $C_\rho(u, v)$ is the number of tiles with thesaurus entries $u$ and $v$ that satisfy spatial relationship $\rho$. A multi-modal SEC structure is a hypercube whose dimensions are defined by image features extracted from the image tiles. A three-dimensional example, with two texture axes and one color axis, is shown in Figure 6.4(c).

## 6.4 Association Rules

Association rule approach was first introduced in [3] as a way of discovering interesting patterns in large transactional databases. Transactional databases consists of transactions, where each transaction consists of a set of items, for example, database of check-out cash registers at a large supermarket store. Each customer purchase is logged in a database as a transaction, and consists of the items purchased by that customer (i.e. groceries, magazines, cleaning products). Each record in the database may consist of one of dozens different items that were purchased, arranged in an order that they were scanned at the cash register. In such a framework the problem is to discover all associations and correlations among items where the presence of one set of items in a transaction implies (with a certain degree of confidence) the presence of other items. The objectives for applying association rule algorithms to this traditional transaction databases is to derive association rules that identify the items and co-occurrences of different items that appear with the greatest frequencies, i.e. a primary objective of

market basket analysis is to determine optimal product placement on store shelves.

However, the objectives for mining association rules in multimedia datasets are less obvious at this early research stage in perceptual data mining when the limits of what is technically feasible are unknown. Ideally, association rules [3] would provide insight into the prominent trends in the dataset, such as interesting but non-obvious spatial or temporal causalities.

### 6.4.1 Apriori Algorithm

This section provides a general description of association rules and outlines a widely used Apriori algorithm [3] to discover them.

Let $U = \{u_1, ...u_N\}$ be a set of items. A set $A$ is a $K - itemset$, if $A \subseteq U$ and $|A| = K$. An association rule is an expression $A \Rightarrow B$, where $A$ and $B$ are itemsets that satisfy $A \cap B = .$ Lets $D$ be a database of all $T$, i.e. $D = \{T|T \subseteq U\}$. Elements of database $D$ are called *transactions*. Transaction $T \subseteq D$ supports an itemset $A$ if $A \subseteq T$. Support of itemset $A$ over all database transactions $\mathbb{T}$ is defined as:

$$supp(A) = \frac{|\{T \in D|A \subseteq T\}|}{|D|} \tag{6.1}$$

Apriori algorithm discovers combination of items that occur together with greater frequency than might be expected if the values or items were independent. The algorithm selects the most "interesting" rules based on their support and confidence. Rule $A \Rightarrow B$ expresses that whenever a transaction $T$ contains $A$, it probably contains $B$ also.

**Support** measures the statistical significance of a rule:

$$supp(A \Rightarrow B) = \frac{|\{T \in D | A \subseteq T \wedge B \subseteq T\}|}{|D|}. \tag{6.2}$$

**Confidence** is a measure of a strength of a rule:

$$conf(A \Rightarrow B) = \frac{|\{T \in D | A \subseteq T \wedge B \subseteq T\}|}{|\{T \in D | A \subseteq T\}|}. \tag{6.3}$$

The rule confidence probability is defined as the conditional probability $P(B \subseteq T | A \subseteq T)$. Note that association rules can be between more than 2 items, e.g., $(A, B) \Rightarrow C$ where $A,\ B,\ C \subseteq U$. An association rule is strong if its confidence is larger than the user's specified minimum support. Several improvements have been proposed for mining association rules [55, 54, 143]. They deal with complexity of rule mining and separating interesting rules from the generated rule set in a more efficient way.

To perform a search, the user has to specify the minimum support for frequent itemsets. *Every subset of a frequent itemset is also frequent.* Each superset of frequent itemsets and cardinality $K$ belongs to a set of candidate itemsets of size $K$, i.e. $C_K$. The Apriori algorithm that identifies the frequent itemsets used to generate strong association rules is given in Algorithm 4.

**Algorithm 4** Apriori Algorithm

**1. Find frequent item sets**;

$F_1 = \{u_i | \ \|u_i\| > \text{minimum support}\}$

**for** $(K = 2;\ F_{K-1} \neq \emptyset;\ K++)$ **do**

$C_K = \{c_k | |c_k| = K, c_k \text{ is a combination of frequent sets from } F_{K-1}\}$

**for** $(\forall T \subseteq D)$ and $(\forall c_k \in C_K)$ **do**

**if** $(c_k \in T)$ **then**

$\|c_k\| = \|c_k\| + 1;$

**end if**

**end for**

$F_K = \{c_k | \|c_k\| > \text{minimum support }\}$

**end for**

$F = \bigcup_K F_K$

**2. Use the frequent itemsets to generate strong association rules**.

## 6.5 Perceptual Mining

A strong motivation for the research presented in this chapter is to investigate the perceptual association in an image dataset. This, in turn, will allow data mining practitioners to work with domain experts in identifying objectives that are both interesting and feasible. A spatial association rule derived from remote sensed imagery might help discover what kind of land types co-occur frequently in the vicinity of each other over large regions.

Several approaches to applying association rules to image datasets have been proposed. In [92], system explores the co-occurrence of image regions that have been labeled as similar from the Blobworld system [27] using an empirically determined distance measure and threshold. The segmented regions are viewed as items and the images are viewed as transactions so that the resulting rules were of the form, "The presence of regions A and B imply the presence of region C with support X and confidence Y". It is not clear, however, that their results from applying the technique to a dataset of synthetic images composed of basic colored shapes would generalize to real images for which segmentation and notions of region similarity present a significant challenge.

Ding et al. [41] extracted association rules from remote sensed imagery by considering set ranges of the spectral bands as items and the pixels as transactions. They also used auxiliary information at each pixel location, such as crop yield, to derive association rules of the form "Band 1 in the range $[a, b]$ and band 2 in the range $[c, d]$

results in crop yield Z with support X and confidence Y." However, such analysis at the pixel scale is susceptible to noise, unlikely to scale with dataset size, and limited in its ability to discover anything other than unrealistically localized associations, i.e., in reality, what occurs at one pixel location is unlikely to be independent of nearby locations.

Thesaurus entries and their spatial relationships define a non-traditional space for data mining applications. This space can be used to discover interesting rules such as the spatial co-occurrence of orchard and housing regions in aerial images. SECs, described in Section 6.3 allow us to extend the traditional association rule approach to multimedia databases. An association rule [3] of the form $A \Rightarrow B$ is expressing the likelihood that the presence of itemset $A$ implies the presence of itemset $B$. An association rule algorithm discovers the rules that have support and confidence larger than a specified threshold. The bottom-up approach we propose here transforms the raw image data into a form suitable for such analysis in three steps. First, image regions are labeled as perceptual synonyms using a visual thesaurus that is constructed by applying supervised and unsupervised machine-learning techniques to low-level image features. The region labels are analogous to items in transaction databases. Second, the first- and second-order associations among regions with respect to a particular spatial predicate are tabulated using SEC. Finally, higher-order rules are determined using an adaptation of the Apriori association rule algorithm, Perceptual Association Rule Algorithm. These modular steps can be individually tailored, making the framework applicable to a variety of problems and domains.

### 6.5.1 Outline of the perceptual Association Rule Algorithm

The perceptual Association Rule Algorithm supports generalized Apriori algorithm approach to candidate itemset generation in multimedia datasets outlined in [39]. Hand, Mannila and Smyth formulated Apriori algorithm in terms of more abstract notions, suitable for multimedia. Through its use, we test for occurrences of interesting patterns in a dataset, thus avoiding the formulation of transactions.

An attribute value set $T$ contains $N$ thesaurus entries $u_i$. Representative dataset $D$ is a image dataset. Therefore, the first order itemset for thesaurus entry is

$$F_1 = \{u_i|\ \|u_i\| > S_\rho^{(1)}\}. \tag{6.4}$$

For higher order itemsets, we define transaction in terms of whether a certain pattern, i.e. spatial relation, is true about a thesaurus entry. The SEC face entries $C_\rho(u, v)$ mark the frequency of codeword tuples that satisfy binary relation $\rho$. For $F_2$ we build the conjunction of two thesaurus entries $(u_i, u_j)$ and compute the corresponding SEC entries. If:

$$C_\rho(u_i, u_j) > S_\rho^{(2)}, \tag{6.5}$$

then $(u_i, u_j) \in F_2$. Define $F_K^\rho$ as a set of frequent itemsets of size $K$. Multiple entry itemsets, i.e those with $K > 2$, will reduce to ones of smaller order, with different entries. Define $S_\rho^{(K)}$ as a minimum support value for item $(u_1, u_2, ..u_K) \in F_K^\rho$. Our goal is to find $F^\rho = \bigcup_K F_K^\rho$, i.e., sets of thesaurus entries that show some dependency among tile spatial configurations.

SECs are built on the binary relationship $\rho$. With higher order candidate sets, spatial ordering of candidates formulates a transaction. Then, we go back to the representative dataset $D$ and record the occurrences of new candidates that satisfy the spatial ordering. Only the itemsets with support larger that the user specified minimum $S_\rho^{(K)}$ qualify for frequent itemset of size $K$. Note that the processing rule can differ for different itemset sizes.

For higher cardinality itemsets, there are more ways to spatially organize their elements. Association rules can be between 3 items of the form $(u_i, u_j) \Rightarrow u_k$, where $(u_i, u_j, u_k) \subseteq D$. If $C_\rho(u_i, u_j, u_k)/C_\rho(u_i, u_j)$ is larger than the minimum confidence required, the rule $(u_i, u_j) \Rightarrow u_k$ is a valid rule. For the "right neighbor" example, the rule could be formulated as "If codeword $u_j$ is the 'right neighbor' of codeword $u_i$, that might imply that $u_k$ is on the right side of $u_j$". The extended association rule algorithm for finding frequent itemsets for spatial relationship $\rho$ is given by Algorithm 5.

**Algorithm 5** Perceptual Association Rule

**1. Find frequent item sets**

$F_1^\rho = \{u_i|\ \|u_i\| > S_\rho^{(1)}\}$;

$F_2^\rho = \{(u_i, u_j)|C_\rho(u_i, u_j) > S_\rho^{(2)}\}$;

**for** $(K = 3;\ F_{K-1} \neq \emptyset;\ K++)$ **do**

Candidate $K$-item frequent itemset $C_K$ is formed of $K$

joint elements from any frequent $F_{K-1}^\rho$ item set|

$C_K = \{c_k|c^{(a)}, c^{(b)} \in F_{K-1}\}$, where:

$c_k = (u_{i_1}, ..., u_{i_{k-2}}, u_{i_{k-1}}, u_{i_k})$

$c^{(a)} = (u_{i_1}, ..., u_{i_{k-2}}, u_{i_{k-1}})$

$c^{(b)} = (u_{i_1}, ..., u_{i_{k-2}}, u_{i_k})$

**for** $(\forall c_k \in C_K)$ and $(\forall \sigma_i \in R^K)$ **do**

$\|c_k\| = \|\{\sigma_i|\sigma_i$ satisfies spatial relationship$\}\|$

AND $(\tau(P_1) = u_1) \wedge ... \wedge (\tau(P_K) = u_K)$,

where $\sigma_i = (P_1, ..., P_K) \in R^K$

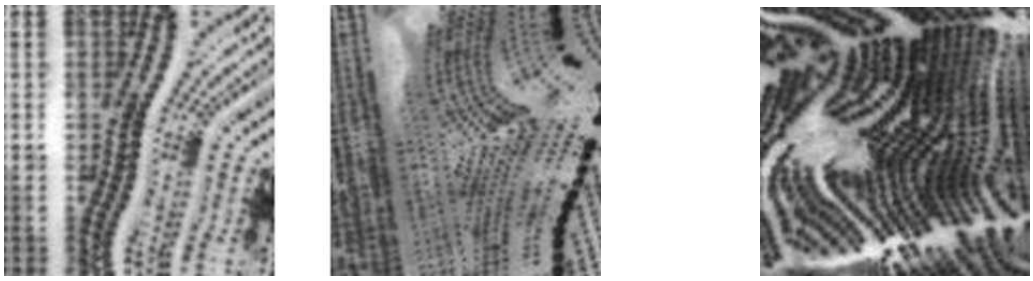
**end for**

$F_K = \{c_k|\|c_k\| > S_\rho^{(K)}\}$

**end for**

$F^\rho = \bigcup_K F_K^\rho$

**2. Use the frequent itemsets to generate association rules.**;



(a) class 16 (b) class 17

Figure 6.5: Three training tiles from two different agricultural classes of the aerial image training set of case study I.

## 6.6 Case Study I: Aerial Image Dataset

The proposed visual mining framework is applied to a dataset of 54 large aerial images of the Santa Barbara County region. The MPEG-7 homogeneous texture descriptor has been effective for characterizing a variety of land-cover types from this dataset[88]. For the study, each $5248 \times 5248$ pixel image is divided into $128 \times 128$ pixel non-overlapping tiles, resulting in a dataset of 90,744 tiles. A 60-dimension texture feature vector is then extracted for each tile, as described in Section 3.2.

A visual thesaurus of the tiles is constructed, as described in Section 6.2. The set of manually labeled tiles is used to train the supervised learning stage of the classification algorithm (Section 6.2.2). An example of manual data labelling is shown in Figure 6.5. This training set contains 60 land-cover classes, such as agricultural fields, water, parking lots, etc. The 60 classes are further partitioned into 308 codewords using the data clustering techniques described in Section 6.2.3. These codewords form

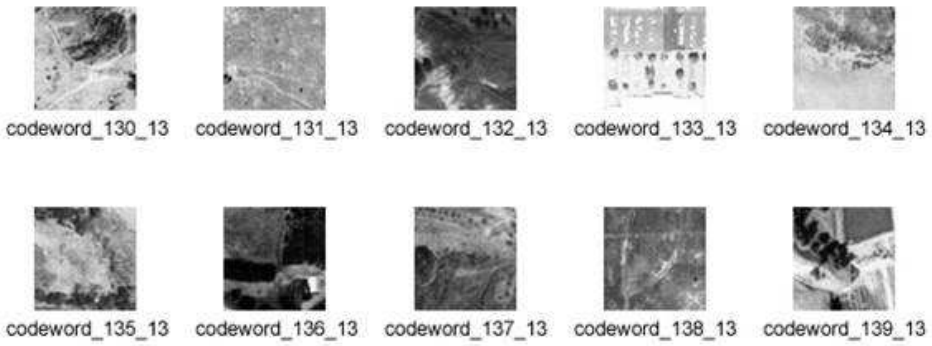
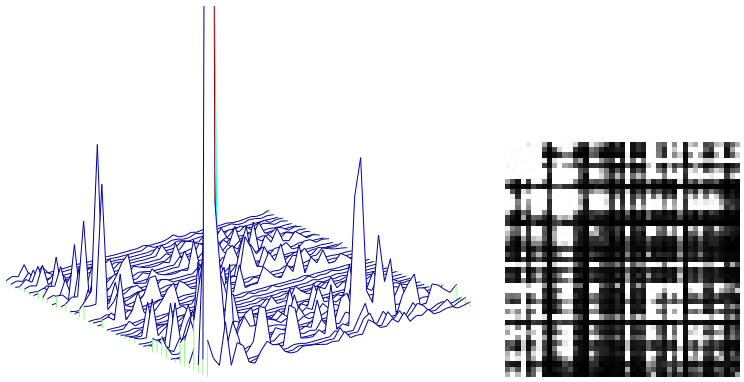



Figure 6.6: Thesaurus entries corresponding to the training class 13 (of the 60 training classes).

the thesaurus entries. Every tile in the dataset is labeled with one of these codewords. The thesaurus entries that correspond to codewords representing clusters in training class 13 are shown in Figure 6.6.

Spatial Event Cubes are constructed using tile adjacency as the spatial relation. Adjacency is defined as the 8-connectivity neighborhood. The dominant spatial arrangements of the labeled image tiles over the entire dataset are readily observable from the SEC faces or cross-sections. An SEC faceplate subspace can be visualized as a three-dimensional graph or a two-dimensional image, as shown in Figures 6.7. In Figure 6.7(a), the $x$ and $y$ axes of the graph correspond to the codewords and the $z$ axis indicates the relative co-occurrence of two codewords with respect to the spatial relation. When an SEC faceplate is viewed as an image, the co-occurrence value corresponds to the image intensity.

Figure 6.7 shows a faceplate of the SEC for the 60 classes in the aerial image dataset using adjacency as the spatial relation. We expect large homogeneous regions in the



(a) 3D (b) 2D

Figure 6.7: (a) 3D, and (b) 2D visualizations of the SECs constructed based on the 8 nearest neighbor rule for the aerial image dataset of case study I.

**22** **32** **26** **35** **41**

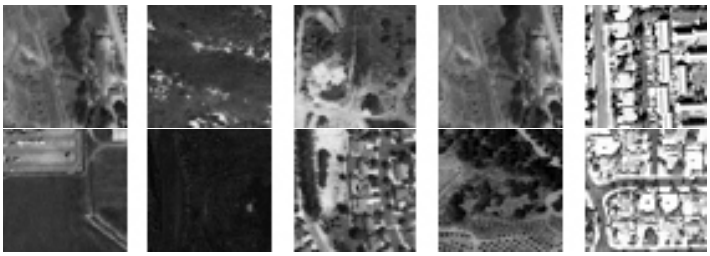
Figure 6.8: Codeword tiles corresponding to the most frequent elements in the first-order item set $F_1^{\rho}$.

dataset to result in large values along the diagonal of the faceplate. The spike in Figure 6.7 corresponds to the entries representing ocean-water tiles. This results from the fact that the aerial images of Santa Barbara County contain large regions of the Pacific Ocean. The SEC visualization enables fast homogeneous region analysis in an image dataset.

The most frequent first- and second- order codeword itemsets for the aerial image dataset are presented in Tables 6.1 and 6.2, respectively. The itemsets are computed

| $i$ | 22 | 32 | 26 | 35 | 41 |
|---|---|---|---|---|---|
| $C_\rho(u_i, u_i)$ | 24298 | 20970 | 18030 | 8368 | 7133 |

Table 6.1: Codeword elements of the first-order item set $F_1^\rho$ and their corresponding frequencies.



**(26,2)** **(32,11)** **(22,8)** **(26,46)** **(332,315)**

Figure 6.9: Codeword tiles corresponding to the most frequent elements in the second-order item set $F_2^\rho$.

using the 308 codewords of the visual thesaurus and adjacency as the spatial relation. The most frequent elements of the first-order itemset $F_1^\rho$ correspond to homogeneous regions. Figure 6.8 shows the corresponding visual thesaurus codewords, namely pasture and ocean tiles.

Higher order itemsets provide information about relations between tuples of codewords. Figure 6.9 shows the visual thesaurus codewords for the most frequent elements of the second order itemset $F_2^\rho$. Figure 6.10 shows a combination of the the most frequent tuples and triples resulting from perceptual association rules. Ocean and pasture tiles exhibit strong composite spatial arrangements.

| $(i, j)$ | 26,2 | 32,11 | 22,8 | 26,46 | 332,315 |
|---|---|---|---|---|---|
| $C_\rho(u_i, u_j)$ | 855 | 672 | 633 | 552 | 445 |

Table 6.2: Codeword elements of the second-order item set $F_2^\rho$ and their corresponding frequencies.

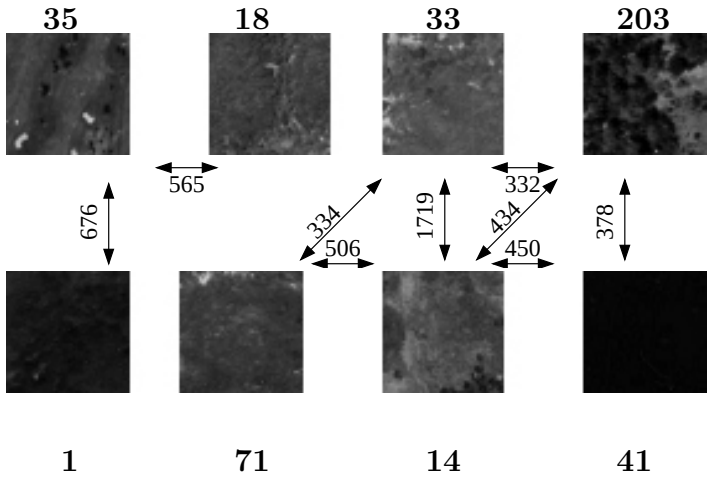



Figure 6.10: Composite spatial arrangement of ocean and pasture tiles in an aerial dataset.

## 6.7 Case Study II: Amazonia Video Dataset

The proposed technique is next applied to a collection of aerial videos of Amazonia made available by the Institute for Computational Earth System Science (ICESS) at UCSB. Aerial videography is an affordable alternative to expensive high–resolution aerial and satellite imagery. It is particularly attractive for areas plagued by cloud cover, such as the Amazon, since the videography aircraft are flown at low altitudes. The sample dataset was captured using a high-end commercial video camera and is geo-referenced. The aerial videos are temporally sub-sampled to create a sequence of just-overlapping frames, that are treated as a collection of images. One hour of

Figure 6.11: Extracted frame No. 238 from the Amazonia videography dataset.

video results in approximately 450 frames of size $720 \times 480$ pixels. Figure 6.11 shows a sample frame. The results presented here are for one hour of the 40-hour dataset. The proposed method can alternatively be applied to image mosaics created from the video, although care must be taken that the mosaicking process does not introduce unwanted artifacts.

| **Class** | **Description** | **Training Set Size** |
|---|---|---|
| 0 | Pasture | 285 |
| 1 | Forest | 238 |
| 2 | Agricultural | 116 |
| 3 | Road | 185 |
| 4 | Urban | 116 |

Table 6.3: Training set for 5 class manual labelling, with a total of 940 training tiles.

The primary land cover types are identified as pasture, forest, agriculture, road, and urban. Water is not considered separately since it does not occur often in a homogeneous tile. Roads occur frequently and are distinct so the final basic land cover types are pasture, forest, agricultural, road, and urban. Table 6.3 lists the size of the five classes in the training set. Note that the training set measures less than three percent of the entire dataset.

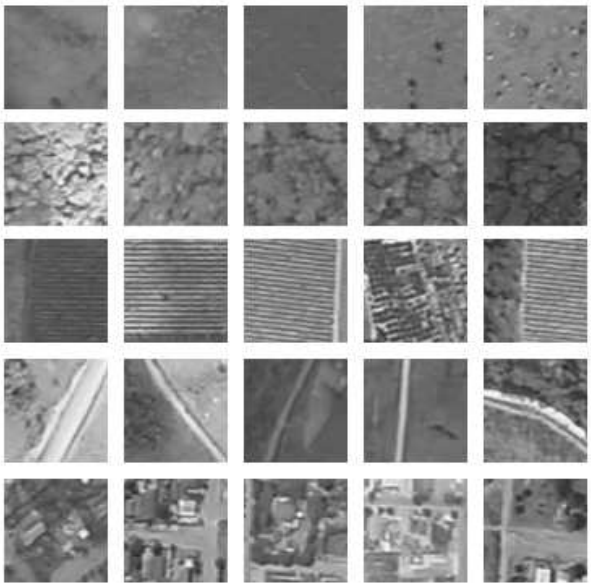
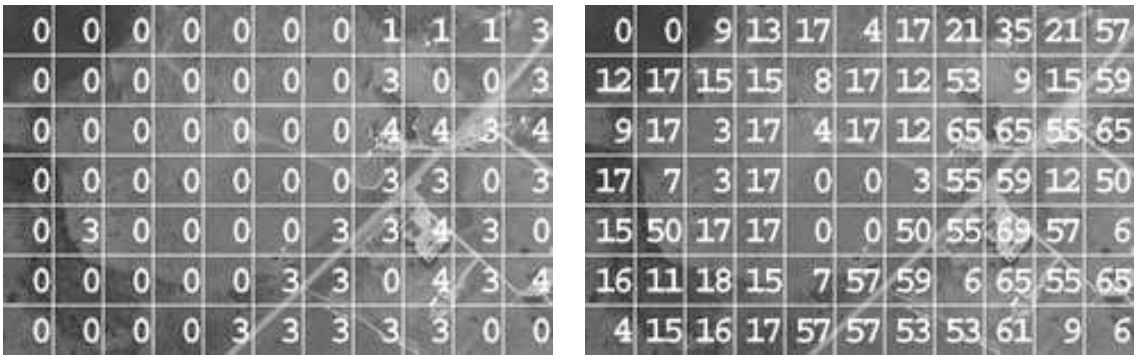


Figure 6.12: Training set samples, from top to bottom by row: pasture, forest, agriculture, road, and urban.

Since the texture descriptor captures the spatial distribution of relative pixel values, it is less sensitive to changes in lighting conditions than spectral descriptors. This is a significant advantage for the analyzing of the aerial videos of Amazonia, which contain a mixture of sunny and cloud-shaded regions. The texture descriptors are extracted in an automated fashion by dividing the 450, $720 \times 480$ pixel video frames into non-overlapping $64 \times 64$ pixel tiles and applying Gabor filters tuned to combinations of six orientations and five scales, Section 3.2. The first- and second-order moments of the filter outputs form the texture feature vector. Thus, one hour of video results in approximately 35,000 tiles each characterized by a 60-dimension vector. The visual similarity between tiles is computed by defining a distance function on the high–dimensional feature space to be Euclidean.




(a) Class Labels (b) Codeword Labels

Figure 6.13: (a) Frame No. 238 labeled with the 5 classes used in training, and (b) Frame No. 238 labeled with the 73 codewords from the Amazonia thesaurus.

The resulting codewords can be considered as fine-tuned subclasses of the manually chosen training classes. Figures 6.13(a) and 6.13(b) show the results of using the visual thesaurus to label the frame in Figure 6.11. Figure 6.13(a) shows the class assignments, which are mostly correct. Figure 6.13(b) shows the codeword labels, which are subclasses of the training set labels. For example, codeword labels $0 - 19$ correspond to subsets of the pasture class, and labels $49-63$ correspond to subsets of the road class. An SEC is computed from the codeword labelled frames using 8-neighbor adjacency as the spatial predicate. The diagonal entries of the SEC indicate the number of times a codeword appears next to itself. Thus, the largest diagonal values correspond to the homogeneous regions of the dataset. Table 6.4 lists the diagonal entries with the largest values. These entries correspond to pasture and forest codewords.

The support (6.2) and confidence (6.3) of the rule $u_i \Rightarrow u_j$ constructed for 8-connectivity neighborhood can be derived from the SEC entries:

| $i$ | $\|u_i\|$ | $C_\rho(u_i, u_i)$ | $C_\rho(u_i, u_i)/\|u_i\|$ |
|---|---|---|---|
| 21 | 3621 | 2366 | 0.653411 |
| 0 | 2555 | 2273 | 0.889628 |
| 35 | 2330 | 1608 | 0.690129 |
| 12 | 2903 | 1566 | 0.539442 |
| 17 | 2081 | 1054 | 0.506487 |
| 33 | 1183 | 796 | 0.672866 |
| 24 | 884 | 508 | 0.575226 |
| 41 | 943 | 508 | 0.538706 |
| 4 | 1289 | 492 | 0.382079 |
| 38 | 728 | 450 | 0.618819 |

Table 6.4: Corresponding frequencies of largest diagonal SEC entries for 8-connectivity neighborhood rule.

$$supp(u_i \Rightarrow u_j) = \frac{C_\rho(u_i,u_j)}{|D|} \text{ and } conf(u_i \Rightarrow u_j) = \frac{C_\rho(u_i,u_j)}{2*|u_i|},$$

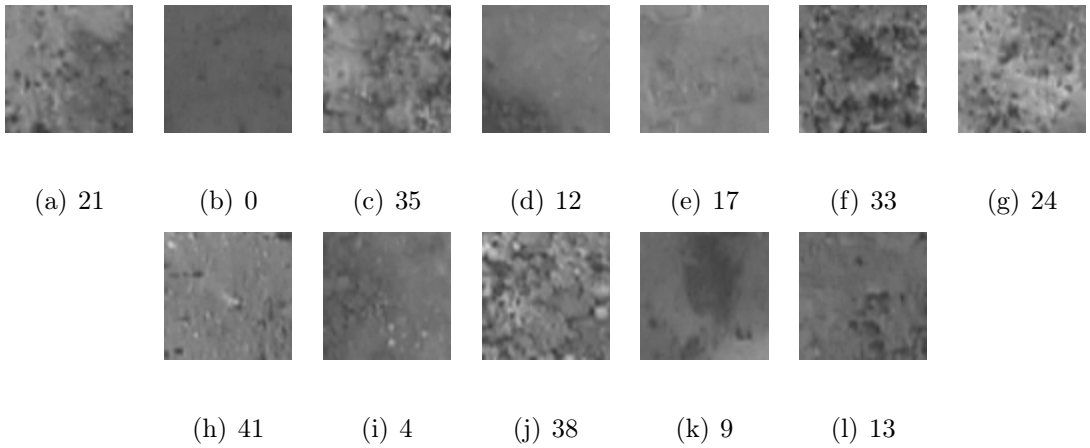
where the factor of 2 in the confidence denominator is needed since $\rho$ is symmetric in this case.

Constructed associated rules from $F_2^\rho$ are listed in tables 6.4 and 6.5, respectively. The corresponding confidence values are tabulated. Figure 6.14 shows the corresponding codewords to the most frequent elements of the second order item set $F_2^\rho$. Codewords in the range $0-19$ correspond to pasture subclasses and codewords in the range $20-39$ correspond to forest subclasses.

A second SEC is computed next using a directional spatial adjacency predicate $\rho$:

$$(x_1, y_1)\rho(x_2, y_2) \Leftrightarrow (x_1 = x_2) \wedge (y_1 + 1 = y_2),$$

where $x_i$ and $y_i$ are the horizontal and vertical coordinates of tile $i$. This predicate allows spatial analysis along the direction of the flight for the aerial video. Table 6.6



(a) 21 (b) 0 (c) 35 (d) 12 (e) 17 (f) 33 (g) 24

(h) 41 (i) 4 (j) 38 (k) 9 (l) 13

Figure 6.14: Amazonia thesaurus entries for the most frequent elements of $F_1^\rho$, and $F_2^\rho$.

lists the diagonal entries with the largest values. These entries again correspond to pasture and forest codewords.

While it is no surprise that forest and pasture are the most frequently occurring land types, Table 6.7 indicates that specifically forest codeword 21 and pasture codeword 0 occur most frequently. And the righthand neighbor rule discovers that the pasture codeword 13 is more likely to be a righthand neighbor of forest codeword 21 than vice versa.

For the second SEC, associated rules constructed from $F_2^\rho$ are listed in Tables 6.6 and 6.7, respectively. Note that the support and the confidence of the rule $u_i \Rightarrow u_j$ constructed for the "right-hand neighborhood rule" is:

$$supp(u_i \Rightarrow u_j) = \frac{C_\rho(u_i,u_j)}{|D|} \text{ and } conf(u_i \Rightarrow u_j) = \frac{C_\rho(u_i,u_j)}{|u_i|}.$$

Since the results from Table 6.7 and Table 6.8 are almost identical to the ones for 8-

| $i$ | $j$ | $C_\rho(u_i, u_j)$ | conf$(u_i \Rightarrow u_j)$ | conf$(u_j \Rightarrow u_i)$ |
|---|---|---|---|---|
| 12 | 0 | 2461 | 0.423872 | 0.481605 |
| 17 | 12 | 1764 | 0.423835 | 0.303824 |
| 35 | 33 | 1095 | 0.234979 | 0.462806 |
| 17 | 4 | 939 | 0.225613 | 0.364236 |
| 12 | 9 | 876 | 0.150878 | 0.344611 |
| 35 | 24 | 781 | 0.167597 | 0.441742 |
| 38 | 24 | 601 | 0.412775 | 0.339932 |
| 21 | 13 | 571 | 0.078846 | 0.402113 |

Table 6.5: Confidence of generated rules from $F_2^\rho$ set for 8-connectivity neighborhood rule.

neighborhood adjacency, it can be concluded that the dataset is isotropic with respect to adjacency.

The mining rule for the third order itemset is formulated as "If codeword $u_j$ is the right-hand neighbor of codeword $u_i$, $u_k$ is on the right side of $u_j$ with confidence conf$((u_i, u_j) \Rightarrow u_k)$". The candidates in $C_3^\rho$ were constrained to form a "right-hand neighbor" chain. The association rules constructed from the $F_3^\rho$ are listed in Table 6.8.

## 6.8 Discussion

This chapter introduced a novel approach to spatial event representation for large image datasets. Image features are first classified using a combination of supervised and unsupervised learning techniques. Spatial relationships between the labelled features are summarized using SECs, which are shown to be effective for visualizing non-obvious dataset spatial characteristics such as frequently occurring land cover arrangements in

| $i$ | $\|u_i\|$ | $C_\rho(u_i, u_i)$ | $C_\rho(u_i, u_i)/\|u_i\|$ |
|---|---|---|---|
| 21 | 3621 | 1159 | 0.320077 |
| 0 | 2555 | 1093 | 0.427789 |
| 35 | 2330 | 677 | 0.290558 |
| 12 | 2903 | 1760 | 0.261798 |
| 17 | 2081 | 425 | 0.204229 |
| 33 | 1183 | 329 | 0.278107 |
| 24 | 884 | 191 | 0.216063 |
| 41 | 943 | 209 | 0.221633 |
| 4 | 1289 | 225 | 0.174554 |
| 38 | 728 | 167 | 0.229396 |

Table 6.6: Corresponding Frequencies of largest diagonal SEC entries for the "right-hand neighbors" spatial rule.

aerial images. SECs are a convenient way of analyzing multi-feature relationships, and also support the extension of the association rule approach to multimedia databases for identifying frequently occurring itemsets. In particular, the *perceptual association rule algorithm*, as a novel extension of the traditional association rule algorithm, is used to distill the frequent perceptual events in large image datasets in order to discover interesting patterns. The focus is on spatial associations, although the method is equally applicable to associations within or between other dimensions, e.g. spectral or, in the case of video, temporal. The proposed approach is modular, consisting of three steps that can be individually adapted to particular applications. The proposed approach was applied to two different sets of aerial imagery to demonstrate the kinds of knowledge perceptual association rules can help discover. The two different datasets, which were large collections of aerial photos of the Santa Barbara County region and aerial videos of Amazonia, served as good demonstrations for the use and possible

| $i$ | $j$ | $C_\rho(u_i, u_j)$ | conf$(u_i \Rightarrow u_j)$ | conf$(u_j \Rightarrow u_i)$ |
|---|---|---|---|---|
| 0 | 12 | 521 | 0.203914 | 0.179470 |
| 12 | 0 | 507 | 0.174647 | 0.198434 |
| 21 | 35 | 480 | 0.132560 | 0.206009 |
| 35 | 21 | 442 | 0.189700 | 0.122066 |
| 17 | 12 | 346 | 0.166266 | 0.119187 |
| 12 | 17 | 335 | 0.115398 | 0.160980 |
| 35 | 33 | 227 | 0.097425 | 0.191885 |
| 33 | 35 | 208 | 0.175824 | 0.089270 |
| 12 | 9 | 202 | 0.069583 | 0.158930 |
| 9 | 12 | 198 | 0.155783 | 0.068205 |
| 17 | 4 | 185 | 0.088900 | 0.143522 |

Table 6.7: Confidence of generated rules from $F_2^\rho$ set, for the "righthand neighbor" rule.

| | |
|---|---|
| (12, 0, 0) | $conf((12,0) \Rightarrow 0) = 0.23077$ |
| (9, 12, 12) | $conf((9,12) \Rightarrow 12) = 0.23737$ |
| (21, 35, 35) | $conf((21,35) \Rightarrow 35) = 0.27708$ |
| (21, 17, 0) | $conf((12,17) \Rightarrow 0) = 0.095$ |

Table 6.8: The confidences of rules generated from $F_3^\rho$ set, for the "right-hand neighbor" rule.

applications of the perceptual association rules for image datasets.

# Chapter 7

# Summary and Future Work

Capturing and organizing vast volumes of multimedia data requires new information processing techniques in the context of pattern recognition and data mining. This problem has received much attention in the last decade, especially following the rapid advancements in storage technologies and corresponding growth in media data and its use. There is a strong need for new information processing technologies to handle large media data processing and understanding. The challenges dealing with these data are numerous, including in image databases such inter-related issues as high dimensionality of image feature descriptors, similarity metrics, and indexing. The unique nature of the media data makes the problem significantly more difficult and interesting with many commercial and scientific applications. The primary motivation of this dissertation work is to enable an organized, easily accessible and searchable database of a vast amount of images being generated by commercial users or in scientific applications.

## 7.1 Compression of the MPEG-7 Feature Space

The texture descriptor proved to be effective for image and video database search and retrieval. In this thesis we proposed a novel approach to dimensionality reduction and normalization of the feature vector based on the MPEG-7 descriptor origin that is cost effective and removes data redundancies. The result is a modified texture descriptor that has comparable performance, but half the dimensionality and less computational expense. Furthermore, it is easy to compute the new feature using the old one, without having to repeat the computationally expensive filtering step of the entire database. This work offers orders of magnitude improvement compared to the existing methods.

Future research directions include variable length descriptor signature file of fewer dimensions, and a similarity measure that sets a lower bound on the distance comparison along the dimensions if filter outputs follow different distributions. This, we believe, will result in a more efficient and effective perceptual similarity computation, since we do not give actual weighting to the components that are not comparable (i.e., we will not compare apples and oranges), and it will allow more efficient 2-level indexing for approximate similarity search.

p. 130, second para POTENTIAL FUTURE DIRECTIONS INCLUDE extension of spatial event cubes to include associations within or between other dimensions, i.e., spectral DIMENSIONS, or, in the case of video, temporal DIMENSIONS.

[delete the last sentence of this para]

7.6 CONCLUSIONS (I thought we decided to change just this chapter section title of Conclusions; all others to DISCUSSION.

## 7.2 Approximate Nearest Neighbor Search

Approximate nearest neighbor (ANN) searches are of particular interest in the case of large media databases where feature descriptors represent the data only approximately. ANN simplifies the computations in high dimensional feature spaces.

We have demonstrated that a search mechanism can benefit from the characteristic of multimedia feature descriptors. We have presented a novel adaptive indexing scheme to index high dimensional image/video objects using parametric estimation of data distribution. Parameters are easily derived from the data sample, and complexity of nearest neighbor search over large high dimensional datasets is reduced. We have demonstrated our approach using homogenous texture descriptor database. Rayleigh approximation and equalization of data proved to be a very good regularization approach for large MPEG-7 texture feature datasets. We proposed an adaptive indexing structure that enables efficient ANN search over large multimedia databases without significantly compromising retrieval quality. Future directions include developing the same parametric models and approximation search schemes for other MPEG-7 descriptors like edge histogram descriptor (EHD) and scalable color descriptor (SCD).

## 7.3 Relevance Feedback in Nearest Neighbor Search

Relevance feedback was introduced to facilitate interactive learning for refining the retrieval results. However, nearest neighbor computations over a large number of dataset items alone are expensive. This is further aggravated by the fact that image descriptors are in very high dimensions, as well as the need to perform this search repetitively in relevance feedback. Our contribution here is a nearest neighbor search method to considerably accelerate interactive and iterative searches. The proposed scheme exploits correlations between two consecutive nearest neighbor sets and significantly reduces the overall search complexity for general distance metric updates and compression-based data indexing.

Future work will focus on vector quantization based indexing and efficient approximate search for non-linear relevance feedback scenarios. Approximate search in those scenarios can be sped up by (1) estimating kernels in the mapped space using previously suggested estimated kernels of interest in the original space, and (2) using cluster representatives in the filtering phase.

## 7.4 Learning Perceptual Clusters in High Dimensions

Meaningful semantic analysis and knowledge extraction require data representations that are understandable at a semantic level. The framework must efficiently summarize information contained in the image data; it must provide scalability with respect to the nature, size and dimension of a dataset; and it must offer simple representations of the results. Our contribution is a summarized data information derived from the low-level features of an aerial image dataset and Amazon video key frames. To accomplish this, visually similar tiles are assigned the same class label by partitioning the high dimensional feature space. Data clustering is then used to minimize the impact of the sparsity of the high-dimensional feature space within the class. A representative codeword is selected for each cluster to form the visual thesaurus entry.

Future work includes implementing several learning mechanisms for more efficient high-dimensional feature clustering in order to develop a generative statistical models for other types of multimedia data. Another direction is further exploration of the joint clustering and compression schemes that provides effective filtering of both the irrelevant data and irrelevant information about the remaining data by imposing certain structural constraints on building the thesaurus. The idea is to use an algorithm which iteratively optimizes (for a fixed clustering map) the compression of data entries in each cluster, and then optimizes (for a fixed compression design) the clustering map.

## 7.5 Multimedia Mining

In media databases, feature descriptors fail to capture image content. Humans can instantly answer the question "Is this highway going through a desert?" just by looking at an aerial photograph of a region. This query, essentially formulated as a visual concept, raises many research issues, and the semantic analysis of multimedia content is imperative. Our contribution is introduction of a framework for summarizing basic semantic concepts in order to detect coarse spatial patterns and visual concepts in image and video aerial data. Association rules are introduced as a way of discovering interesting patterns in transactional databases. Visual thesaurus entries and their spatial relationships define a non-traditional space for data mining applications. This space can be used to discover coarse spatial relationships in a dataset. We use Spatial Event Cubes to distill the frequent visual patterns in image and video datasets. A primary contribution is the derivation of image equivalents for the traditional association rule components, namely the items, the itemsets, and the rules.

Potential future directions include extension of spatial event cubes to include associations within or between other dimensions, i.e., spectral dimensions, or, in the case of video, temporal dimensions.

## 7.6 Discussion

The proposed methods in this thesis will enable the development of an organized, easily searchable database of digital images and videos that will support complex search queries and user interactions. This methods are urgently needed both in scientific and in commercial applications dealing with large amounts of images and video data. We strongly believe this will enable us to further investigate patterns that are not perceived by human inspection, an allow us to efficiently search in data bases for relevant phenomena.

# References


[1] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional spaces. In Jan Van den Bussche and Victor Vianu, editors, *Proceedings of 8th International Conferenceon Database Theory (ICDT)*, volume 1973 of *Lecture Notes in Computer Science*, pages 420–434, London, UK, 2001. Springer.

[2] Charu C. Aggarwal and Philip S. Yu. Finding generalized projected clusters in high dimensional spaces. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 70–81, Dallas, Texas, United States, 2000. ACM Press.

[3] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, pages 487–499, Santiago de Chile, Chile, September 1994. Morgan Kaufmann.

[4] A. H. Andersen and J. E. Kirsch. Analysis of noise in phase contrast MR imaging. volume 6, pages 857Ű–869, 1996.

[5] Mihael Ankerst, Bernhard Braunmüller, Hans-Peter Kriegel, and Thomas Seidl. Improving adaptable similarity query processing by using approximations. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB)*, pages 206–217, New York City, NY, August 1998. Morgan Kaufmann.

[6] Sunil Arya and Ho-Yam Addy Fu. Expected-case complexity of approximate nearest neighbor searching. In *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 379–388. Society for Industrial and Applied Mathematics, 2000.

[7] Jeffrey R. Bach, Charles Fuller, Amarnath Gupta, Arun Hampapur, Bradley Horowitz, Rich Humphrey, Ramesh C. Jain, and Chiao-Fe Shu. The virage image search engine: an open framework for image management. In Ishwar K. Sethi and Ramesh C. Jain, editors, *Proceedings of SPIE Storage and Retrieval for Still Image and Video Databases IV*.

[8] Rudolf Bayer and Edward M. McCreight. Organization and maintenance of large ordered indices. *Acta Informatica*, 1:173–189, 1972.

[9] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. The r*-tree: an efficient and robust access method for points and rectangles. In *Proceedings of the 1990 ACM SIGMOD international conference on Management of data*, pages 322–331. ACM Press, 1990.

[10] Richard E. Bellman. *Adaptive Control Processes.* Princeton University Press, Princeton, NJ, 1961.

[11] J.L. Bentley. Multidimensional binary search trees used for associative searching. *Communication of the ACM*, 18(9):39–50, September 1975.

[12] Stefan Berchtold, Christian Böhm, Bernhard Braunmüller, Daniel A. Keim, and Hans-Peter Kriegel. Fast parallel similarity search in multimedia databases. *SIGMOD Record*, 26(2):1–12, 1997.

[13] Stefan Berchtold, Christian Böhm, H.V. Jagadish, Hans-Peter Kriegel, and Jörg Sander. Independent quantization: a index compression technique for high-dimensional data spaces. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*, pages 577–588, San Diego, CA, March 2000. IEEE Computer Society.

[14] Stefan Berchtold, Christian Böhm, Daniel A. Keim, Florian Krebs, and Hans-Peter Kriegel. On optimizing nearest neighbor queries in high-dimensional data spaces. In Jan Van den Bussche and Victor Vianu, editors, *Proceedings of 8th International Conference on Database Theory (ICDT)*, volume 1973 of *Lecture Notes in Computer Science*, pages 435–449. Springer, January 2001.

[15] Stefan Berchtold, Christian Böhm, Daniel A. Keim, and Hans-Peter Kriegel. A cost model for nearest neighbour search. In *Proceedings of 16th ACM Symposium on Principles of Database Systems (PODS)*, pages 78–86, Tucson, AZ, May 1997. ACM Press.

[16] Stefan Berchtold, Christian Böhm, and Hans-Peter Kriegel. The pyramid-tree: Breaking the curse of dimensionality. In *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 142–153. ACM Press, 1998.

[17] Stefan Berchtold, Daniel A. Keim, and Hans-Peter Kriegel. The x-tree: An index structure for high-dimensional data. In T. M. Vijayaraman, Alejandro P. Buchmann, C. Mohan, and Nandlal L. Sarda, editors, *Proceedings of 22th International Conference on Very Large Data Bases (VLDB)*, pages 28–39, Mumbai (Bombay), India, September 1996. Morgan Kaufmann.

[18] Michael Berthold and David J. Hand, editors. *Intelligent Data Analysis: An Introduction*. Springer, 1999.

[19] Kevin S. Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When Is "Nearest Neighbor" Meaningful? In *Proceeding of the 7th International Conference on Database Theory*, pages 217–235. Springer-Verlag.

[20] Sitaram Bhagavathy, Jelena Tešić, and Bangalore S. Manjunath. On the rayleigh nature of gabor filter outputs. In *Proceedings of IEEE International Conference on Image Processing (ICIP)*, Barcelona, Spain, September 2003. IEEE Computer Society.

[21] C. Böhm, H.-P. Kriegel, and T. Seidl. Adaptable Similarity Search using Vector Quantization. In *Proceedings of 3rd Conference on Data Warehousing and Knowledge Discovery (DaWak)*, pages 28–39, Munich, Germany, September 2001.

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

[23] I. Borgand and J.Lingoes. *Multidimensional similarity structure analysis*. Springer-Verlag Inc., New York, NY, 1987.

[24] Leon Bottou and Yoshua Bengio. Convergence properties of the $K$-means algorithms. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 585–592. The MIT Press, 1995.

[25] Tolga Bozkaya and Meral Ozsoyoglu. Indexing large metric spaces for similarity search queries. *ACM Transactions on Database Systems (TODS)*, 24(3):361–404, 1999.

[26] Phil Brodatz. *Textures: A Photographic Album for Artists and Designers*. Dover Pubns, August 1999.

[27] Chad Carson, Serge Belongie, Hayit Greenspan, and Jitendra Malik. Region-based image querying. In *Proceedings of CVPR Workshop on Content-Based Access of Image and Video Libraries*, San Juan, Puerto Rico, June 1997. IEEE Computer Society.

[28] Youssef Chahir and Liming Chen. Efficient content-based image retrieval based on color homogeneous objects segmentation and their spatial relationship characterization. In *Proceedings of the IEEE International Conference on Multimedia*

*Computing and Systems*, volume 2, pages 705–709, Florence, Italy, June 1999. IEEE Computer Society.

[29] Kaushik Chakrabarti and Sharad Mehrotra. The Hybrid tree: An Index Structure for High Dimensional Feature Spaces. In *Proceedings of the 15th International Conference on Data Engineering (ICDE)*, pages 440–447, Sydney, Austrialia, March 1999. IEEE Computer Society.

[30] Tianhorng Chang and C.-C. Jay Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE Transactions on Image Processing*, 2(4):429–441, October 1993.

[31] Yuan-Chi Chang, Lawrence D. Bergman, Vittorio Castelli, Chung-Sheng Li, Ming-Ling Lo, and John R. Smith. The onion technique: Indexing for linear optimization queries. In Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein, editors, *Proceedings of ACM International Conference on on Management of data (SIGMOD)*, May 2000.

[32] Paolo Ciaccia and Marco Patella. Pac nearest neighbor queries: Approximate and controlled search in high-dimensional and metric spaces. In *Proceedings of the 16th International Conference on Data Engineering (ICDE)*, pages 244–255, San Diego, CA, March 2000. IEEE Computer Society.

[33] Paolo Ciaccia, Marco Patella, and Pavel Zezula. M-tree: An efficient access method for similarity search in metric spaces. In Matthias Jarke, Michael

Carey, Klaus R. Dittrich, Fred Lochovsky, Pericles Loucopoulos, and Manfred A. Jeusfeld, editors, *Proceedings of the 23rd International Conference on Very Large Data Bases (VLDB)*, pages 426–435, Athens, Greece, August 1997. Morgan Kaufmann Publishers, Inc.

[34] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. Second edition.

[35] Ingemar J. Cox, Matt L. Miller, Thomas P. Minka, Thomas V. Papathomas, and Peter N. Yianilos. The bayesian image retrieval system, pichunter: Theory, implementation, and psychophysical experiments. 9(1):20–37.

[36] Ingemar J. Cox, Matt L. Miller, S.M. Omohundro, and Peter N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *Proceedings of 13th International Conference on Pattern Recognition (ICPR)*, volume 3, pages 361–369, August 1996.

[37] Bin Cui, Beng Chin Ooi, Jianwen Su, and Kian-Lee Tan. Contorting high dimensional data for efficient main memory knn processing. In *Proceedings of ACM International Conference on on Management of data (SIGMOD)*, June 2003.

[38] J. G. Daugman. Complete discrete 2D gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7), July 1988.

[39] Heikki Mannila David J. Hand and Padhraic Smyth. *Principles of Data Mining.* MIT Press, Cambridge, MA, August 2001.

[40] Yining Deng and B. S. Manjunath. Unsupervised segmentation of color-texture regions in images and video. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(8), August 2001.

[41] Qin Ding, Qiang Ding, and William Perrizo. Association rule mining on remotely sensed images using p-trees. In *Proceedings of 6th Pacific-Asia Conference Advances in Knowledge Discovery and Data Mining (PAKDD)*, volume 2336, pages 66–79, Taipel, Taiwan, 2002.

[42] Dennis F. Dunn and William E. Higgins. Optimal gabor filters for texture segmentation. *IEEE Transactions on Image Processing*, 4(7):947–964, July 1995.

[43] Christos Faloutsos and King-Ip Lin. Fastmap: a fast algorithm for indexing, datamining and visualization of traditional and multimedia datasets. In *Proceedings of the ACM SIGMOD international conference on Management of data*, pages 163–174, San Jose, CA, 1995. ACM Press.

[44] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining.* AAAI/MIT Press, March 1996.

[45] Hakan Ferhatosmanoglu, Ertem Tuncel, Divyakant Agrawal, and Amr El Abbadi. Vector approximation based indexing for non-uniform high-dimensional

data sets. In *Proceedings of 9th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 202–209, McLean, Virginia, United States, November 2000. ACM Press.

[46] R. A. Finkel and J. L. Bentley. Quad trees: A data structure for retrieval of composite keys. *Acta Informatica*, 4(1):1–9, 1974.

[47] Myron Flickner, Harpreet Sawhney, Wayne Niblack, Jonathan Ashley, Qian Huang, Byron Dom, Monika Gorkani, Jim Hafner, Denis Lee, Dragutin Petković, David Steele, and Peter Yanker. Query by image and video content: The qbic system. *IEEE Computer*, 28(9):23–32, September 1995.

[48] Venkatesh Ganti, Raghu Ramakrishnan, Johannes Gehrke, Allison L. Powell, and James C. French. Clustering large datasets in arbitrary metric spaces. In *Proceedings of the 15th International Conference on Data Engineering (ICDE)*, pages 502–511, Sydney, Austrialia, March 1999. IEEE Computer Society.

[49] Johannes Gehrke, Raghu Ramakrishnan, and Venkatesh Ganti. Rainforest - a framework for fast decision tree construction of large datasets. *Data Mining and Knowledge Discovery*, 4(2/3):127–162, July 2000.

[50] Alen Gersho and Robert M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, fourth edition, 1992.

[51] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Third edition, November 1996.

[52] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. CURE: An efficient clustering algorithm for large databases. In *Proceedings of ACM International Conference on Management of Data (SIGMOD)*, pages 73–84, Seattle, WA, 1998. ACM Press.

[53] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In Beatrice Yormark, editor, *Proceedings of ACM International Conference on Management of Data (SIGMOD)*, pages 47–57, Boston, Massachusetts, June 1984.

[54] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein, editors, *Proceedings of ACM International Conference on on Management of data (SIGMOD)*, May 2000.

[55] Jiawei Han, Jianyong Wang, Ying Lu, and Petre Tzvetkov. Mining top-k frequent closed patterns without minimum support. In *Proceedings of IEEE International Conference on Data Mining (ICDM)*, pages 211–218, Maebashi City, Japan, December 2002. IEEE Computer Society.

[56] R.M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. In *IEEE Transactions on Systems, Man, and Cybernetics SMC-3*, pages 610–621. IEEE Computer Society, 1973.

[57] Serkan Hatipoglu, Sanjit K. Mitra, and Nick G. Kingsbury. Image texture description using complex wavelet transform. In *Proc of IEEE International Con-*

*ference on Image Processing (ICIP)*, volume 2, pages 530–533, Vancouver, B.C., Canada, September 2000. IEEE Computer Society.

[58] Simon Haykin. *Communication Systems*. John Wiley & Sons, Cambridge, MA, 4 edition, May 2000.

[59] Alexander Hinneburg and Daniel A. Keim. Optimal Grid-Clustering: Towards Breaking the Curse of Dimensionality in High-dimensional Clustering. In Malcolm P. Atkinson, Maria E. Orlowska, Patrick Valduriez, Stanley B. Zdonik, and Michael L. Brodie, editors, *Proceedings of 25th International Conference on Very Large Data Bases VLDB*, Edinburgh, Scotland, UK, September 1999.

[60] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613, Dallas, TX, 1998. ACM Press.

[61] Yoshiharu Ishikawa, Ravishankar Subramanya, and Christos Faloutsos. Mindreader: Querying databases through multiple examples. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *Proceedings of 24rd International Conference on Very Large Data Bases (VLDB)*, pages 218–227, New York City, NY, August 1998. Morgan Kaufmann.

[62] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, New York, NY, second edition, October 2002.

[63] Roberto J. Bayardo Jr., Rakesh Agrawal, and Dimitrios Gunopulos. Constraint-based rule mining in large, dense databases. In *Proceedings of the 15th International Conference on Data Engineering (ICDE)*, pages 188–197, Sydney, Australia, March 1999. IEEE Computer Society.

[64] Norio Katayama and Shin'ichi Satoh. The sr-tree: an index structure for high-dimensional nearest neighbor queries. In *Proceedings of the ACM SIGMOD international conference on Management of data*, pages 369–380, Tucson, Arizona, United States, 1997. ACM Press.

[65] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*, volume 30. John Wiley & Sons, nineth edition, March 1990.

[66] Daniel A. Keim. Visual exploration of large data sets. *Communications of the ACM (CACM)*, 44(8):38–44, January 2001.

[67] Deok-Hwan Kim and Chin-Wan Chung. Qcluster: relevance feedback using adaptive clustering for content-based image retrieval. In *Proceedings of ACM international conference on on Management of data (SIGMOD)*, pages 599–610, San Diego, California, 2003. ACM Press.

[68] Irwin King and Jin Zhong. Integrated probability function and its application to content-based image retrieval by relevance feedback. *Pattern Recognition*, 36(9):2177–2186, September 2003.

[69] Teuvo Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, third edition, September 2001.

[70] Chen Li, Edward Chang, Hector Garcia-Molina, and Gio Widerhold. Clustering for approximate similarity search in high-dimensional spaces. volume 14, pages 792–808. IEEE Computer Society, July/August 2002.

[71] Xuequn Li and Irwin King. Gaussian mixture distance for information retrieval. In *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN)*, pages 2544–2549, Washington DC, July 1999.

[72] Fang Liu and Rosalind W. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, July 1996.

[73] Wei-Ying Ma and B. S. Manjunath. A texture thesaurus for browsing large aerial photographs. *Journal of the American Society for Information Science and Technology (JASIST)*, 49(7):633–648, September 1998.

[74] Wei-Ying Ma and Bangalore S. Manjunath. Texture features and learning similarity. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 425–430, San Francisco, CA, June 1996. IEEE Computer Society.

[75] Wei-Ying Ma and Bangalore S. Manjunath. NeTra: a toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, May 1999.

[76] B. S. Manjunath and Wei-Ying Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, August 1996.

[77] B.S. Manjunath and Rama Chellappa. Image texture. pages 694–702, 2004.

[78] B.S. Manjunath, Philippe Salembier, and Thomas Sikora, editors. *Introduction to MPEG–7: Multimedia Content Description Interface*. John Wiley & Sons Ltd., June 2002.

[79] B.S. Manjunath, Chandra Shekhar, and Rama Chellappa. A new approach to image feature detection with applications. *Pattern Recognition*, 29:627–640, 1996.

[80] Jianchang Mao and Anil K. Jain. Texture classification and segmentation using multiresolution simultaneous autoregressive models. *Pattern Recognition*, 25(2):173–188, 1992.

[81] Christophe Meilhac and Chahab Nastar. Relevance feedback and category search in image databases. In *Proceedings IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 512–517, Florence, Italy, June 1999.

[82] Thomas P. Minka and Rosalind W. Picard. Interactive learning with a "society of models". In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 447–452, San Francisco, CA, June 1996. IEEE Computer Society.

[83] Ullrich Mönich, Till Quack, Lars Thiele, and B.S. Manjunath. Cortina: Scalable content based image retrieval system, 2004.

[84] Henning Müller, Wolfgang Müller, Stéphane Marchand-Maillet, Thierry Pun, and David Squire. Strategies for positive and negative relevance feedback in image retrieval. In *Proc of 15th Intl. Conf. on Pattern Recognition (ICPR)*, volume 1, pages 5043–5042, September 2000.

[85] Henning Müller, Wolfgang Müller, and David Squire. Learning feature weights from user behavior in content-based image retrieval. In Simeon J. Simoff, Chabane Djeraba, and Osmar R. Zaïane, editors, *Proceedings of 3rd International Workshop on Multimedia Data Mining (MDM/KDD)*, pages 67–72, Edmonton, Alberta, Canada, July 2002. University of Alberta.

[86] Milind R. Naphade and Thomas Huang. Detecting semantic concepts using context and audiovisual features. In *Proceedings IEEE Workshop on Detection and Recognition of Events in Video (EVENT)*, pages 92–98, Vancouver, BC, Canada, July 2001. IEEE Computer Society.

[87] Apostol Natsev, Rajeev Rastogi, and Kyuseok Shim. Walrus: a similarity retrieval algorithm for image databases. In Alex Delis, Christos Faloutsos, and Shahram Ghandeharizadeh, editors, *Proceedings ACM SIGMOD International Conference on Management of Data (SIGMOD)*, pages 395–406, Philadephia, Pennsylvania, USA, June 1999. ACM Press.

[88] Shawn Newsam, Jelena Tešić, and B.S. Manjunath. Mpeg-7 homogeneous texture descriptor demo, 2001.

[89] Shawn Newsam, Jelena Tešić, Lei Wang, and B.S. Manjunath. Issues in Mining Video Datasets. In *SPIE Int. Symp. On Electronic Imaging, Storage and Retrieval Methods and Applications for Multimedia*, San Jose, California, January 2004.

[90] Raymond Ng and Jaiwei Han. CLARANS: A method for clustering objects for spatial data mining. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 14(5):1003–1016, February 2002.

[91] Anne H. H. Ngu, Quan Z. Sheng, Du Q. Huynh, and Ron Lei. Combining multi-visual features for efficient indexing in a large image database. *VLDB Journal*, 9(4):279–293, April 2001.

[92] Carlos Ordonez and Edward Omiecinski. Discovering association rules based on image content. In *Proceedings of the IEEE Forum on Research and Technology Advances in Digital Libraries*, Baltimore, Maryland, March 1999. IEEE Computer Society.

[93] Michael Ortega, Yong Rui, Kaushik Chakrabarti, Sharad Mehrotra, and Thomas S. Huang. Supporting similarity queries in mars. In *Proceedings of 5th ACM International Conference on Multimedia (MM)*, pages 403–413, Seattle, WA, November 1997. ACM Press.

[94] G.S. Orton, B.M. Fisher, K.H. Baines, S.T Stewart, A.J Friedson, J.L Ortiz, M. Marinova, M. Ressler, A. Dayal, W. Hoffmann, J. Hora, S. Hinkley, V. Krishnan, M. Mašanović, J. Tešić, A. Tziolas, and K.C. Parija. Characteristics of the Galileo probe entry site from Earth-based remote sensing observations. *Journal of Geophysical Research*, September 1998.

[95] Peyton Z. Peebles. *Probability, Random Variables, and Random Signal Principles*. McGraw-Hill, fourth edition, 2001.

[96] Alex Pentland, Rosalind W. Picard, and Stan Scarloff. Photobook: Tools for content-based manipulation of image databases. In *Proceedings of SPIE Storage and Retrieval for Image and Video Databases*, volume 2, pages 34–47, San Jose, CA, February 1994.

[97] Rosalind W. Picard. A society of models for video and image libraries. *IBM Systems Journal*, 35(3/4):292–312, 1996.

[98] Aparna Lakshmi Ratan, Oded Maron, W. Eric L. Grimson, and Tomás Lozano-Pérez. A framework for learning query concepts in image classification. In *Proceedings IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1423–1431, Fort Collins, CO, June 1999. IEEE Computer Society.

[99] K. V. Ravi Kanth, Divyakant Agrawal, and Ambuj Singh. Dimensionality reduction for similarity searching in dynamic databases. In *Proceedings of the 1998*

*ACM SIGMOD international conference on Management of data*, pages 166–176, Seattle, WA, 1998. ACM Press.

[100] S. O. Rice. Bell system technological journal. *Mathematical analysis of random noise*, 23(282):129–146, 1944.

[101] Yong Man Ro, Munchurl Kim, Ho Kyung Kang, B.S.Manjunath, and Jinwoong Kim. Rmpeg-7 homogeneous texture descripter. *ETRI Journal, Information , Telecommunications & Electronics,*, 23(2), June 2001.

[102] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–146, 1976.

[103] John T. Robinson. The k-d-b-tree: a search structure for large multidimensional dynamic indexes. In *Proceedings of the 1981 ACM SIGMOD international conference on Management of data*, pages 10–18, Ann Arbor, Michigan, 1981. ACM Press.

[104] Joseph J. Rocchio. Relevance feedback in information retrieval. In Gerard Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323, Englewood Cliffs, NJ, 1971. Prentice Hall.

[105] Yong Rui and Thomas Huang. Optimizing learning in image retrieval. In *Proceedings IEEE International Conference on Computer Vision and Pattern Recog-*

*nition (CVPR)*, volume 1, pages 236–243, Hilton Head, SC, June 2000. IEEE Computer Society.

[106] Yong Rui and Thomas S. Huang. A novel relevance feedback technique in image retrieval. In *Proceedings of 7th ACM International Conference on Multimedia (MM)*, pages 67–70, Orlando, FL, 1999. ACM Press.

[107] Yong Rui, Thomas S. Huang, and Sharad Mehrotra. Content-based image retrieval with relevance feedback in mars. In *Proc of IEEE International Conference on Image Processing (ICIP)*, volume 2, pages 815–818, Washington, DC, October 1997. IEEE Computer Society.

[108] Yasushi Sakurai, Masatoshi Yoshikawa, Ryoji Kataoka, and Shunsuke Uemura. Similarity search for adaptive ellipsoid queries using spatial transformation. In Peter M. G. Apers, Paolo Atzeni, Stefano Ceri, Stefano Paraboschi, Kotagiri Ramamohanarao, and Richard T. Snodgrass, editors, *Proceedings of 27th International Conference on Very Large Data Bases (VLDB)*, pages 231–240, Roma, Italy, September 2001. Morgan Kaufmann.

[109] Hanan Samet. *The design and analysis of spatial data structures*. Addison-Wesley Longman Publishing Co., Inc., August 1989.

[110] Simone Santini and Ramesh Jain. Similarity measures. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 21(9), August 1999.

[111] Shashi Shekhar and Yan Huang. Discovering spatial co-location patterns: A

summary of results. In *Proceedings of 7th International Symposium on Advances in Spatial and Temporal Databases (SSTD)*.

[112] Jan Sijbers, Arnold J. den Dekker, Paul Scheunders, and Dirk Van Dyck. Maximum likelihood estimation of rician distribution parameters. *IEEE Transactions on Medical Imaging*, 17(3):357–361, June 1998.

[113] B.W. Silverman. *Density Estimation for statistics and data analysis*. Chapman and Hall, 1 edition, January 1986.

[114] Craig Silverstein, Sergey Brin, Rajeev Motwani, and Jeffrey D. Ullman. Scalable techniques for mining causal structures. *Data Mining and Knowledge Discovery*, 4(2/3):163–192, July 2000.

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

[116] John R. Smith and Shih-Fu Chang. Visualseek: A fully automated content-based image query system. In *Proceedings of 4th ACM International Conference on Multimedia (MM)*, pages 87–98, Boston, MA, November 1996. ACM Press.

[117] Panu Somervuo and Teuvo Kohonen. Self-organizing maps and learning vector quantization for feature sequences. *Neural Processing Letters*, 10(2):151–159, October 1999.

[118] Henry Stark and John W. Woods. *Probability, random processes, and estimation theory for engineers.* Third edition.

[119] Robert Tansley. Automating the linking of content and concept. In *Proceedings of 9th ACM International Conference on Multimedia (MM)*, pages 445–447, Los Angeles, CA, October 2000. ACM Press.

[120] Robert Tansley. *The Multimedia Thesaurus: Adding A Semantic Layer to Multimedia Information.* PhD thesis, University of Southampton, UK, August 2000.

[121] Jelena Tešić, Sitaram Bhagavathy, and B. S. Manjunath. Issues Concerning Dimensionality and Similarity Search. In *International Symposium on Image and Signal Processing and Analysis (ISPA)*, Rome, Italy, September 2003.

[122] Jelena Tešić and B. S. Manjunath. Nearest Neighbor Search for Relevance Feedback. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 643–648, Madison, WI, June 2003.

[123] Jelena Tešić, Shawn Newsam, and B.S. Manjunath. Scalable Spatial Event Representation. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, Lausanne, Switzerland, August 2002.

[124] Jelena Tešić, Shawn Newsam, and B.S. Manjunath. Mining Image Datasets using Perceptual Association Rules. In *SIAM Sixth Workshop on Mining Scientific and Engineering Datasets in conjunction with SIAM/SDM*, San Francisco, CA, May 2003.

[125] Mihran Tuceryan and Anil K. Jain. Texture analysis.

[126] Ertem Tuncel, Hakan Ferhatosmanoglu, and Kenneth Rose. Vq-index: An index structure for similarity searching in multimedia databases. In *Proceedings of 10th ACM International Conference on Multimedia (MM)*, pages 543–552, Juan-les-Pins, France, December 2002. ACM Press.

[127] Nuno Vasconcelos. On the complexity of probabilistic image retrieval. In *Proceedings of 8th International Conference On Computer Vision (ICCV)*, volume 2, pages 400–407, Vancouver, BC, Canada, July 2001. IEEE Computer Society.

[128] Remco C. Veltkamp and Mirela Tanase. Content-based image retrieval systems: A survey. Technical Report UU-CS-2000-34, Utrect University, The Netherlands, October 2000.

[129] James Z. Wang, Jia Li, and Gio Wiederhold. SIMPLIcity: Semantics-sensitive integrated matching for picture libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 23(9), August 2001.

[130] Jason Tsong-Li Wang, Xiong Wang, King-Ip Lin, Dennis Shasha, Bruce A. Shapiro, and Kaizhong Zhang. Evaluating a class of distance-mapping algorithms for data mining and clustering. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 307–311, San Diego, California, United States, 1999. ACM Press.

[131] Jason Tsong-Li Wang, Xiong Wang, King-Ip Lin, Dennis Shasha, Bruce A.

Shapiro, and Kaizhong Zhang. Evaluating a class of distance mapping algorithms for data mining and clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge discovery and data mining (KDD)*, pages 307–311, San Diego, California, August 1999. ACM Press.

[132] Xiong Wang, Jason Tsong-Li Wang, King-Ip Lin, Dennis Shasha, Bruce A. Shapiro, and Kaizhong Zhang. An Index Structure for Data Mining and Clustering. *Knowledge and Information Systems*, 2(2), 2000.

[133] Roger Weber and Klemens Böhm". Trading quality for time with nearest-neighbor search. *Lecture Notes in Computer Science*, 1777:21–36, 2000.

[134] Roger Weber, Hans-Jörg Schek, and Stephen Blott. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. In Ashish Gupta, Oded Shmueli, and Jennifer Widom, editors, *Proceedings of 24rd International Conference on Very Large Data Bases (VLDB)*, pages 194–205, New York City, NY, August 1998. Morgan Kaufmann.

[135] Thomas P. Weldon, William E. Higgins, and Dennis F. Dunn. Gabor filter design for multiple texture segmentation. *Optical Engineering*, 235(10):2852–2863, 1996.

[136] David A. White and Ramesh Jain. Similarity indexing with the ss-tree. In Stanley Y. W. Su, editor, *Proceedings of 12th IEEE International Conference on Data Engineering (ICDE)*, pages 516–523, New Orleans, Louisiana, February 1996. IEEE Computer Society.

[137] M. E. J. Wood, Barry T. Thomas, and Neill W. Campbell. Iterative refinement by relevance feedback in content-based digital image retrieval. In *Proceedings of 6th ACM International Conference on Multimedia (MM)*, pages 13–20, Bristol, UK, September 1998. ACM Press.

[138] Peng Wu and B.S. Manjunath. Adaptive nearest neighbor search for relevance feedback in large image datasets. In *Proceedings of 9th ACM International Conference on Multimedia (MM)*, pages 89–97, Ottawa, Canada, October 2001. ACM Press.

[139] Peng Wu, B.S. Manjunath, and Shivkumar Chandrasekaran. An adaptive index structure for high-dimensional similarity search. In *Proceedings of 2nd IEEE Pacific Rim Conference on Multimedia, Advances in Multimedia Information Processing (PCM)*, volume 2195 of *Lecture Notes in Computer Science*, pages 71–77. Springer, October 2001.

[140] Peng Wu, B.S. Manjunath, and H.D. Shin. Dimensionality reduction for image search and retrieval. In *Proceedings of the IEEE International. Conference on Image Processing (ICIP 2000)*, Vancouver, Canada,, September 2000.

[141] Yi-Leh Wu, King-Shy Goh, Beitao Li, Huaxing You, and Edward Y. Chang. The anatomy of a multimodal information filter. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 462–471, Washington, D.C., 2003. ACM Press.

[142] Yaowu Xu, Pinar Duygulu, Eli Saber, Murat Tekalp, and Fatos Yarman Vural. Object based image labeling through learning-by-example and multi-level segmentation. *Pattern Recognition*, 36(6):1407–1423, June 2003.

[143] Xifeng Yan, Jiawei Han, and Ramin Afshar. Clospan: Mining closed sequential patterns in large databases. In Daniel Barbará and Chandrika Kamath, editors, *Proceedings of 3rd SIAM International Conference on Data Mining (SDM)*, San Francisco, CA, May 2003.

[144] Atsuo Yoshitaka and Tadao Ichikawa. A survey on content-based retrieval for multimedia databases. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 11(1), February 1999.

[145] Tian Zhang, Raghu Ramakrishnan, and Miron Livny. Birch: an efficient data clustering method for very large databases. *SIGMOD Record*, 25(2):103–114, 1996.

[146] Xiang Sean Zhou and Thomas Huang. Biasmap for small sample learning during multimedia retrieval. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 425–430, San Francisco, CA, June 2001. IEEE Computer Society.

[147] Xiang Sean Zhou and Thomas Huang. Relevance feedback for image retrieval: a comprehensive review. *ACM Multimedia System*, 8(6), 2003.

# Appendix A

# Appendix

## A.1 Properties of Rice Distribution

The magnitude of the Gabor filter output can be represented as 3.10:

$$t(x) = |r(x)| = \sqrt{{n'_I}^2(x) + n_Q^2(x)}, \tag{A.1}$$

where $n'_I(x) = U + n_I(Q)$ and $n_I(x)$ and $n_Q(x)$ have independent, zero mean Gaussian distribution with variance $\gamma^2$. The probability distribution function (PDF) is Rician:

$$p_a(r) = \frac{r}{\gamma^2} \exp\left(-\frac{r^2 + U^2}{2\gamma^2}\right) I_0\left(\frac{Ur}{\gamma^2},\right) \tag{A.2}$$

where $I_0(x)$ is the zero-order modified Bessel function of the first kind.

Signal to noise ratio (SNR) is defined as $a = U/\gamma$.

If $a \to 0$ i.e. **SNR is low**, the modified Bessel function of order $\nu$ is approximated like:

$$I_\nu(\frac{ar}{\gamma}) \sim (\frac{ar}{2\gamma})^\nu \Gamma(\nu+1) \qquad (a \to 0). \tag{A.3}$$

For $\nu = 0$ and low SNR, $I_0(ax) \sim 1$. In this case, the Rician distribution can be approximated with Rayleigh distribution i.e.

$$p_a(r) = \frac{r}{\gamma^2} \exp\left(-\frac{r^2}{2\gamma^2}\right) e^{-\frac{a^2}{2}} I_0\left(\frac{Ur}{\gamma^2}\right) \sim \frac{r}{\gamma^2} \exp\left(-\frac{r^2}{2\gamma^2}\right) \qquad (a \to 0). \tag{A.4}$$

If $a \gg 0$ i.e. **SNR is high**, the modified Bessel function of order $\nu$ is approximated with

$$I_\nu(\frac{ar}{\gamma}) \sim exp\left(\frac{ar}{\gamma}\right) \sqrt{\frac{\gamma}{2\pi ar}} \qquad (a \to \infty). \tag{A.5}$$

Therefore:

$$p_a(r) = \frac{r}{\gamma^2} \exp\left(-\frac{r^2+U^2}{2\gamma^2}\right) I_0\left(\frac{Ur}{\gamma^2}\right) \sim \frac{r}{\gamma^2} \exp\left(-\frac{r^2+U^2-2rU}{2\gamma^2}\right) \frac{\gamma}{\sqrt{2\pi rU}}. \tag{A.6}$$

If $\sqrt{(\frac{r}{U})} \to 1$for $a \to \infty$, Rice pdf can be approximated with Gaussian:

$$p_a(r) \sim \frac{1}{\sqrt{2\pi\gamma^2}} e^{-(r-U)^2/(2\gamma^2)} \sqrt{\frac{r}{U}} \sim \frac{1}{\sqrt{2\pi\gamma^2}} e^{-(r-U)^2/(2\gamma^2)}. \tag{A.7}$$

If a variable $Z$ is the magnitude of a sum of 2A Gaussian distributed random variables, and its underlying distribution is the *generalized Rice* distribution (3.17).

$$Z = \sqrt{\sum_{i=1}^{2} An_i^2}, \tag{A.8}$$

where $n_i$ has independent, Gaussian distributions with mean $U_i$ and variance $\sigma$.

If $U_o^2 = \sum_i U_i^2$ pdf of $Z$ is can be written as:

$$p_Z(z) = \frac{z}{\sigma^2}\left(\frac{z}{U_o}\right)^{A-1} exp(-\frac{z^2+U_o^2}{2\sigma^2})I_{A-1}\left(\frac{zU_o}{\sigma^2}\right). \tag{A.9}$$

Signal to noise ratio is defined as $a = U_o/\sigma$.

For a **low SNR** ratio (A.3) the distribution of $Z$ can be approximated by a generalized Rayleigh distribution. General Rayleigh distribution is defined as a $\chi$-distribution (Chi) with scale parameter equal to 1.

$$p_Z(z) = \frac{2z^{2A-1}}{(2\sigma^2)^A\Gamma(A)}e^{-\frac{z^2}{2\sigma^2}} = \frac{1}{\sigma^2}\chi(\frac{z}{\sigma}, 2A). \tag{A.10}$$

For a **high SNR** ratio (A.5) the distribution of $Z$ can be approximated by a Gaussian distribution if $\sqrt{z}U_o \to 1$ when $a \to \infty$:

$$p_a(z) \sim \frac{1}{\sqrt{2\pi\sigma^2}}\left(\sqrt{\frac{z}{U_o}}\right)^{A-1}\exp\left(-\frac{z^2+U_o^2-2zU_o}{2\sigma^2}\right) \sim \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(z-U_o)^2}{2\sigma^2}\right). \tag{A.11}$$