

G60/155

No Reservations.

MONASH UNIVERSITY
THESIS ACCEPTED IN SATISFACTION OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

ON..... 12 July 2005.....
.....

Sec. Research Graduate School Committee

Under the Copyright Act 1968, this thesis must be used only under the normal conditions of scholarly fair dealing for the purposes of research, criticism or review. In particular no results or conclusions should be extracted from it, nor should it be copied or closely paraphrased in whole or in part without the written consent of the author. Proper written acknowledgement should be made for any assistance obtained from this thesis.

ERRATA

p 108 last para, line 3: "CBIR" for "CBI".

ADDENDUM

p iii para 3, line 5: Insert "of the six colour spaces evaluated," after "It also shows that".

p iii para 4, line 4: Add at the end of the sentence "among the methods evaluated".

p iv para 2, line 3: Delete "eyeMap is the best way of" and add "eyeMap is a better method than the traditional methods for".

p 154: Add at the end of para 1:

"One possible theoretical explanation for the increased performance is as follows. Visual perception requires two types of processing: first, visual system for receiving stimulus; and second, further neural processing to actively select which stimuli to attend to (attention) [1]. Attention directs the visual system to stimuli we want to perceive and affects how the information is processed. Consequently, attention can enhance the perception of stimuli we are paying attention to and decrease the perception of stimuli being ignored. The increased performance of eyeMap suggests it is more sympathetic to this phenomenon. Images in eyeMap were organised in such a way that locations which were more likely to contain target images attracted more attention, thus enhancing users' perception of images in those areas. For this reason, users could find target images faster using eyeMap than using the traditional display."

Bibliography

- [1] E B Goldstein. *Sensation and Perception*. Wadsworth, Wadsworth-Thomson Learning, 2002.

Feature Extraction, Browsing and Retrieval of Images

Suryani Lim

BComp (Hons), Monash University

A thesis submitted for the degree of
Doctor of Philosophy

Gippsland School of Computing and Information Technology

Monash University

Victoria, Australia

February 2005

© Suryani Lim

Typeset in Computer Modern by T_EX and L^AT_EX 2_ε.

Abstract

The large volume of digital image databases challenges the usefulness of the classical means of manual image annotation and demands a more efficient and effective way to generate feature vectors and retrieve images. This need inspires the research into image retrieval using feature vectors generated from the content of the images rather than their annotation, hence the name content-based image retrieval (CBIR). It has been an active research area for over a decade but to date has not been widely used in real world applications.

This thesis aims to make CBIR more useful in real world applications by using innovative approaches to (1) evaluate the suitability of colour spaces for colour-based CBIR (2) develop a new feature extraction method to generate feature vectors for colour images (3) formulate a framework for facilitating intuitive and effective image browsing and retrieval (4) evaluate the performance of this framework for image browsing and retrieval.

To improve the retrieval effectiveness and efficiency for retrieving colour images, this research first resolves which colour space is most suitable for colour-based CBIR by evaluating the suitability of six colour spaces. This is crucial because it justifies the choice of colour space and provides insights into why some colour spaces are more suitable than others. It also shows that HSV colour space is most suitable for colour-based CBIR as it is at least as effective as but more efficient than any of the other colour spaces.

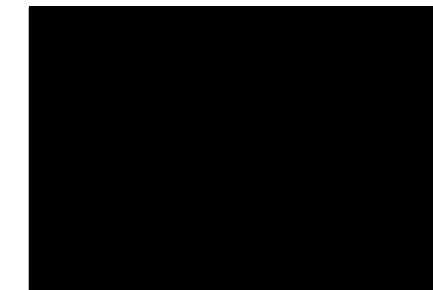
Further, this research determines how to best use the spatial relationships of colours for retrieving colour images. It proposes a new feature extraction method, I-autocorrelogram (I-auto), for colour images and compares it against contemporary methods. I-auto is found to be, overall, the most preferred method.

In addition, to facilitate intuitive and effective image retrieval, this research first formulates eyeMap, an image browsing framework for large image databases which can be integrated into a CBIR system. eyeMap not only provides users an overview of all images in the database, but also helps them to select an appropriate sample image to start the query-by-example retrieval process. This research, then develops colour-based eyeMap for browsing colour image databases and integrates it into a colour-based CBIR system developed using I-auto for retrieving colour images.

Significantly, when colour-based eyeMap and traditional methods were tested on users, colour-based eyeMap was found to be more effective, efficient and most preferred. It thus shows that eyeMap is the best way of interacting with CBIR systems in the real world. This research finally demonstrates how eyeMap can be used with texture image databases by creating texture-based eyeMap.

Declaration

I declare that this thesis has not been submitted for any other degree in any institutions, and to the best of my knowledge, except where otherwise indicated, this thesis is my own original work.



Suryan Lini

February 2005

Acknowledgements

Writing the acknowledgements is one of the most joyous moments of my PhD candidature, for it gives me the opportunity to thank people who have contributed to the completion of this thesis. I am fortunate to have Associate Professor Guojun Lu to guide me through this challenging period. Thank you for teaching me analytical and critical thinking, keeping me on track and giving me the freedom to explore. Many thanks also to Dr Ray Smith, my associate supervisor, for sharing your knowledge and giving encouragement. I also appreciate the regular meetings within our research group in which I have the opportunity to discuss, debate and *debug* my research with both supervisors, other fellow students and academics. Of special mention is Dr Alistair Carr from the School of Applied Science and Engineering for his sense of humour and free mathematics lessons.

I am blessed with supportive and caring staff at the Gippsland School of Information and Technology (GSCIT), who not only encourage me but also organise social events and sporting challenges for the research students and staff. In particular, I would like to thank Ms Kerry Luke for prompt administrative assistance and Dr Manzur Murshed, GSCIT's research director, for looking after my needs as a research student throughout my studies. I acknowledge the support from Monash University in providing the scholarships and travel grants. I am also grateful for the additional travel grants from GSCIT.

I am lucky to be surrounded by many wonderful friends. More specifically, I thank Dr Kaya Prpic whose passion in the quest for and transfer of knowledge is most inspirational. I have learnt much from you. Special thanks also go to Ms Julie Murray for volunteering to proof-read this thesis. I also thank you for your encouragement and positive attitude, and I value your expertise, effort and patience in teaching me how to

write clearly.

I am grateful for having a loving family, who provide me with undeniable excuses for holidays and look after me whenever I am with you. More specifically, I thank my parents for showing me the meaning of determination and inspiring me to strive for this virtue, without which I would not have completed this thesis. Many thanks also to Adi and Dian for all the pleasant surprises and company. Finally, I would like to thank my soul mate, Simon, for absorbing all the impacts of seemingly endless cycles of hope and despair throughout my studies. Thank you for having the confidence in me when I had absolutely none left, as your faith was uplifting even when my morale hit rock bottom.

Contents

Abstract	iii
Declaration	v
Acknowledgements	vii
List of Acronyms	xix
1 Introduction	1
1.1 Background	1
1.2 Objectives	2
1.3 Contributions of Thesis	3
1.3.1 Evaluating Suitability of Colour Spaces for Colour-Based CBIR .	3
1.3.2 Improving Effectiveness of Colour-Based Feature Vectors	3
1.3.3 Formulating eyeMap, a Framework for Browsing Large Image Databases	4
1.3.4 Developing Colour-Based eyeMap for Colour Images	5
1.3.5 Usability Study of Colour-Based eyeMap	5
1.3.6 Creating Texture-Based eyeMap for Texture Images	6
1.4 Research Overview	7
1.5 Structure of Thesis	8

2 A Review of Content-Based Image Retrieval	9
2.1 Colour-Based Methods	10
2.1.1 Colour Distribution	11
2.1.1.1 Globally Quantised Colours	13
2.1.1.2 Locally Quantised Colours	16
2.1.1.3 No Colour Quantisation	18
2.1.2 Spatial Relationships of Colours	19
2.1.2.1 Pixel-Based: Pixel Classification	20
2.1.2.2 Pixel Based: Spatial Descriptors	23
2.1.2.3 Block-based: Vector Quantisation (VQ)	24
2.2 Texture-Based Methods	25
2.3 Shape-Based Methods	26
2.4 Integrated Methods	27
2.5 Conclusions	28
3 Evaluating Suitability of Different Colour Spaces for Colour-Based CBIR	31
3.1 Evaluation Method and Experimental Design	32
3.1.1 Evaluation Criteria	33
3.1.1.1 Retrieval Effectiveness	33
3.1.1.2 Retrieval Efficiency	34
3.1.2 Image Databases	34
3.1.3 Retrieval Software	35
3.2 The Six Colour Spaces Evaluated	36

3.2.1 Characteristics of Six Colour Spaces	36
3.2.1.1 Characteristics of RGB	36
3.2.1.2 Characteristics of HSV	37
3.2.1.3 Characteristics of LUV and LAB in Cartesian Co-ordinates	39
3.2.1.4 Characteristics of LUV and LAB in Polar Co-ordinates (pLUV and pLAB)	40
3.2.2 Parameters for Colour Space Quantisations	42
3.2.2.1 Colour Space Quantisations in Cartesian Co-ordinates	43
3.2.2.2 Colour Space Quantisations in Polar Co-ordinates	45
3.3 Results and Discussion	47
3.3.1 Retrieval Effectiveness	47
3.3.1.1 Effectiveness of Each Colour Space in Cartesian Co-ordinates	48
3.3.1.2 Effectiveness of Each Colour Space in Polar Co-ordinates	52
3.3.1.3 Comparing Retrieval Effectiveness of Six Colour Spaces	57
3.3.2 Retrieval Efficiency	58
3.3.2.1 Method one: comparing the size of feature vectors	59
3.3.2.2 Method two: comparing the effectiveness	59
3.4 Conclusions	62
4 Spatial Information for Image Feature Extraction	65
4.1 Colour Autocorrelogram	65
4.1.1 Feature Extraction	66
4.1.2 Distance Calculation for Retrieval	66

4.1.3	Analysis	67
4.2	Improving Autocorrelogram	68
4.3	Experimental Parameters	70
4.3.1	Dissimilarity Metric	70
4.3.2	Choice of $[d]$	71
4.3.3	Colour Space	71
4.3.4	Weighting of w_1	71
4.4	Experimental Setup and Evaluation Criteria	72
4.5	Results and Discussion	72
4.5.1	Effect of Dissimilarity Metric	72
4.5.2	Effect of $[d]$	72
4.5.3	Effect of Colour Space	75
4.5.4	Weighting of w_1	77
4.5.5	Effectiveness of New Feature Extraction Method (I-auto)	77
4.6	Evaluating I-auto Against Contemporary Feature Extraction Methods	80
4.6.1	Spectrally Layered Colour Indexing (LCI)	80
4.6.2	MPEG-7 Colour Structure Descriptor (CSD)	81
4.6.3	Experimental Setup and Evaluation Criteria	82
4.6.4	Results and Discussion	82
4.6.4.1	Analysing Results of I-auto and CSD	83
4.6.4.2	Analysing Results of I-auto and LCI	83
4.7	Summary	84
4.8	Conclusions	84

5	Browsing Large Scale Image Databases	87
5.1	Previous Work	88
5.2	Towards Browsing Large Scale Image Databases	93
5.3	Proximity Visualisation	93
5.3.1	Basic Concept	94
5.3.2	Deriving Layouts for Proximity Visualisation Using Multidimensional Scaling (MDS)	94
5.4	eyeMap, an Image Browsing Framework for Large Image Databases	96
5.4.1	Image Clustering	97
5.4.2	Removal of Image Overlapping	98
5.4.2.1	Distortion Oriented Displays (DOD)	98
5.4.2.2	Linear Display	103
5.4.3	Searching for a Target Image by Elimination	103
5.4.4	Summary of eyeMap Specification	103
5.5	Applications of eyeMap	104
5.5.1	Browsing	104
5.5.2	Finding a Sample Image to Initiate a Visual Query	106
5.6	Conclusions	107
6	Colour-Based eyeMap: Browsing Colour Image Databases	109
6.1	Feature Selection: Evaluation Method and Design	109
6.1.1	Evaluation Criteria	110
6.1.1.1	Spatial Precision and Recall Graphs (Spatial PR Graphs)	111
6.1.1.2	Visual Inspection	112

6.1.2	Generating Layouts: Colour Features and Their Parameters . . .	112
6.2	Results and Discussion	113
6.2.1	Spatial PR Graphs	113
6.2.2	Relationships between PR Graphs and Spatial PR Graphs . . .	114
6.2.3	Visual Inspection	116
6.2.4	Further Discussions on Contextual Meaningfulness	121
6.3	Implementations of Colour-Based eyeMap	124
6.4	Conclusions	127
7	Usability Study of Colour-Based eyeMap	129
7.1	Evaluation Methods and Design	130
7.1.1	Evaluation Criteria	130
7.1.2	Descriptions of Existing Methods and Colour-Based eyeMaps . .	133
7.1.3	Experimental Design	137
7.1.3.1	Procedure	140
7.1.3.2	Image Sets	142
7.1.3.3	Pilot Study	143
7.1.3.4	Participants	144
7.2	Results and Discussion	145
7.2.1	Successful Search	145
7.2.1.1	In Solving Page 0 Problem	145
7.2.1.2	In Browsing	147
7.2.2	Search Efficiency	149
7.2.2.1	In Solving Page 0 Problem	152

7.2.2.2	In Browsing	153
7.2.2.3	Usefulness of Visual Query and Visualisation Layout . .	155
7.2.2.4	Differences in Search Strategies	157
7.2.3	Program Ratings	159
7.2.4	Preferred Program	166
7.3	Conclusions	167
8	Texture-Based eyeMap: Browsing Texture Image Databases	169
8.1	How to Browse a Texture Image Database	170
8.1.1	Texture Retrieval Descriptor (TRD)	170
8.1.1.1	Feature Extraction	170
8.1.1.2	Deriving a Layout Using MDS	172
8.1.2	Texture Browsing Descriptor (TBD)	174
8.1.2.1	Deriving a Layout Using MDS	175
8.1.2.2	Deriving a Layout by Feature Selection	175
8.2	Experimental Setup and Evaluation Criteria	176
8.3	Results and Discussion	177
8.3.1	Spatial PR Graphs	177
8.3.2	Visual Inspection	179
8.3.2.1	MDS-EMD Layout	179
8.3.2.2	MDS-L1 Layout	179
8.3.2.3	MDS-Cum Layout	180
8.3.2.4	MDS-TBD Layout	180
8.3.2.5	2D-TBD Layout	181

8.3.2.6	Outliers in MDS-EMD and MDS-L1	187
8.3.3	Summary and Implications	192
8.4	Conclusions	192
9	Conclusions	195
9.1	Summary of Main Findings	195
9.2	Potential Future Research Directions	198
	Bibliography	199
A	Key Publications	217
B	Supplementary Results for Colour Space Evaluations Presented in Chapter 3	219
B.1	PR Graphs in PCD for Cartesian Coordinate Colour Spaces	219
B.2	PR Graphs in PCD for Polar Coordinates Colour Spaces	224
B.3	PR Graphs in PCD for Most Effective Quantisation Options	231
B.4	PR Graphs in PCD for Quantisation Options at Similar Sizes	232
C	Supplementary Results for Colour Spatial Features Presented in Chapter 4	233
C.1	Autocorrelogram - Effect of Dissimilarity Metrics in CCD and PCD	234
C.2	Autocorrelogram - Effect of d in PCD	236
C.3	Autocorrelogram - Effect of Colour Space in PCD	237
C.4	I-auto - Weighting of w_1 in PCD	238
C.5	Effect of Colour Spaces on I-auto in PCD	241
C.6	Effectiveness of I-auto Compared to LCI and CSD in PCD	242

D	Post Experiment Questionnaire of Usability Study Presented in Chapter 7	243
---	---	-----

List of Acronyms

- ANOVA Analysis of Variance A hypothesis testing method which tests if the difference of means between multiple populations is statistically significant.
- CCD Common Colour Database A database of 5,466 natural colour images used by the MPEG-7 committee for testing colour descriptors.
- CCQ Common Colour Queries The 50 images defined as the query images in CCD. The relevant images for these 50 images have been predefined by the MPEG-7 standard. See also CCD.
- CSD Colour Structure Descriptor A feature extraction method defined in the MPEG-7 standard.
- DOD Distortion Oriented Displays A type of display which shows both the overview and detailed information at the same time.
- GIFT Gnu Image Finding Tool An open source client server CBIR framework useful for benchmarking the effectiveness of feature descriptors in CBIR.
- LCI Spectrally Layered Colour Indexing A colour descriptor which derives its colour spatial relationships in the frequency domain.
- MDS Multidimensional Scaling A multivariate statistical analysis method which is often used for dimensionality reduction. Unlike PCA, MDS is a non-linear transformation of data from the high dimension into low dimension. See also PCA.

-
- MPEG..... Moving Picture Experts Group An International Standard Organisation (ISO) committee.
- MPEG-7..... also known as "Multimedia Content Description Interface". It is an ISO/IEC standard developed by MPEG for providing a set of tools which describe multimedia content including still images.
- PCA..... Principal Component Analysis A multivariate statistical method which is often used for dimension reduction. PCA linearly transform feature vectors from high dimension to low dimension. See also MDS.
- PCD..... Proprietary Colour Database A database of 10,112 natural colour images used by the multimedia research group at Monash University, Gippsland School of Computing and Information Technology.
- TBD..... Texture Browsing Descriptor A texture descriptor defined in the MPEG-7 standard for browsing texture images.
- TRD..... Texture Retrieval Descriptor A texture descriptor defined in the MPEG-7 standard for retrieving texture images.

Introduction

1.1 Background

The birth of digital devices for capturing data and the low cost of electronic storage saw an explosive increase in the volume of archived data. The first stage of this revolution was marked by the storage of textual documents and, later, other media such as audio, video and still images. Research in text retrieval for the last thirty years has made the retrieval of textual data a breeze; thus, images are often annotated with text, thereby transforming the problem of image retrieval to text retrieval. However, text annotation is tedious, expensive, and for image databases it is also inaccurate because of the subjectivity and difficulty of describing the visual features; consequently, the rate of successful retrieval is low [42, 74].

Another approach to image retrieval is to automatically generate feature vectors from the content of images so that retrieval can be performed by searching the feature vectors; hence, the name content-based image retrieval (CBIR). CBIR differs from the traditional text-based systems in two ways: generating of feature vectors (feature extraction) and initiating a query. In terms of feature extraction, text-based systems can directly use the content of a document to build the feature vector. In contrast, CBIR *cannot* directly use the content of an image to build the feature vector as images are made up of pixel arrays which are meaningless; image feature extraction can only be achieved by extracting useful features from the raw data. The retrieval effectiveness largely relies on feature extraction, and this has been the research focus in CBIR for the last decade [3, 34, 71, 142, 155, 162]. Because object recognition is highly unreliable,

feature extraction has been and still is limited to low level features such as colours, shapes, textures, or combinations of these, and in most cases, this is acceptable given that similar objects tend to have similar low level features.

The second difference between CBIR and text-based systems lies in how to initiate a query. The input into text-based systems is obviously text whilst for CBIR, it is no longer as straightforward. This issue receives far less attention than feature extraction, for it has always been assumed that users must first issue a text query, and once they have a sample image, then they issue a visual query, that is, by supplying a sample image to retrieve a set of relevant images. In the absence of text annotation or a sample image, users are assumed to be capable of sketching a sample image. This expectation is unrealistic as it requires users not only to sketch, but also to have an intimate knowledge of the system's use of the feature in order to sketch successfully. This problem, the problem of initiating a visual query without a sample image, is known as the Page 0 problem.

1.2 Objectives

To date, CBIR systems have had little practical use mainly because of low retrieval effectiveness, poor efficiency and the Page 0 problem. The main objective of the research in this thesis is to bring CBIR systems one step closer to real world applications by improving the effectiveness and efficiency of colour-based feature vectors and image retrieval, and solving the Page 0 problem. The framework developed for solving the Page 0 problem is useful for browsing and searching large collections of different image database types. A colour-based implementation of this framework is suitable for use with large general colour image databases i.e. commercial photo stock libraries, personal photographs and digital art collections, whereas a texture-based implementation is appropriate for texture images in the real world i.e. textiles, carpets and wallpapers.

1.3 Contributions of Thesis

The research contributions of this thesis can be divided into six major sections:

1. evaluating the suitability of colour spaces for colour-based CBIR;
2. improving the effectiveness of colour-based feature vectors;
3. formulating eyeMap, a framework for browsing large image databases;
4. developing colour-based eyeMap for colour images;
5. evaluating the usability of colour-based eyeMap; and
6. creating texture-based eyeMap for texture images.

1.3.1 Evaluating Suitability of Colour Spaces for Colour-Based CBIR

A colour is described in at least three coordinates and a collection of all colours in the coordinate system is called a colour space. Evaluating the suitability of colour spaces is important because colour-based features remain popular for describing the content of general colour images, and the use of any colour-based features requires the selection of a colour space. However, there has been no comprehensive study on how to select a suitable colour space. The research in this thesis comprehensively studied the effectiveness and efficiency of the most commonly used colour spaces (that is, RGB, LUV and LAB in Cartesian co-ordinates as well as HSV, LUV and LAB in polar coordinates) for colour-based CBIR. This study is important because it justifies the choice of colour space and provides insights into why some colour spaces are more suitable than others. We found that HSV is most suitable for colour-based CBIR because it is both effective and efficient, and reported the results in [62].

1.3.2 Improving Effectiveness of Colour-Based Feature Vectors

More recent colour-based feature vectors have incorporated the rich information provided by the spatial relationships of colours previously ignored. In this thesis, we

studied autocorrelogram, a well known and promising feature vector, which incorporates the spatial relationships of colours. The findings acquired from the study led us to the discovery of I-autocorrelogram (I-auto), an even more effective and efficient feature vector. I-auto was then evaluated against other contemporary feature vectors and was found to be most preferred. The findings related to the research in feature vectors incorporating colours spatial relationships are published in [63, 64].

1.3.3 Formulating eyeMap, a Framework for Browsing Large Image Databases

Research in CBIR is mostly restricted to retrieval but browsing is equally important, as it allows users to have an overall view of the entire database and to more easily find a sample image to initiate a visual query, practically solving the Page 0 problem. Browsing a large scale image database is difficult because the area required to display all images easily exceeds the available screen area. The requirements of browsing are different from those of retrieval. For browsing, the browser must display all images in the database so that users have an overview of its content. The relationships between images must also be clear to users so they can decide where to browse next. These relationships can only be established if the display or layout is contextually meaningful; that is, the display reflects the perceptual differences and similarities of the images: a random display does not facilitate browsing.

The development of the browsing framework in this thesis, eyeMap, took a more holistic approach to solving the issues related to browsing large image databases. We investigated techniques for handling large amounts of data and adapted these techniques so that they are suitable for browsing large image databases. These techniques are generic, so they can be used for other image database types. eyeMap differs from existing image browsing frameworks in that it facilitates the display of a large image collection at any one time and allows users to intuitively focus on the area of interest. The implementation of eyeMap provides a powerful browsing tool and, when it is integrated with a CBIR system, it is a useful tool for finding a sample image to initiate a

visual query, thus solving the Page 0 problem.

1.3.4 Developing Colour-Based eyeMap for Colour Images

Because eyeMap is a browsing framework which can be used for browsing any types of image databases, we first explored the use of eyeMap for browsing colour image databases by creating colour-based eyeMap. To illustrate why eyeMap is useful for browsing, Fig. 1.1 on the following page shows a layout of a general colour image database produced by colour-based eyeMap. The images are grouped by visual similarity, so users can direct their attention to the area of interest by moving the ellipse. The ellipse identifies the area of interest (focal region), and users can clearly see all images within the ellipse because the images are enlarged and any image overlapping can be removed. In order to ensure that layouts of images are meaningful, we established which colour feature, among several, was more suitable for browsing. It was found that a cumulative histogram colour feature is more suitable for browsing general colour images. The work related to the development of colour-based eyeMap was published in [65].

1.3.5 Usability Study of Colour-Based eyeMap

This usability study pioneers the evaluation of image browsing systems. The purpose of the study was to show that eyeMap is useful for browsing and solving the Page 0 problem by comparing colour-based eyeMap against traditional linear browsing methods. The study establishes that eyeMap is the best framework, as the systems developed based on eyeMap are most effective, efficient and preferred by users. In addition, this study provides insights into how humans search for images, so the findings are also useful for designers of any image browsing or search applications.

1.5 Structure of Thesis

Chapter 2 contains an overview of existing CBIR techniques which use colours, textures, shapes and combinations of these. The aim of this chapter is to review and analyse the strengths and limitations of relevant work in CBIR by reviewing the current literature.

Chapter 3 presents the results of experiments in different colour spaces. The purpose of this chapter is to identify which colour space is most suitable for colour-based CBIR and to explain why some are more suitable than others by conducting retrieval experiments in different colour spaces.

Chapter 4 describes the studies related to colour-based feature vectors which incorporate colours spatial relationships. The objective of this chapter is to demonstrate how to improve the effectiveness and efficiency of these feature vectors.

Chapter 5 focuses on browsing large image databases. The purpose of this chapter is to solve the problems associated with browsing large image databases by formulating eyeMap, a new image browsing framework.

Chapter 6 discusses the use of eyeMap for browsing general colour images, and this implementation of eyeMap is known as colour-based eyeMap. In order to implement colour-based eyeMap, we demonstrate which colour features are more suitable for browsing colour images by evaluating different colour features.

Chapter 7 aims to determine whether eyeMap is useful for browsing and solving the Page 0 problem by having users test colour-based eyeMap and traditional methods. This study also provides insights into how humans search for images and the findings are useful for designing any image browsing or search method.

Chapter 8 describes texture-based eyeMap. This chapter shows how to use eyeMap for browsing homogeneous texture images by investigating different texture descriptors. The purpose of the study is to demonstrate how to visualise texture images and to determine which texture descriptor is more suitable for browsing.

Chapter 9 summarises the main findings and provides potential future directions.

A Review of Content-Based Image Retrieval

When computers were first used to store a large volume of textual digital documents, searching for a specific document was challenging. The solution to this problem was the use of automatic feature extraction to generate feature vectors to facilitate retrieval. This marked the birth of information retrieval. The need for a good search engine is more important than ever as the volume of documents will only increase. With cheaper and higher storage capacity, we can now even store large quantity of digitised multimedia data, such as still images. Searching for images presents research communities with an even bigger challenge because extracting features from text documents is straightforward as the words in the documents can be directly used as the feature vectors; but, for images, what aspects could be used as feature vectors?

Traditionally, each image is manually annotated and users search the annotation to find the desired images. However, manual annotation is expensive, time consuming and subjective. For example, Furnas et al. reported that the probability of two people using the same words to describe the same objects varies between 7%–18% depending on the objects [42]. The effectiveness of the search engine is largely influenced by the degree of agreement between the annotator and users; as a result, searching annotated images remains ineffective. Another approach to image retrieval is to extract feature vectors using the content of images so that they can be retrieved by searching from these feature vectors, hence the name content-based image retrieval (CBIR). It is a simple concept but extremely difficult to implement accurately because automatic image and object

recognition is a very difficult task. The alternative is to extract the low level features such as colours, textures and shapes. This is acceptable under most circumstances because visually similar images or objects tend to have similar low level features.

This chapter contains a review of feature extraction methods, and a taxonomy of these methods is given in Fig. 2.1. Colour-based methods are normally used for general colour images; texture-based methods for homogeneous texture images such as textiles, carpets, wall papers and finger prints; and shape-based methods for bi-level shape images i.e. images which have only black and white pixels such as logos. This literature review also covers feature extraction methods which generate feature vectors by integrating several low level features.

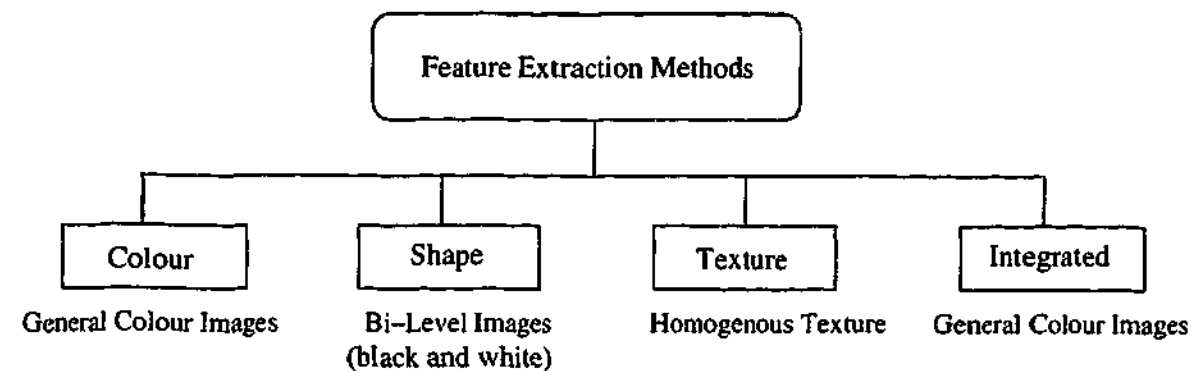


Figure 2.1: Taxonomy of feature extraction methods used in CBIR.

2.1 Colour-Based Methods

All colour-based methods are proposed based on the observation that similar images tend to have similar colour content. So, all these methods evolve around how best to describe the colour content of the images so that similar images will have similar colour descriptions and dissimilar images will have dissimilar colour descriptions. Colours are described using at least three coordinates, and a collection of all colours in a coordinate system is called a colour space. The very first decision in extracting colour-based feature vectors is to choose a colour space to describe the colours. (The methodology of colour space selection is beyond this chapter; it is an objective to be studied in Chapter 3). Computers can differentiate over 2 million colours but human eyes can differentiate

substantially fewer colours. To describe the colour content effectively, all colour-based methods massively reduce the number of colours either by quantising the colour space or describing only their statistical properties. Colour space quantisation can be written formally as follows [124]. Let C be a colour space. Let P be a quantisation space (a subset of C) and $P = \{c_1, c_2, \dots, c_i, \dots, c_n \mid c_i \in C, n \ll \|C\|\}$ where n is the bin size, the number of groups of colours. The groups of colours are generated by using a quantisation function Q , which maps each colour in C to P , and it is defined as:

$$Q : C \rightarrow P$$

Colour-based feature extraction methods can be broadly classified into colour distribution and colour spatial methods based on whether they capture only colour distribution or the spatial relationships of colours. The colour distribution methods are further classified into three classes i.e. globally quantised colours, locally quantised colours and no quantisation, depending on whether they use any quantisation function, and if they do, the nature of the quantised colours.

The second main category of feature extraction methods is colour spatial methods. Unlike the colour distribution methods, which capture only the distribution of colours, these methods capture the spatial relationships of colours. These methods can be further divided into two types i.e. pixel-based or block-based depending on whether they operate at pixel level or groups of pixels (block-based) level. Pixel-based methods can be further classified into two main classes: pixel classification and spatial descriptors based on how the spatial information among colours are captured. The taxonomy of colour-based feature extraction methods can be found in Fig. 2.2 on the following page. The following sections describe all of these methods.

2.1.1 Colour Distribution

Colour distribution methods capture the distribution of colours in an image. The methods in this category are further classified into three classes: globally quantised

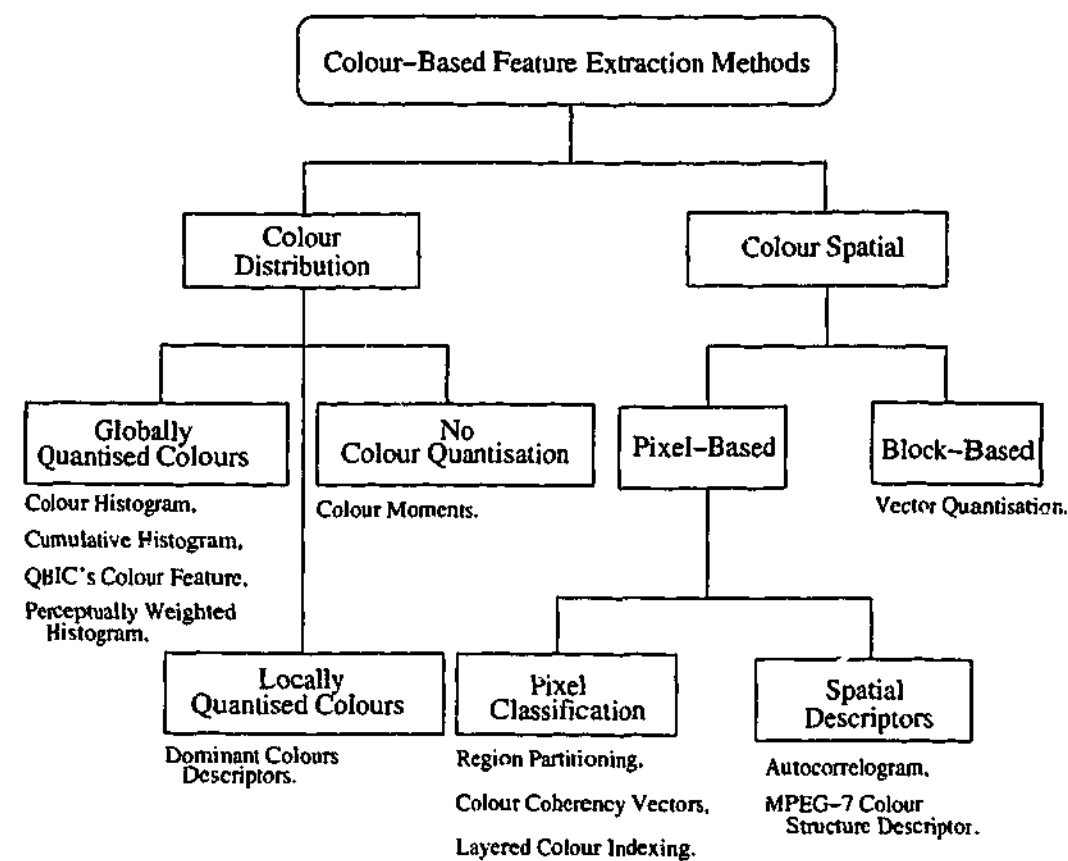


Figure 2.2: Taxonomy of colour-based feature extraction methods.

colours, locally quantised colours and no quantisation. The methods using globally quantised colours use the same quantised colours to describe the colour distribution of any image. Examples of such methods are colour histogram, cumulative histogram, QBIC's colour feature extraction and perceptually weighted histogram. In contrast, methods using locally quantised colours generate a new set of quantised colours for each image, and examples of these methods include variations of dominant colour descriptors. At the other extreme of the colour quantisation spectrum, the last type of colour distribution methods does not use any quantised colours. An example of this method is colour moment.

All colour distribution methods generate feature vectors which are rotation and translation invariant; however, only methods using globally quantised colours and methods using no quantised colours can generate the feature vectors more efficiently.

2.1.1.1 Globally Quantised Colours

This section describes four feature extraction methods using globally quantised colours: colour histogram, cumulative histogram, QBIC's colour feature and perceptually weighted histogram.

Colour Histogram

In the early nineties, Swain and Ballard proposed an algorithm which recognises objects by measuring the colour distribution of the image in which the object is present [141]. This algorithm is known as colour histogram, which is simply a count of pixels of the quantised colours in the image. The histogram feature vector for an image I is (h_1^I, \dots, h_M^I) where h_i is the count of pixels for bin i and M is the number of bins. The distance between feature vectors for image Q and I is calculated using histogram intersection:

$$d_H = 1 - \frac{\sum_{i=1}^M \min(h_i^Q, h_i^I)}{\sum_{i=1}^M h_i^I} \quad (2.1)$$

If the total count of both histograms are equal, or normalised to one, then d_H is equivalent to the $L1$ dissimilarity metric [142]:

$$d_{L1} = \sum_{i=1}^M |h_i^Q - h_i^I| \quad (2.2)$$

Colour histogram has since been extended from object recognition to image recognition. The main weaknesses of the simple histogram are that:

1. it is highly sensitive to the number of quantised colours;
2. it ignores the contribution of colours in the neighbouring bins which could be of perceptually similar colours; and
3. it ignores the spatial relationships of colours.

Several methods have been proposed to solve these problems. Stricker and Orengo overcame the first problem by proposing the cumulative histogram method with which they claimed to have reduced the effect of quantisation intervals [139]. Faloutsos et al. [34] and Lu and Phillips [67] addressed the second problem by considering the contribution of pixels in the neighbouring bins. The last problem was overcome by methods which incorporate colours spatial relationships, and these methods belong with the colour spatial methods (see Fig. 2.2). This section discusses the methods which attempt to solve the first two problems, and discussions on the methods for overcoming the last problem will be covered in Section 2.1.2.

Cumulative Histogram

The cumulative histogram method first constructs a histogram feature vector, then it accumulates the value from the previous bin to the next bin [139]. This process can be illustrated with the following simple example. Suppose all colours are quantised into four bins and the histogram feature vector of an image is normalised to one: (0.25, 0.75, 0, 0). The cumulative histogram for the same feature vector is derived by adding the value of bin 0 to bin 1, and then from bin 1 to bin 2, so the cumulative histogram for the same image is now (0.25, 1, 1, 1). The $L1$ dissimilarity metric used for colour histogram can also be used for cumulative histogram.

QBIC's Colour Feature Extraction

The QBIC's colour feature method quantises the colour space into M number of bins using agglomerative clustering. Then, a representative colour is chosen for each bin [34]. Finally, a histogram is constructed by counting the number of pixels closest to each representative colour. To also include the contribution of pixels in neighbouring bins, a correlation matrix is constructed for the quantised colour space. This correlation matrix is supposed to capture the colours correlation of perceptual similarity among

different bins. The difference between feature vectors for image Q and I is defined as:

$$d^2(Q, I) = (Q - I)A(Q - I) = \sum_{j \in M} \sum_{i \in M} a_{i,j} (h_i^Q - h_i^I)(h_j^Q - h_j^I) \quad (2.3)$$

where M is the number of bins and the dissimilarity matrix A contains $a_{i,j}$ entries, which describe the perceptual similarity of colour i and colour j .

This method is less efficient than the traditional histogram method for two reasons. First, during feature extraction, it has to find the closest representative colour for each pixel. Second, the use of a correlation matrix A increases the computational cost during retrieval. It is unclear if the additional processing time is justified, as there has been no comparative study.

Perceptually Weighted Histogram (PWH)

PWH is another feature vector which considers the contribution of pixels in neighbouring bins [67]. Like the colour feature in QBIC, it finds M representative colours by clustering the colour space. The difference between PWH and QBIC is in feature extraction. PWH finds ten closest bins to a pixel and calculates the pixel's colour distance to each representative colour of the ten bins. It then assigns the weights to each bin in inverse proportion to the colour distance. The weight for bin i is calculated as:

$$w_i = \frac{1/d_i}{1/d_1 + 1/d_2 + \dots + 1/d_{10}} \quad (2.4)$$

where d_i is the colour distance of the pixel to the i^{th} closest representative colour.

The distance between two feature vectors is calculated using the $L1$ dissimilarity metric. During feature extraction, PWH is more computationally expensive compared to QBIC's colour feature; however, it is more efficient during retrieval, as the distance between two bins is calculated only once. In QBIC, to consider the contribution of neighbouring bins, the calculations between two bins takes place more than once, and the number of calculation is dictated by the size of the correlation matrix.

2.1.1.2 Locally Quantised Colours

It is argued that using globally quantised colours is inflexible as the retrieval effectiveness is dictated by the quantisation intervals granularity [154]. A coarse quantisation has lower effectiveness than fine quantisation but it is more efficient; however, increasing the number of quantisation intervals does not necessarily guarantee higher effectiveness. The level of quantisation is a complex issue, as it is not only an issue of balancing effectiveness and efficiency but there also seems to exist an optimum number of quantisation intervals. This issue receives little attention within the CBIR community, and it is an objective to be studied in Chapter 3.

To overcome the inflexibility of global colour quantisation, dominant colour feature extraction methods generate a new set of quantised colours for each image. The dominant colour feature vector for an image is $(c_1, h_1, \dots, c_M, h_M)$, where c_i is dominant colour i , h_i is the histogram of dominant colour i , and M is the number of dominant colours (the value of M is variable). A quantised colour is considered dominant if its number of pixels exceed a predefined threshold. Each feature vector not only has its own dominant colours but also a different number of dominant colours. For these reasons, the similarity or dissimilarity metrics need to (1) consider the dissimilarity of dominant colours including their histograms and (2) deal with feature vectors with different number of dominant colours. In fact, the main difference between dominant colour feature extraction methods lies in the types of metrics.

Rubner et al. used Earth Mover's Distance (EMD) as their dissimilarity metric [105, 117, 118, 119]. This dissimilarity metric will be used in Chapters 6 and 8, so it is discussed here in detail. The EMD is based on a problem in operations research more commonly known as the transportation problem, in which the goal is to optimise the cost of transporting goods from a set of sources to a set of destinations. The amount of goods and routes for transporting the goods is defined by the flow $\mathcal{F} = [f_{ij}]$, where f_{ij} is the amount of goods to be transported from source i to destination j . To use EMD for image retrieval, Rubner et al. redefined the cost of *transporting* goods from sources to destinations into *transforming* one feature vector to another [119]. To calculate the

distance between two adaptive feature vectors \mathcal{Q} and \mathcal{I} , the optimal \mathcal{F} for transforming \mathcal{Q} into \mathcal{I} is found. Once the optimal \mathcal{F} is found, the distance between the two feature vectors is calculated using EMD:

$$d_{EMD}(\mathcal{Q}, \mathcal{I}) = \frac{\sum_{i=1}^{M^Q} \sum_{j=1}^{M^I} d_{ij} \times f_{ij}}{\sum_{i=1}^{M^Q} \sum_{j=1}^{M^I} f_{ij}} \quad (2.5)$$

where d_{ij} is the distance between colours i and j , f_{ij} is the amount of colours to be transformed from colours i to j , M^Q is the number of dominant colours in \mathcal{Q} and M^I is the number of dominant colours in \mathcal{I} . The d_{ij} is also known as the ground distance. Rubner showed how EMD could be used for other feature vectors, such as texture feature vectors, by simply changing the ground distance [118], thus demonstrating the flexibility of this metric. In fact, in Chapter 8, we use EMD to measure the distance between two texture feature vectors by simply changing the ground distance.

Another approach to calculating the distance between two dominant colour feature vectors is by using a weighted correlation w_{ij} of colours i and j in the dissimilarity metric. Variations of this dissimilarity metric can be found in the work of Leow and Li, Kankahalli et al. and Ohm et al. [52, 58, 97].

Dominant colour feature vectors have the advantage of having a smaller size compared to feature vectors using globally quantised colours, as these feature vectors describe only about eight to nine colours. The smaller size, however, comes at the cost of effectiveness: it is even less effective than the traditional histograms [58, 113, 119]. The solution to this problem would be to increase the number of dominant colours in each image, but unfortunately, doing so will increase the computational cost during feature extraction, as the clustering process is computationally expensive [117]. It also increases the computational cost during retrieval because even the most efficient metric is computationally expensive; for example, the complexity of Leow and Li's distance metric is $O(M^Q \times M^I)$, where M^Q and M^I are the number of dominant colours in feature vectors \mathcal{Q} and \mathcal{I} respectively.

2.1.1.3 No Colour Quantisation

All colour feature extraction methods described above require colour quantisation. Colour moments is a feature extraction method which requires no colour quantisation. This method will be used to generate feature vectors in Chapter 6, so it will be described in detail here. Stricker and Orengo proposed colour moments as a means of capturing the statistical contents of the colours in an image [139]. The colour moments are the average, the standard deviation and the skewness of colours in an image and they are defined as:

$$\mu_i = \frac{1}{N} \sum_{j=1}^N p_{ij}, \sigma_i = \left[\frac{1}{N} \sum_{j=1}^N (p_{ij} - \mu_i)^2 \right]^{\frac{1}{2}}, s_i = \left[\frac{1}{N} \sum_{j=1}^N (p_{ij} - \mu_i)^3 \right]^{\frac{1}{3}} \quad (2.6)$$

where i is the colour channel, j is the pixel number and N is the total number of pixels in the image. So p_{ij} is the value of pixels number j for channel i . They used HSV colour space, so there are three colour channels: H , S and V ; therefore, the feature vector of an image is $(\mu_H, \sigma_H, s_H, \mu_S, \sigma_S, s_S, \mu_V, \sigma_V, s_V)$. The distance between two feature vectors \mathcal{Q} and \mathcal{I} is defined as:

$$d_{mom} = \sum_{i=1}^C w_{i\mu} |\mu_i^{\mathcal{Q}} - \mu_i^{\mathcal{I}}| + w_{i\sigma} |\sigma_i^{\mathcal{Q}} - \sigma_i^{\mathcal{I}}| + w_{is} |s_i^{\mathcal{Q}} - s_i^{\mathcal{I}}| \quad (2.7)$$

where $w_{il} \geq 0$ is the weight for channel i and statistical measure l (either μ , σ or s), and C is the number of colour channels, which is normally 3. Stricker and Orengo proposed three different sets of w_{il} but found their effect on retrieval effectiveness is negligible.

One set of the recommended w_{il} is:

	H	S	V
μ	1	2	1
σ	1	2	1
s	1	2	1

Based on the table above, the weight for channel H and statistical measure μ ($w_{H\mu}$) is 1, and the weight for channel S and statistical measure μ ($w_{S\mu}$) is 2.

Compared to dominant colour feature vectors, the size of colour moments feature vectors is even smaller. The main weakness of this method is that it extracts information from each colour channel separately, and as a consequence, perceptually different colours can have exactly the same feature vectors. This defect, of course, has a negative impact on retrieval effectiveness. Colours are described by combinations of at least three colour channels, but by describing each channel separately, colour moments treats the colours as though they can be described independently. Figure 2.3 illustrates this problem. It shows three bi-colour images, including their HSV values and their colour moment feature vectors in HSV colour space. Of these three images, image (b) is visually most different from (a) but they both have exactly the same colour moments feature vectors. In contrast, image (c) is visually most similar to image (a) but they have different feature vectors. If we use these feature vectors to retrieve one image most similar to (a), then (b), which is least similar to (a), will be retrieved. This is why extracting the information from each colour channel independently has a negative impact on retrieval effectiveness.

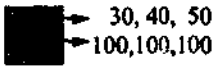
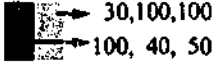
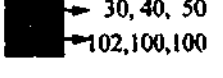
Image	H S V			Colour Moment Feature Vectors								
	μ_H	σ_H	s_H	μ_S	σ_S	s_S	μ_V	σ_V	s_V			
(a) 	65	35	0	70	30	0	75	25	0			
(b) 	65	35	0	70	30	0	75	25	0			
(c) 	66	36	0	70	30	0	75	25	0			

Figure 2.3: The main weakness of colour moments feature extraction. The arrows point to the HSV values of each colour in each image. Image (b) is visually most different to (a) but their feature vectors are exactly the same. On the other hand, image (c) is visually most similar to (a) but they have different feature vectors.

2.1.2 Spatial Relationships of Colours

All feature extraction methods discussed above can only extract information about the distributions of colours (that is, the *quantity* of colours) in an image, not the spatial relationships between colours. As a result, an image where the pixels have been

scrambled will have exactly the same feature vectors as the original even though they are visually very different. As an example, Fig. 2.4 shows three bi-colour images. The ranking of the other two images in terms of visual similarity to image (a) should be (b) followed by (c); however, by using histogram feature vectors, it is impossible to differentiate image (c) from (b) because the quantity of each colour in each image is exactly the same. Although the example uses the colour histogram method, the use of any other methods described earlier will result in exactly the same ranking. This problem can be rectified by incorporating the spatial relationships of colours.

Query Image			
	colour	histogram	
(a)	red	0.25	
	yellow	0.75	
Other Images			
	colour	histogram	L1
(b)	red	0.25	0.0
	yellow	0.75	
(c)	red	0.25	0.0
	yellow	0.75	

Figure 2.4: These three images are perceptually very different but they have exactly the same histograms.

The methods which incorporate spatial relationships of colours can be broadly categorised into two classes: pixel-based and block-based. Pixel-based methods operate at the pixel level, while block-based methods operate at the block-of-pixels level. This section first describes the pixel-based methods, then discusses the block-based methods. The pixel-based methods are further divided into pixel classification and spatial descriptors, depending on how they capture the spatial relationships of colours.

2.1.2.1 Pixel-Based: Pixel Classification

The feature extraction methods which derive spatial relationships using pixel classification perform two operations. First, they classify every pixel in the image into different categories using either one of the two following classification criteria. The first classification criterion is the location of the pixel in the image. The use of this criterion implies that the image is partitioned into sub-images (regions) and each region constitutes a category. The methods using this criterion are collectively known as region

partitioning. The second pixel classification criterion is based on the coherency of the quantised colour a pixel belongs to. The methods using this criterion include colour coherency vectors and Spectrally Layered Colour Indexing (LCI).

After classifying the pixels into different categories, these methods then extract features from each category to form the feature vector $F = (f_1, f_2, \dots, f_n)$, where f_i is the feature vector for category i , and n is the number of categories; f generally includes one of the colour distribution methods described earlier. By describing the colour distribution of each category separately, these methods thus capture the spatial relationships of colours. The main drawback of these methods is that they are highly inefficient because the size of their feature vectors is large i.e. $n \times \text{sizeof}(f)$.

The three methods using pixel classification are described in the following sections.

Region Partitioning

The simplest way to incorporate the spatial relationships of colours is by partitioning an image into sub-images (regions) [130, 138], so each region constitutes one category. Figure 2.5 shows several methods of partitioning, and each region could be either overlapping or non-overlapping. For non-overlapping partitioning, a pixel can only belong to one category whilst for overlapping partitioning, a pixel can belong to more than one category.



Figure 2.5: Several methods of partitioning an image.

Smith and Natsev partitioned an image into 16 non-overlapping regions [130], thereby creating 16 categories. The feature vector of this method is $F_{\text{region}} = (f_1, \dots, f_{16})$, and f_i is the colour histogram, texture and edges of pixels in category i . Stricker and Dimai partitioned an image into five overlapping regions, and the importance of each region is indicated by its weight - the region in the centre has the highest weight [138].

The feature vector of this method is $F_{region1} = (f_1, \dots, f_5)$, and to reduce the size of feature vector F , they used colour moments for f . Feature vectors of overlapping regions are relatively more robust when subject to translation, rotation or both than those of non-overlapping regions. The level of robustness depends on the degree of overlapping, and it is most robust with a 100% overlapping; however, such a high degree of overlapping makes it equivalent to the colour distribution methods.

Colour Coherent Vectors (CCV)

CCV was proposed based on the observation that the spatial relationships of pixels are either coherent (clustered together) or incoherent (spread far apart) [101]. A pixel is classified as coherent if it is part of a region, which is a group of similar-coloured pixels of which the number exceeds a predefined threshold. The number of categories in CCV is two i.e. coherent and incoherent, and $F_{CCV} = (f_1, f_2)$, where f_1 is the histogram for coherent pixels and f_2 is the histogram for incoherent pixels.

Spectrally Layered Colour Indexing (LCI)

The two methods described above classify pixels in the spatial domain. Qiu and Lam proposed LCI, which classifies pixels in the frequency domain [107]. In the spatial domain, the measurement revolves around the *value* of a particular pixel at column x and row y in an image but in the frequency domain, it revolves around the *frequency* of pixels in an image. An image is said to have a high frequency if the pixel values change rapidly and it has a low frequency if the pixel values change slowly, as seen in Fig. 2.6.

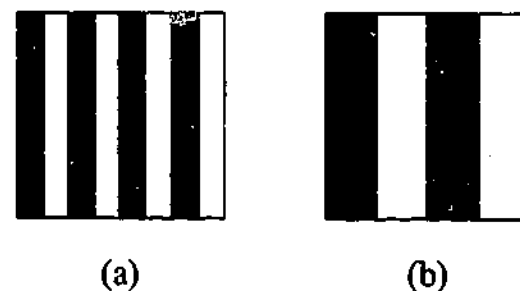


Figure 2.6: The pixel values of image (a) in the vertical direction change more frequently than that of image (b), so image (a) is said to have a higher frequency than image (b).

The frequency domain is often used in digital signal processing as it allows one to perform frequency analysis or to analyse the physical properties of a signal. An image can be viewed as a series of digital signals where the rate of change of the signal is indicated by the rate of change of the pixel values; therefore, a high frequency signal can be associated with a high variation in pixels which may indicate object boundaries, while a low frequency signal can be associated with a smooth area. By classifying pixels into different frequencies, LCI effectively separate coherent pixels from less coherent ones. LCI classifies the pixels into four levels of frequency, so the feature vector $F_{LCI} = (f_1, \dots, f_4)$, where f_i is the histogram of colours at frequency level i .

2.1.2.2 Pixel Based: Spatial Descriptors

All colour spatial feature extraction methods described earlier are based on pixel classification. As mentioned before, these methods are highly inefficient because the size of the feature vector $F = (f_1, \dots, f_n)$ is large i.e. $n \times \text{sizeof}(f_i)$, where n is the number of categories and f_i is the feature vector for category i . Spatial descriptors are much more efficient because they capture the spatial relationships of colours by calculating numerical values to capture the coherency of colours in the image. The methods which use spatial descriptors include colour autocorrelogram and MPEG-7 Colour Structure Descriptor (CSD).

Colour Autocorrelogram

Huang et al. propose using colour autocorrelogram to describe the spatial relationships of colours in images [49, 48]. Among all feature extraction methods which incorporate spatial relationships, colour autocorrelogram is the most well known and is cited more frequently in the CBIR literature than any of the other methods [106, 107, 121, 124, 130, 145]. We found a weakness with autocorrelogram which adversely affects its retrieval effectiveness, and propose a method to improve its effectiveness, and to a certain extent, its efficiency. The proposed method is known as I-autocorrelogram

(I-auto) and will be described in Chapter 4.

MPEG-7 Colour Structure Descriptor (CSD)

The CSD method is defined in the MPEG-7 standard. To capture the spatial relationships of colours, CSD uses a histogram-like method as follows. An 8×8 pixel mask visits each pixel in the image at least once and when a particular colour appears in the mask, the count for the corresponding bin is incremented by one. After incrementing the count of the corresponding bins within the mask, the mask moves by one pixel. The spatial relationships of colours are described by the histogram because given the same number of pixels, coherent colours will have higher histogram count than incoherent ones. CSD will be described in more detail in Chapter 4 where it is compared with I-auto.

2.1.2.3 Block-based: Vector Quantisation (VQ)

The feature extraction methods described above operate at pixel level, while VQ operates at group of pixels (block) level. The VQ methods are, in principle, most similar to colour histogram but instead of counting the occurrence of each quantised colour in an image, it counts the occurrence of each block pattern in an image. The block patterns are stored in a codebook. The first step in using VQ is, therefore, the generation of a codebook; an entry in the codebook stores the pattern of a block of pixels and the spatial relationships of colours are captured within the block. The VQ feature vector is defined as (h_1, \dots, h_M) , where h_i is the histogram count of block pattern i and M is the size of the codebook.

The use of VQ for extracting features and retrieving images was first proposed by Idris and Panchanathan [50] for grey level images, and later, by Teng and Lu, Qiu, and Zhu for colour images [106, 149, 163]. Zhu treated each entry in the codebook as a keyblock and extended the spatial relationships of the keyblock to two blocks (bi-block) and three blocks (tri-block) hoping that the feature vector could capture more spatial information, and consequently, increase the overall retrieval effectiveness. Surprisingly,

the use of bi-block and tri-block did not result in increased effectiveness [163].

The main shortcoming of VQ methods is that the codebook is database dependent, so a codebook must be generated for every image database. Also, a new codebook needs to be regenerated as new images are added to the database. Apart from being database dependent, the size of the feature vector based on VQ is large. Zhu reported that retrieval effectiveness is highly dependent on the quality of the codebook, which is proportional to the number of codewords [163]. Unfortunately, a good quality codebook requires large number of codewords - the minimum recommended size of a codebook is 1024 codewords [149]. These two problems (database dependence and large feature vector size) may make the use of VQ methods for real world applications less practical.

2.2 Texture-Based Methods

The previous sections discuss feature extraction methods which extract colour features from colour images. This section covers methods which extract texture features from homogeneous texture images.

Texture is hard to define but we often know what it is when we see it: the pattern of fabrics, barks, grass and sand. It is important to accurately define what texture is because the definition dictates the type of information to be extracted. Unfortunately, there is no universally accepted definition for texture except to say that it is the repetition of perceptually similar patterns over a region [98]. The type of information to be extracted was eventually determined in several psychophysics experiments, and it includes repetitiveness (periodicity), directionality, granularity, complexity, coarseness, contrast, busyness and texture strength [34, 39]. A texture descriptor therefore must capture some, if not all, of these features, and ideally, be rotation and scale invariant.

Extraction of texture features was initially restricted to grey-scale texture images, and later, some researchers extracted colour texture features from colour texture images [79, 80, 106]. When colours are involved, the measure of similarity depends on who makes the judgement. In the textile industry, textile designers consider similar-

ity of textures to be more important than similarity of colours; on the other hand, the novice considers similarity of colours is more important [37]. Feature extraction methods found in the literature include coloured pattern appearance model, fractal dimensions, Wold model, Gabor filters and wavelet filters [66, 71, 73, 79, 80, 104, 105, 117, 118, 151, 155, 160, 163].

Although texture feature extraction is not the focus of this research, in Chapter 8, we will describe Gabor filters and illustrate how this method can be used to generate a layout suitable for browsing. The description of texture feature extraction is now complete, and the next section covers methods which extract shape features from shape images.

2.3 Shape-Based Methods

Methods which extract shape features normally work on bi-level images, that is, images which have only black and white pixels, so images which are not already in black and white require some preprocessing. Shape images can be broadly classified into two types: contour and region. Contour shapes have no information within the boundary but region shapes do (see Fig. 2.7). Methods for extracting shape features, similarly, can be classified into contour and region methods; however, some contour methods are generic enough and can also be used to extract meaningful features from region shapes.

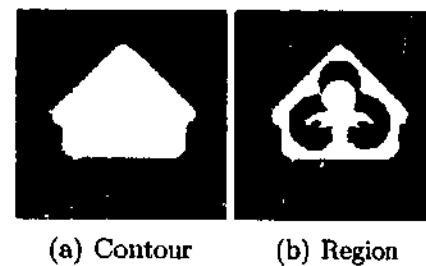


Figure 2.7: Two types of shapes. Contour shapes have no information within the boundary but region shapes do.

For contour shapes, Fourier descriptors are most popular. To use Fourier descriptors, it is necessary to first extract the contour shapes' properties, which could be the position of the boundary from its centroid, distance of the boundary from its centroid,

chord length, cumulative angular function, curvature signature and area. To achieve rotational invariance, the extracted properties are then transformed into the frequency domain using one dimensional discrete Fourier transform (hence, the name Fourier descriptors):

$$a_n = \frac{1}{N} \sum_{t=0}^{N-1} u(t) e^{(-j2\pi t)/N}, \quad n = 0, 1, \dots, N-1 \quad (2.8)$$

where $u(t)$ is the value of the measurement sampled at t and N is the number of samples taken. If the shape is described using distance of boundaries from the centroid, then $u(t)$ is the distance sampled at t . Traditional Fourier descriptors are sensitive to affine transformation i.e. transformation where parallel lines remain parallel, but contemporary Fourier descriptors are insensitive to affine transformation [4, 5].

Other contour shape descriptors include contour scale space descriptor (CSSD), anglogram, grid descriptor and generic Fourier descriptor (GFD) [7, 69, 86, 87, 144, 161]. The main disadvantage of CSSD is that it cannot capture shallow concavities of shapes correctly, and as a result, two shapes with very different concavities are considered similar. The enhanced CSSD is more sensitive to concavities, and therefore, can represent the shapes more faithfully [3].

Capturing the information in region shapes, the second type of shape image, is more complex than that for contour shapes because of the additional region information within the boundary. Of all contour shape methods mentioned above, only GFD and grid descriptor can capture region information [159, 161]. Other region shape feature extraction methods include geometric moment descriptor and Zernike moment descriptor [10, 31, 85, 103, 147].

2.4 Integrated Methods

The previous sections contain a survey of methods which extract colour features from general colour images, texture features from homogeneous texture images and shape features from shape images. This section discusses integrated methods, that is methods

which use combinations of these three features to describe the content of general colour images.

To use either texture or shape features for CBIR, it is better to segment the images first because texture features are meaningful only if the texture is homogeneous and shape features are useful only if the objects have been segmented. Most integrated methods in the literature automatically segment the images into regions where each region is a section of the image with uniform colours or textures or both. Others create the regions by simply partitioning the images [134]. Each region is then described by its colours, textures, shapes or combinations of these.

"Blobworld" is the first true region-based CBIR, and later, many others were proposed [13, 14, 15, 20, 22, 36, 53, 78, 127, 145, 155]. It was found that the retrieval effectiveness of region-based CBIR is high only if the image has a distinct object and if the object can be successfully segmented from the background [14, 124, 163]. Another problem facing region-based methods is the issue of over segmentation. To address this problem, Chen [20] and Wang [155] employed fuzzy matching which matches a region in the query image with multiple regions in another image, but it is unclear how effective this method is in overcoming the problem because there was no comparative study.

Because the use of either texture or shape features requires image segmentation, which is inaccurate when automated [124] and time consuming when segmented either manually or semi-manually, colour-based feature extraction methods remain popular for generating the feature vectors from general colour images.

2.5 Conclusions

In traditional image databases, each image is manually annotated and retrieval is performed by searching the annotation; however, manual annotation is expensive, time consuming and highly subjective. Content-based image retrieval (CBIR) systems automatically generate feature vectors from the content of images. Because it is impossible to accurately extract semantic information from images, current CBIR systems only

extract low level features such as colours, textures and shapes. This chapter reviewed existing feature extraction methods in the literature.

The texture or shape features extracted from general colour images will be more meaningful if the images are segmented first, but as automatic image segmentation still remains an open question, colour features remain popular for describing the content of general colour images. Because all colour-based methods require a colour space, the choice of colour space is an important decision. The focus of the next chapter is to evaluate which colour space is most suitable for colour-based CBIR. Once we have determined the most suitable colour space, it is appropriate to improve the effectiveness of colour-based feature extraction methods which incorporate spatial relationships of colours (Chapter 4).

Evaluating Suitability of Different Colour Spaces for Colour-Based CBIR

Colour is a very distinct feature of images. As discussed in the previous chapter, it is widely used for extracting features from general colour images. A colour is described in at least three co-ordinates and a collection of all colours in the co-ordinate system is called a colour space. A plethora of colour spaces exist in the colour science literature where more than ten have been reviewed [32, 33, 38, 125]. The more commonly used ones for colour-based CBIR include RGB, HSV, LUV and LAB [12, 29, 47, 99, 100, 108, 110, 129, 130, 135, 142]. Although the choice of colour space is fundamental to any colour-based feature extraction method, to date, very little research has been undertaken on evaluating the suitability of colour spaces for colour-based CBIR. The research reported in this chapter addresses that gap. The purpose of this chapter is to establish which colour space is more suitable for colour-based CBIR by using different colour spaces to extract colour features and conducting retrieval experiments. This study is important because it justifies the choice of colour space and provides insights into why some colour spaces are more suitable than others.

Previous work by Tan et al. [143] and Mathias and Conci [76] on this issue is incomplete; hence, the findings remain inconclusive. The main problem with their studies is that they used very small image databases which had only about two hundred images, so the results could be biased towards a certain colour space. To reduce the

bias, a large database (5,000 or more images of wide varieties) should be used instead. To further reduce the bias, more than one database should be used whenever possible. The research in this chapter reduces the bias by using two databases and each database has at least 5,000 images.

This chapter is divided into three main sections. It starts by describing the evaluation method and experimental design, then it identifies the characteristics and experimental parameters of the colour spaces to be evaluated. Finally, it presents the experimental results and analysis.

3.1 Evaluation Method and Experimental Design

The colour spaces evaluated in this study were RGB, LUV and LAB in Cartesian co-ordinates, as well as HSV, LUV and LAB in polar co-ordinates. To resolve which colour space is more suitable for colour-based CBIR, we first generated colour histogram feature vectors from two image databases using uniform colour quantisation. The distance between any two feature vectors were calculated using (2.2), the $L1$ dissimilarity. Then, after performing retrieval experiments, we compared the suitability of the colour spaces for colour-based CBIR using the evaluation criteria described in Section 3.1.1.

To evaluate the suitability of the colour spaces, one could use any colour-based feature extraction method as long as the same method is used throughout the experiment. In this study, the uniform colour quantisation histogram method was chosen for two reasons. First, we could gain more insights by using colour distribution methods because any differences could then be attributed to the differences of the colour spaces, and not due to any inaccurate spatial information of the method. Second, among colour distribution methods, there is no conclusive result which proves that other methods are much better than the simple uniform quantisation histogram.

The following sections discuss other issues related to the evaluation method such as the evaluation criteria, image databases and software used for retrieval.

3.1.1 Evaluation Criteria

The suitability of a colour space was established by comparing the retrieval effectiveness and efficiency of the feature vectors generated in that colour space.

3.1.1.1 Retrieval Effectiveness

The first evaluation criterion was retrieval effectiveness, which was determined using precision and recall graphs (PR graphs). A PR graph is defined as:

$$P = \frac{r}{N} \text{ and } R = \frac{r}{TR} \quad (3.1)$$

where r is the number of *relevant* images retrieved, N is the number of images retrieved, and TR is the total number of relevant images in the database. In evaluating the effectiveness of two feature extraction methods, the more effective one has a higher precision value at the same recall value; in the graphs, this curve is further away from the origin as illustrated in Fig. 3.1. The main purpose of this evaluation criterion was to establish which colour space, regardless of feature vector sizes, is most effective.

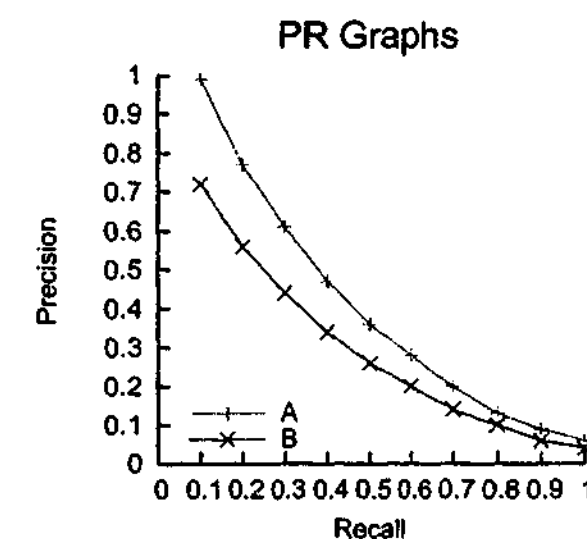


Figure 3.1: Comparing two feature extraction methods A and B using PR graphs - A is more effective than B.

3.1.1.2 Retrieval Efficiency

The second evaluation criterion was the retrieval efficiency of the colour spaces which was measured using two methods:

1. by comparing the size of feature vectors at their most effective quantisation intervals; and
2. by comparing the effectiveness of feature vectors of similar size.

The first method simply measures the feature vector size i.e. given two feature vectors, the shorter one is more efficient. In contrast, the second method measures the effectiveness of feature vectors of *similar size*. Although the second efficiency measurement compares retrieval effectiveness, it serves a different purpose than the first evaluation criterion described earlier. The second measurement was necessary because in practice, due to the issue of efficiency, we may not be able to use the most effective quantisation option. In order to find the most effective quantisation option at an acceptable feature vector size, it was essential to compare the effectiveness of the colour spaces quantised to the same size. More specifically, it was used to find a compromise between retrieval effectiveness and efficiency.

3.1.2 Image Databases

Most studies in CBIR are restricted to only one image database. Unless there is a standard database commonly used by most CBIR researchers for evaluating the effectiveness of feature vectors, more than one database should be used to reduce the bias towards any colour space. Because there is no standard database yet, retrieval experiments should be conducted using more than one database. All retrieval experiments in this chapter and Chapter 4 were tested on two databases: MPEG-7 Common Colour Database (CCD) and Proprietary Colour Database (PCD).

CCD is a database compiled by the MPEG-7 committee, and it is used widely within the MPEG-7 group [72]. It has 5466 images and established ground truths

making up the 50 Common Colour Queries (CCQ) with defined relevant images. PCD is a proprietary database of 10,112 images used within our research group, and it also has established ground truths. There are 32 queries in PCD, and the relevant images for each query were established from a subject test. In the study, 29 volunteers identified the relevant images, and these were then divided into four levels of agreement: 20%, 30%, 50% and 70% [148]. If the level of agreement is 20%, then at least 20% of the participants selected the images as relevant to the query image.

The main difference between CCD and PCD is in how the relevant images were obtained. In CCD, the relevant images for each query are often generated from a sequence of video shots and only images from the same video sequences are considered relevant. For this reason, the relevant images in CCD have no different levels of agreement. On the other hand, in PCD, the relevant images for each query were established from a subject test, and because the similarity judgement was subjective, they were divided into several levels of agreement. In short, the relevant images in PCD are more subjective compared to those of CCD.

3.1.3 Retrieval Software

A CBIR system is made up of three components: (1) building of feature vectors using feature extraction methods, (2) ranking of retrieval results using a similarity or dissimilarity metric and (3) displaying of ranked retrieval results. Research in CBIR is mostly conducted using a specially written software in which all three components are rewritten each time; hence, productive time and effort are spent on developing software irrelevant to research questions. To speed up software development, several researchers in CBIR produced open frameworks which allow component reuse; for instance, Gunther and Beretta developed BIRDS-I and Müller et al. developed GIFT [43, 44]. Both BIRDS-I and GIFT provide the second and third components of a CBIR system i.e. the ranking and display components, so experimenters only need to build feature vectors and provide a similarity or dissimilarity metric to the ranking component. The frameworks thus cut down software development time and effort.

Both GIFT and BIRDS-I run in client and server mode, where the server stores the feature vectors and performs the retrieval whilst the client queries the server and displays the ranked retrieval results. GIFT, however, is more flexible because its server can be installed locally. For this reason, all retrieval experiments in this thesis were conducted using GIFT. Note that GIFT is only a framework: experimenters must build the feature vectors, then develop and link the dissimilarity metrics to GIFT as plug-ins [43, 89, 90, 91, 92, 93]. We, therefore, implemented all the feature extraction methods and dissimilarity metrics, and linked the dissimilarity metrics to GIFT as plug-ins.

3.2 The Six Colour Spaces Evaluated

The purpose of this study was to evaluate the suitability of six commonly used colour spaces for colour-based CBIR: RGB, LUV and LAB in Cartesian co-ordinates, as well as HSV, LUV and LAl in polar co-ordinates. The following sections describe the characteristics of the six colour spaces (Section 3.2.1) and explain how they were quantised (Section 3.2.2).

3.2.1 Characteristics of Six Colour Spaces

This section describes the characteristics of the six colour spaces and their relationships with each other. It is necessary to know their characteristics because the quantisation parameters of each colour space in Section 3.2.2 depend on the characteristics of the colour space.

3.2.1.1 Characteristics of RGB

The RGB colour space is used in CRT monitors. A colour is described by the values of Red, Green and Blue. A display device capable of differentiating 256 intervals for each R, G and B channel has a colour depth of 24 bits and can show about sixteen million colours.

3.2.1.2 Characteristics of HSV

RGB is hardware oriented, and can be easily used by monitors but not by humans. HSV is a user oriented colour space proposed in 1978 [128], and it is more intuitive to use because the description of colours corresponds to how humans describe colours:

- hue (H) describes the colourness or tint - such as red, green, yellow;
- saturation (S) describes the intensity or colourfulness - less saturated colours are equivalent to adding white to water colours; and
- brightness (V for *value*) describes the darkness or brightness.

The shape of the HSV colour space is a hexcone. The H axis is in polar co-ordinates specified from 0° to 360° , S from 0 to 1, and V from 0 to 1. A cross section of HSV at $V = 1$ is the same as viewing the RGB cube from the main diagonal axis from achromatic white to black (see Fig. 3.2(a) and (b)). A cross section of HSV at $V < 1$ is a view of the RGB subcube in the same direction (see Fig. 3.2(c)). When each subcube is viewed in the same direction, the result is a single-hexcone (see Fig. 3.2(d)).

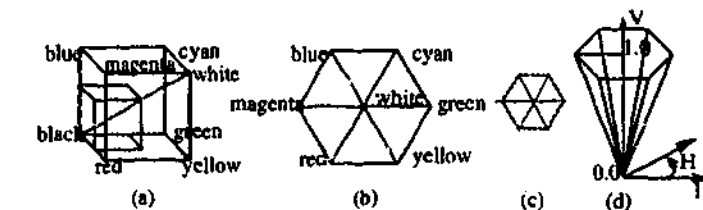


Figure 3.2: (a) RGB cube and a subcube. (b) RGB cube viewed from achromatic diagonal when $V = 1$ (c) when $V < 1$. (d) Single-hexcone HSV colour model.

Figure 3.3 shows the HSV colour space when taken at $V = 0.1$ through to 1. Although the size of the hexagon is smaller as the value of V is smaller, the range for the value of S is constant. The algorithm for transforming RGB into HSV is given by [38]:

```

max = MAX(r,g,b)
min = MIN(r,g,b)
v = max

if max != 0
    s = (max-min) / max
else
    s = 0
endif
if (s = 0)
    h = UNDEFINED
    return
endif

delta = max-min
if (r = max)
    h = (g-b)/delta
else if (g = max)
    h = 2 + (b-r)/delta
else if (b = max)
    h = 4 + (r-g)/delta
endif
h *= 60
if (h < 0)
    h += 360
endif

```

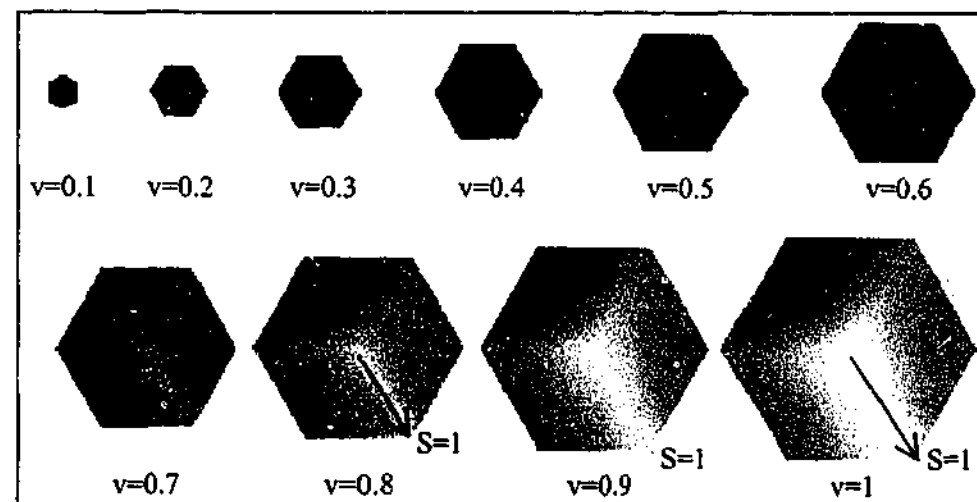


Figure 3.3: Display of HSV colour space for $V=0.1$ through to 1.

3.2.1.3 Characteristics of LUV and LAB in Cartesian Co-ordinates

In 1976, Commission Internationale de l'Eclairage (CIE) introduced LUV and LAB colour spaces. L specifies the lightness (like V in HSV), while u^* and v^* as well as a^* and b^* are the opponent colour axes: approximately red-green versus yellow-blue, that is u^* versus v^* and a^* versus b^* . In LUV, changes in u^* or v^* axes result in changes of hue and chroma. Similarly, in LAB, changes in a^* or b^* axes result in changes of hue and chroma. Chroma is similar to saturation in that it defines the intensity or colourfulness of a colour. However, there is some technical difference between chroma and saturation. Chroma is the colourfulness of a stimulus "relative to the brightness of a similarly illuminated white", whereas saturation is "the colourfulness of stimulus relative to its own brightness" [33]. In this study, this difference is subtle enough to be ignored because we are not concerned with how chroma or saturation is obtained, but rather how chroma or saturation is quantised. From this point onward, saturation also means chroma. More discussion on opponent axes colour theory can be found in [120].

It is important to note that the valid ranges of u^* and v^* fluctuate as illustrated in Fig. 3.4, which shows the LUV colour space when $L=0.1$ through to $L=1$. Consequently, the geometrical shape of the LUV colour space is irregular. The implication of this irregular shape is that during uniform quantisation in Section 3.2.2, some bins will always be empty.

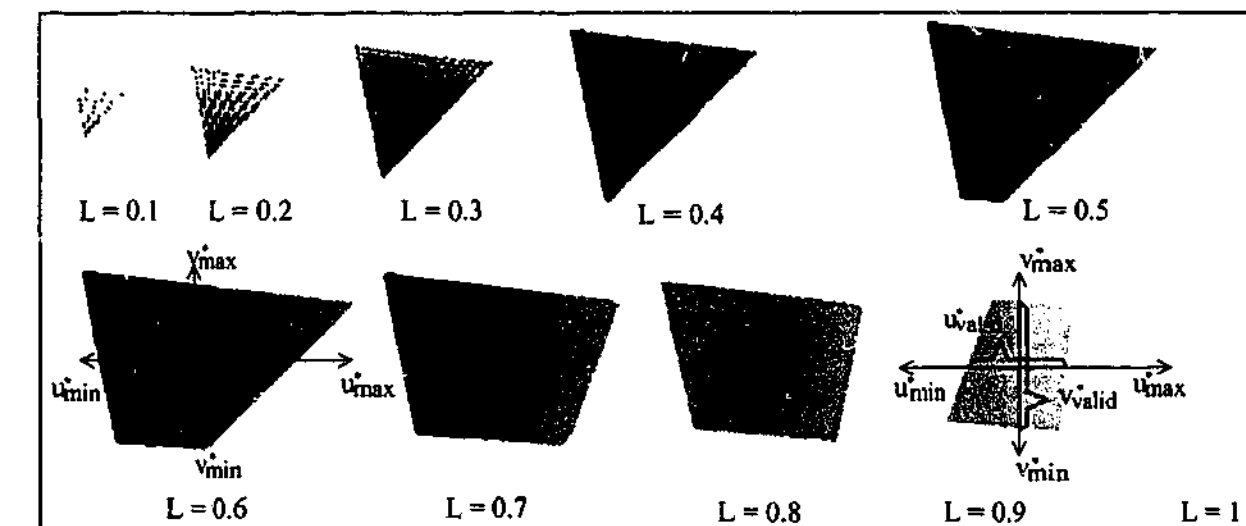


Figure 3.4: Display of LUV colour space when $L = 0.1$ through to 1. Note that the valid ranges of u^* and v^* fluctuate, so the geometrical shape of LUV is irregular.

Like LUV, the LAB colour space also has irregular geometrical shapes because the valid ranges of a^* and b^* also fluctuate (Fig. 3.5 shows the LAB colour space when $L=0.1$ through to $L=1$). So, the implication is similar as in LUV: some bins will always be empty during uniform quantisation.

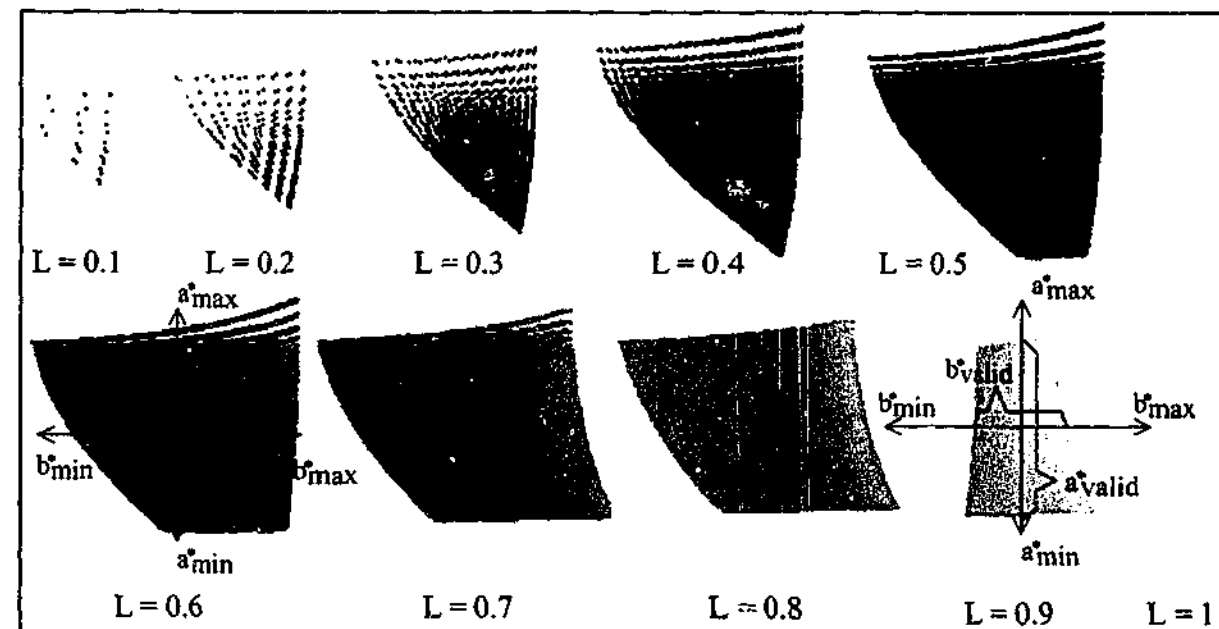


Figure 3.5: Display of LAB colour space when $L = 0.1$ through to 1. The valid ranges of a^* and b^* fluctuate, so like LUV, the geometrical shape of LAB is also irregular.

Both LUV and LAB are supposed to be perceptually uniform, that is, if the Euclidean distance between colours c_i and c_j is d , and if the Euclidean distance of colours c_j and c_k is also d , then c_i and c_j should look as different as c_j and c_k . However, it has been found that this property is true only for small colour differences [120].

3.2.1.4 Characteristics of LUV and LAB in Polar Co-ordinates (pLUV and pLAB)

The CIE also describes both LUV and LAB in perceptual properties of hue, chroma, and lightness or darkness by using polar co-ordinates [120, 156]. The transformation

of LUV into polar co-ordinates is given as:

$$H_{uv} = \tan^{-1} \left(\frac{v^*}{u^*} \right) \quad (3.2)$$

$$C_{uv} = \sqrt{u^{*2} + v^{*2}} \quad (3.3)$$

$$L_{uv} = L \quad (3.4)$$

where H_{uv} is hue, C_{uv} is chroma and L_{uv} is lightness or darkness. For convenience, LUV in the polar co-ordinates from now on will be known as pLUV, where p stands for polar. C_{uv} in this co-ordinate system is similar to saturation or S in HSV, and L_{uv} is similar to the V axis in HSV.

Unlike the transformation of RGB into HSV, where the geometrical shape of the colour space is transformed from a cube into a single hexcone, the transformation from LUV into pLUV does not change the geometrical shape of the colour space: both LUV and pLUV have the same geometrical shape. This is because (3.2) and (3.3) is a transformation in \mathbb{R}^2 from Cartesian co-ordinates into polar co-ordinates, which does not change the positions of the points being transformed. As an illustration, a simple example involving a transformation of point p in \mathbb{R}^2 from the Cartesian co-ordinates into the polar co-ordinates is given in Fig. 3.6. Because C_{uv} is directly derived from u^* and v^* , for which valid values fluctuate, valid ranges of C_{uv} also fluctuate. So, during uniform quantisation, some bins will always be empty.

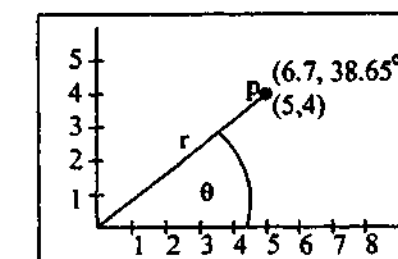


Figure 3.6: Transforming point p from the Cartesian co-ordinates (x, y) into polar co-ordinates (r, θ) . Although the values of p 's polar co-ordinates are different from those of the Cartesian, p remains at the same physical location.

The description of LAB in the perceptual properties of hue, chroma, and lightness, similarly, only requires a transformation from Cartesian co-ordinates into polar co-

ordinates [120, 156]:

$$H_{ab} = \tan^{-1} \left(\frac{b^*}{a^*} \right) \quad (3.5)$$

$$C_{ab} = \sqrt{a^{*2} + b^{*2}} \quad (3.6)$$

$$L_{ab} = L \quad (3.7)$$

For convenience, LAB in the polar co-ordinates from now on will be known as pLAB, where p stands for polar. In this transformation, the geometrical shape of pLAB remains the same as in LAB for the same reason as in pLUV. Similar to C_{uv} , C_{ab} is directly derived from a^* and b^* , and the valid ranges for both a^* and b^* fluctuate; consequently, the valid ranges for C_{ab} also fluctuate. This means during uniform quantisation some bins will always be empty.

The description of the colour spaces is now complete and the next section explains how they were quantised.

3.2.2 Parameters for Colour Space Quantisations

To perform uniform quantisation, each axis in a colour space is quantised into uniform intervals, and this results in grouping the colours into bins. To illustrate this process, Fig. 3.7 on the following page shows the quantisation of RGB where each axis is quantised into four intervals. With uniform quantisation for all colour spaces, it is *hoped* that perceptually similar colours are quantised into the same bin, and perceptually different colours are quantised into different bins. These two conditions ensure that each bin contains only perceptually similar colours, and the colours in each bin are visually distinct from other bins. The effectiveness of a colour-based CBIR system is strongly influenced by how well these two conditions are adhered to. In this study, these two conditions are useful for explaining why some colour spaces are more suitable for colour-based CBIR than others.

The number of quantisation intervals for each axis depends on the importance of an axis, which in turn, relies on the nature of the axis. Hue is the most important property

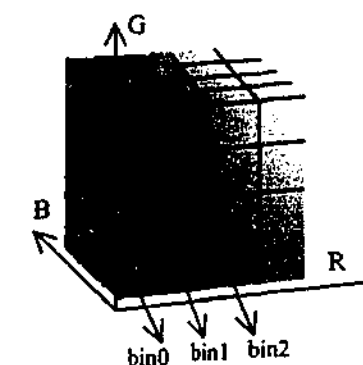


Figure 3.7: Uniform quantisation in RGB when each axis is quantised into four intervals. Each bin contains a group of colours.

of a colour because changes in hue have more effect on the perception of colours than changes in saturation or luminance; for example, people with normal colour vision would agree that yellow and blue are more different compared with yellow and light yellow. For this reason, the axis that controls hue is considered most important, and it should be quantised more finely than the axis that controls saturation or luminance. Besides this, saturation and luminance are easily affected by lights or shadows. To reduce this effect, they should be quantised more coarsely than hue.

The upper bound on the number of quantisation intervals for each axis was determined by slowly incrementing the number of quantisation intervals until there was no further noticeable difference in retrieval effectiveness. This method makes it possible to observe the effect of quantisation on each axis and to obtain the most effective quantisation option for each colour space.

3.2.2.1 Colour Space Quantisations in Cartesian Co-ordinates

Quantisation Parameters for RGB

In RGB, all three axes control the hues of colours. As a result, all axes are equally important, so they were quantised into the same number of intervals. The axes were quantised ranging from four intervals for each R , G and B axis to 28 intervals (from RGB $4 \times 4 \times 4$ to RGB $28 \times 28 \times 28$) with an increment of two, four or eight numbers of intervals, which gives a total of 64 to 21952 bins.

Quantisation Parameters for LUV

In LUV, L controls the luminance of colours while both u^* and v^* control the hue and saturation. The separation of luminance makes it possible to quantise luminance independently of hue and saturation. It was mentioned earlier that changes in hue have the largest effect on the perception of colours, and it should be quantised more finely. Thus, both u^* and v^* were quantised more finely than L . L was quantised into three intervals, while u^* and v^* were quantised from four to 40 intervals (from LUV $3 \times 4 \times 4$ to LUV $3 \times 40 \times 40$) with an increment of four or eight numbers of intervals, giving a total of 48 to 4800 bins. As mentioned earlier, the geometrical shape of LUV is irregular; hence, some bins will always be empty. Out of 48 bins, for example, only 36 bins are effective because 12 bins are always empty, and out of 4800 bins only 2167 bins are effective.

Furthermore, to study the effect of quantisation on the L axis, L was quantised into two, three, four and five intervals, while u^* and v^* were quantised into 20 intervals (LUV $2 \times 20 \times 20$ to LUV $5 \times 20 \times 20$). This approach keeps the number of PR graphs to a minimum without risking the quality of this study.

Quantisation Parameters for LAB

In LAB, L controls the luminance of colours while both a^* and b^* control the hue and saturation of colours. For the same reason as in LUV, L was quantised more coarsely than a^* and b^* . L was quantised into three intervals while a^* and b^* were quantised from four to 40 intervals (from LAB $3 \times 4 \times 4$ to LAB $3 \times 40 \times 40$) with an increment of four or eight numbers of intervals, which results in a total of 48 to 4800 bins. Recall that the geometrical shape LAB is irregular; hence, some bins will always be empty. So, out of 48 bins, only 39 bins are effective because 9 bins are always empty, and out of 4800 bins only 2410 bins are effective.

In addition, to study the effect of quantisation on the L axis, by using the same approach as in LUV, L was quantised into two, three, four and five intervals, while a^* and b^* were quantised into 20 intervals (from LAB $2 \times 20 \times 20$ to LAB $5 \times 20 \times 20$).

This keeps the number of PR graphs to a minimum without affecting the quality of the study.

3.2.2.2 Colour Space Quantisations in Polar Co-ordinates

Quantisation Parameters for HSV

As described in Section 3.2.1.2, each axis in HSV describes a different aspect of a colour. V controls the luminance, and in this regard is similar to L in LUV or LAB. However, unlike LUV or LAB, hue and saturation are described in two independent axes: H for hue and S for saturation. H was quantised from nine to 30 intervals with an increment of three or six numbers of intervals, while S and V were quantised into three intervals (from HSV $9 \times 3 \times 3$ to HSV $30 \times 3 \times 3$), giving a total of 81 to 270 bins. To study the effect of quantisation on the S and V axes, they were quantised into two, three, four and five intervals, and H was quantised into 18 intervals (from HSV $18 \times 2 \times 2$ to HSV $18 \times 5 \times 5$).

Quantisation Parameters for pLUV

For the same reason as in HSV, H_{uv} was quantised into more number of intervals than C_{uv} (chroma) and L_{uv} (luminance). H_{uv} was quantised from nine to 30 intervals with an increment of three or six numbers of intervals, while C_{uv} and L_{uv} were quantised into three intervals (from pLUV $9 \times 3 \times 3$ to pLUV $30 \times 3 \times 3$), giving a total of 81 to 270 bins. As mentioned before, the valid ranges of C_{uv} vary; therefore, some bins are always empty. Out of 81 bins, only 59 bins are effective because 22 bins are always empty, and out of 270 bins, only 175 bins are effective. Although the geometrical shapes of pLUV and LUV are the same, two of the three axes which control the quantisation process differ: in LUV, it is controlled by u^* and v^* axes whilst in pLUV, it is controlled by C_{uv} and H_{uv} axes. To illustrate this difference, Fig. 3.8 on the following page shows a slice of LUV at $L = 0.7$ and pLUV at $L_{uv} = 0.7$ when quantised uniformly (note that $L_{uv} = L$).

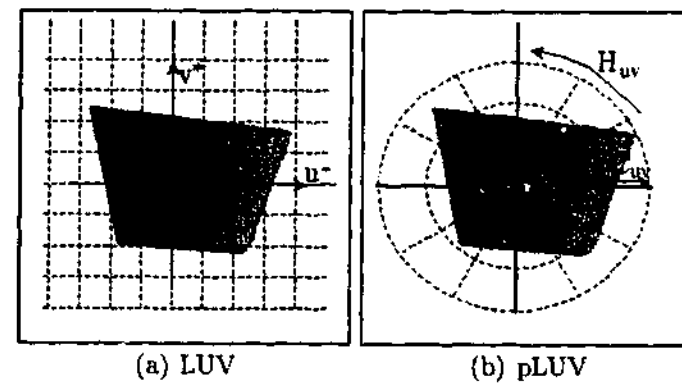


Figure 3.8: A visual description of the quantisation process in LUV and pLUV at $L=0.7$ (note that $L_{uv} = L$).

To study the effect of quantisation on C_{uv} and L_{uv} axes, they were quantised into two, three, four and five intervals, and H_{uv} was quantised into 18 intervals (pLUV $18 \times 2 \times 2$ to pLUV $18 \times 5 \times 5$). As mentioned earlier, the valid ranges of C_{uv} vary; hence, quantising the saturation axis (C_{uv}) in pLUV into three intervals is actually coarser than that (S) for HSV: the geometrical shape of LUV in Fig. 3.4 reveals more colours are quantised into lower values of C_{uv} than for higher values of C_{uv} . This suggests that pLUV $18 \times 3 \times 3$ may be less effective than HSV $18 \times 3 \times 3$. If this is true, then increasing the number of quantisation intervals of C_{uv} may also increase retrieval effectiveness. For this reason, we were interested to see the effect of quantisation on C_{uv} , so it was also quantised from five to 11 intervals with an increment of two number of intervals, while fixing H_{uv} at 18 intervals and L_{uv} at three intervals (pLUV $18 \times 5 \times 3$ to pLUV $18 \times 11 \times 3$).

Quantisation Parameters for pLAB

For the same reason as in HSV and pLUV, H_{ab} was quantised into finer intervals than C_{ab} (saturation) and L_{ab} (luminance). H_{ab} was quantised from nine intervals to 30 intervals with an increment of three or six numbers of intervals, while C_{ab} and L_{ab} were quantised into three intervals (from pLAB $9 \times 3 \times 3$ to pLAB $30 \times 3 \times 3$), giving a total of 81 to 270 bins. Because the valid ranges of C_{ab} vary, some bins are always empty. Out of 81 bins, only 64 bins are effective because 17 bins are always empty, and out of 270 bins, only 193 bins are effective.

Like pLUV and for the same reason, C_{ab} and L_{ab} axes were also quantised into two, three, four and five intervals; and H_{ab} was quantised into 18 intervals (from pLAB $18 \times 2 \times 2$ to pLAB $18 \times 5 \times 5$). Also, quantising the saturation axis (C_{ab}) in pLAB into three intervals is coarser than that (S) for HSV. Therefore, C_{ab} was quantised from five to 11 intervals with an increment of two number of intervals, while fixing H_{ab} at 18 intervals and L_{ab} at three intervals (from pLAB $18 \times 5 \times 3$ to pLAB $18 \times 11 \times 3$).

3.3 Results and Discussion

To evaluate the suitability of colour spaces for colour-based CBIR, we first generated colour histogram feature vectors from two image databases (CCD and PCD) using uniform colour quantisation. In total, twelve sets of feature vectors were built using six colour spaces, one set of feature vectors for each colour space in each database. We then conducted retrieval experiments in each database using GIFT, and calculated the distance of two feature vectors with (2.2), the $L1$ dissimilarity metric. All queries have predefined relevant images: 50 query images for CCD and 32 query images for PCD. The PR graphs for a database presented here (please see Section 3.1.1.1 for the definition of a PR graph) were generated by averaging the PR graphs from the queries in that database; that is, a PR graph in CCD is the average of 50 PR graphs and a PR graph in PCD is the average of 32 PR graphs. In PCD, because the relevant images were divided into four levels of agreement, the PR graphs for PCD were also divided into four levels of agreement i.e. PCD 20%, PCD 30%, PCD 50% and PCD 70%. Finally, we analysed the results using the two evaluation criteria described in Section 3.1.1, that is retrieval effectiveness and retrieval efficiency.

3.3.1 Retrieval Effectiveness

We first analysed the retrieval effectiveness, regardless of retrieval efficiency, of each colour space starting from those in Cartesian co-ordinates followed by those in polar co-ordinates, then compared the effectiveness of all six colour spaces.

3.3.1.1 Effectiveness of Each Colour Space in Cartesian Co-ordinates

Figure 3.9 on the following page shows the PR graphs for CCD and PCD 20%. For LUV and LAB, the number of quantisation intervals of u^* and v^* , as well as a^* and b^* varies, but the number of quantisation intervals of L axes for LUV and LAB are fixed at three. Figure 3.10 on the following page shows the PR graphs of LUV and LAB when the numbers of quantisation intervals of the L axes for LUV and LAB vary. For PCD, as the trend for all levels of agreement is similar, only the graphs from the 20% level of agreement are presented. The graphs for all levels of agreement can be found in Appendix B.1.

Effectiveness of RGB

As shown in Fig. 3.9 on the following page, it is clear that the effectiveness of RGB increases in proportion to the number of quantisation intervals, with RGB $4 \times 4 \times 4$ being the worst. It was mentioned earlier that retrieval effectiveness depends on whether perceptually similar colours are quantised into the same bin, and perceptually different colours are quantised into different bins; therefore, the retrieval trend can be explained by observing the colours in each bin from RGB $4 \times 4 \times 4$. In RGB, we found that perceptually different colours are quantised into the same bin; for example, bin 21 in Fig. 3.11(a) on the following page has orange, yellow, green, cyan, blue, purple and red. As the number of quantisation intervals increases, fewer bins have perceptually different colours; therefore, the effectiveness increases. However, up to a certain number of quantisation intervals (RGB $16 \times 16 \times 16$), no meaningful improvement is observed. In fact, at RGB $28 \times 28 \times 28$, the effectiveness starts declining. We suspect this is because perceptually similar colours are quantised into different bins.

Effectiveness of LUV

From Fig. 3.9 on the following page, we can see that the effectiveness of LUV $3 \times 4 \times 4$ is the worst. Up to a certain number of quantisation intervals, retrieval effectiveness increases as the number of quantisation intervals increases. This happened for the same

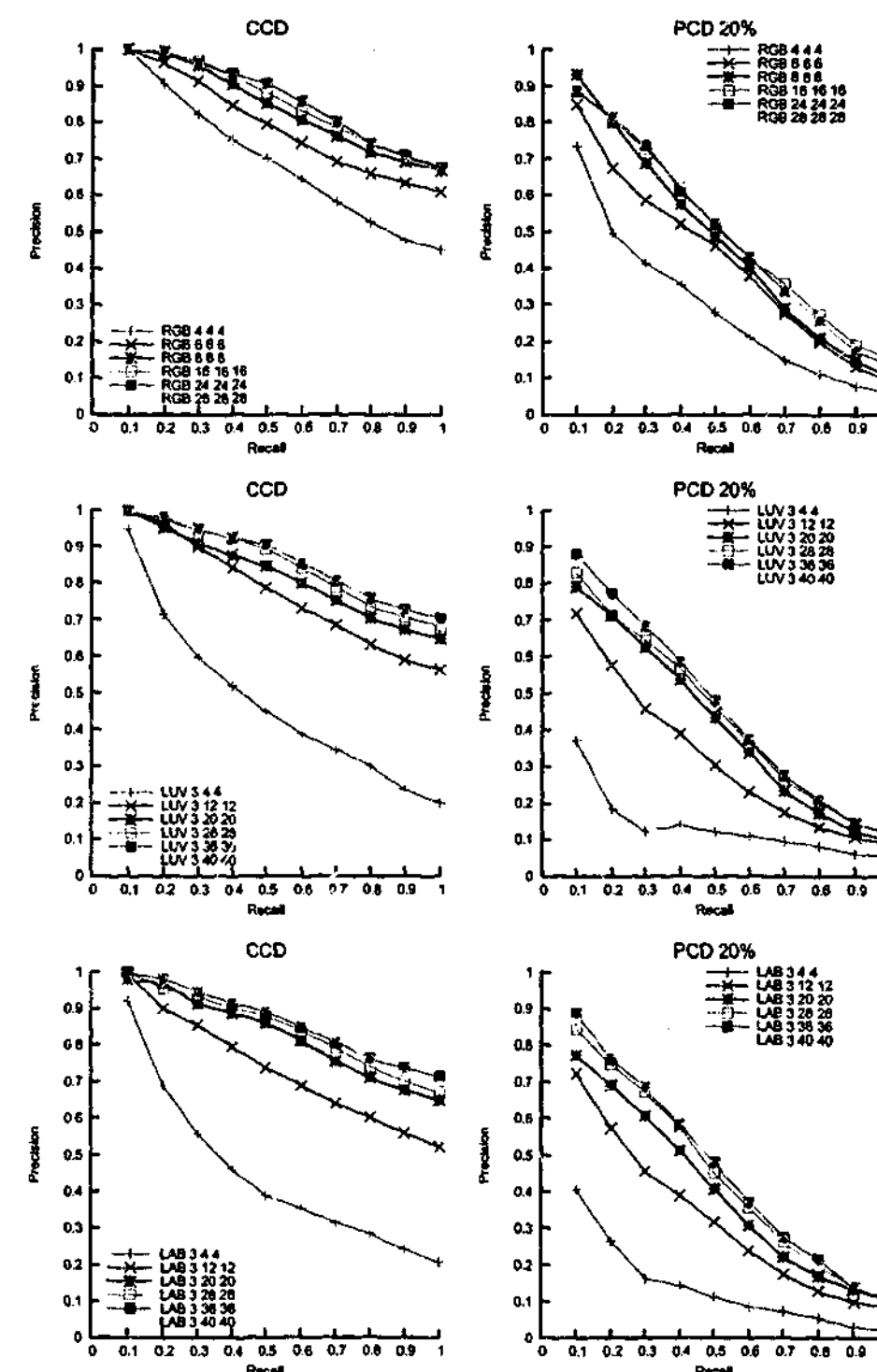


Figure 3.9: Performance of RGB, LUV and LAB colour spaces at different quantisation options in CCD and PCD.

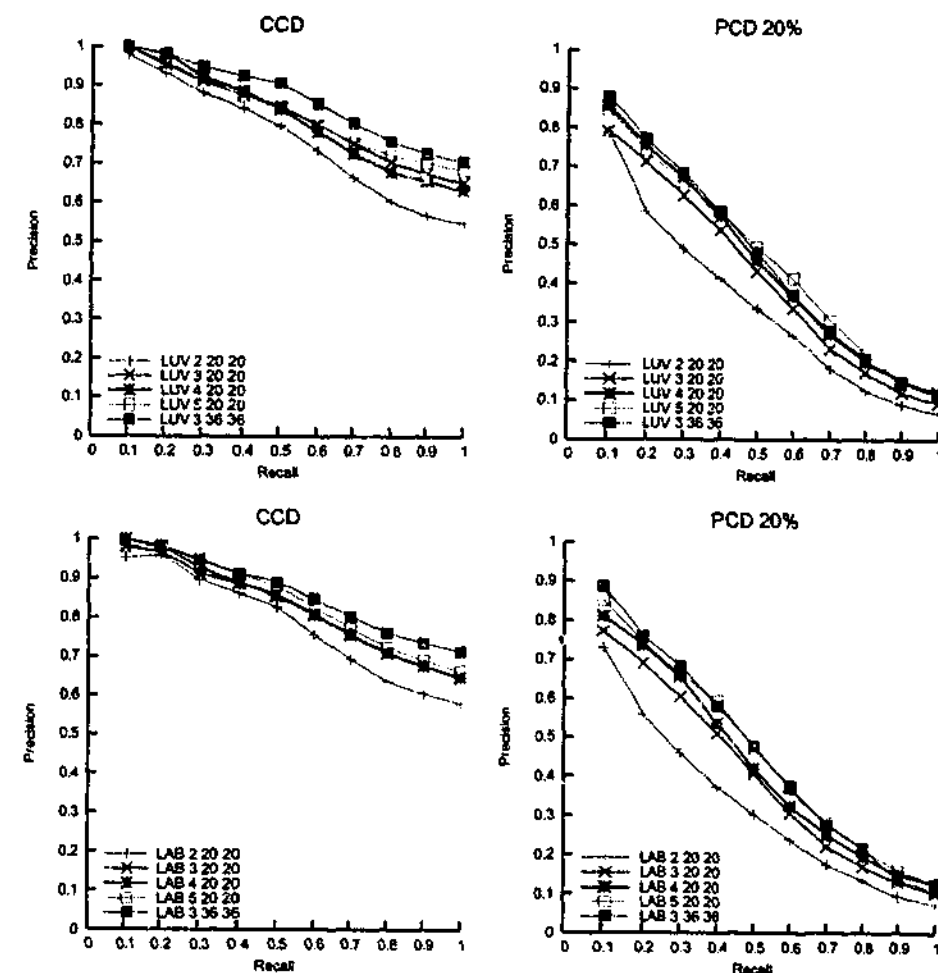


Figure 3.10: Performance of LUV and LAB colour spaces in CCD and PCD when the number of quantisation intervals of r_{uv} and L_{ab} varies.

reason as in RGB; that is, some bins have perceptually different colours. An example of this is given in bin 38 of LUV $3 \times 4 \times 4$ (see Fig. 3.11(b) on the following page). As the number of quantisation intervals increases, fewer bins have perceptually different colours; therefore, the effectiveness increases. At very fine quantisation, perceptually similar colours are quantised into different bins and the effectiveness decreases: the effectiveness starts declining at LUV $3 \times 40 \times 40$. Thus, the finding for RGB is also true for LUV in that higher effectiveness can be gained by having finer quantisation but only up to a certain number of quantisation intervals.

From Fig. 3.10, it appears that quantising L finer than three intervals has little impact on retrieval effectiveness. In CCD, incrementing the number of quantisation intervals of L from two to three results in a noticeable increase in retrieval effectiveness,

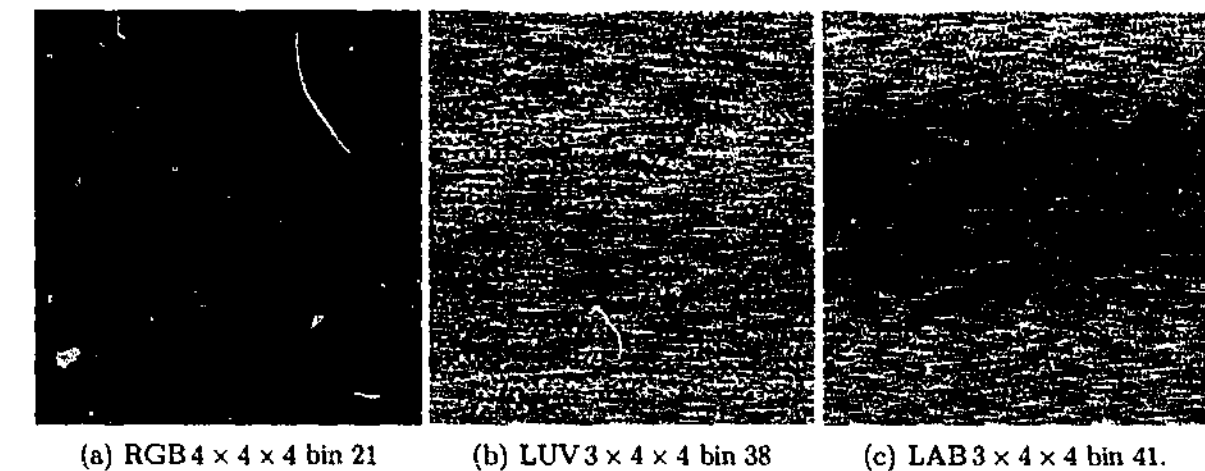


Figure 3.11: Perceptually different colours are quantised into the same bin.

and the effect of quantising L beyond three intervals is negligible. This observation confirms the statement made at the start of Section 3.2.2 that luminance only needs to be quantised coarsely. It also appears that quantising L at three intervals is optimum. Upon reaching the optimum number of quantisation intervals for L , to achieve more effective retrieval, we must increase the number of quantisation intervals of u^* and v^* axes. Hence, to achieve higher effectiveness than LUV $3 \times 20 \times 20$, instead of incrementing L , we incremented u^* and v^* from 20 to 36 intervals (from LUV $3 \times 20 \times 20$ to LUV $3 \times 36 \times 36$). As shown in Fig. 3.10, in CCD, LUV $3 \times 36 \times 36$ is more effective than LUV $3 \times 20 \times 20$ and LUV $5 \times 20 \times 20$. From Fig. 3.10, in PCD 20%, we can also see LUV $3 \times 36 \times 36$ is most effective. To sum up, the retrieval results of LUV in CCD and PCD establish that L should be quantised coarsely, and the optimum number of quantisation intervals for L is three.

Effectiveness of LAB

As shown in Fig. 3.9, the finding for RGB and LUV colour spaces is also true for LAB, that higher effectiveness can be gained from having finer quantisation, but only up to a certain number of quantisation intervals. For the same reasons as in RGB and LUV, this is because some bins have perceptually different colours. An example of this can be found in bin 41 of LUV $3 \times 4 \times 4$ (see Fig. 3.11(c)). As the number of quantisation intervals increases, fewer bins have perceptually different colours; therefore, the

effectiveness increases. However, up to a certain number of quantisation intervals, the effectiveness declines for the same reason as in RGB and LUV: the effectiveness starts declining at $LAB 3 \times 40 \times 40$.

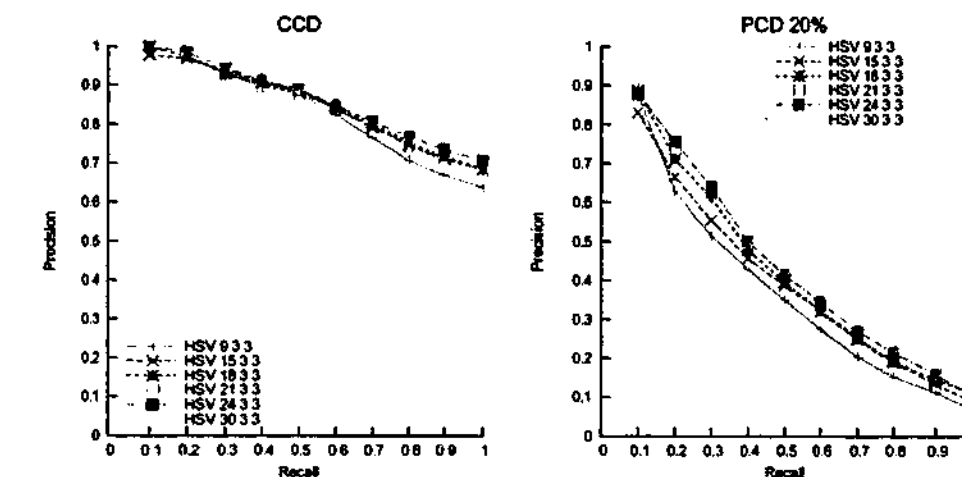
With reference to Fig. 3.10, the most effective quantisation option for LAB in CCD is the same as those for LUV in CCD, that is $3 \times 36 \times 36$ is most effective. Likewise, it appears that quantising L at three intervals is optimum. To achieve better retrieval effectiveness, we must further quantise a^* and b^* when reaching the optimum quantisation of L . For PCD in Fig. 3.10, the most effective quantisation option is the same as those for LUV in PCD, that is $3 \times 36 \times 36$ is most effective. In summary, the retrieval results of LAB in CCD and PCD suggest that L should be quantised coarsely, and the optimum number of quantisation intervals for L is three.

3.3.1.2 Effectiveness of Each Colour Space in Polar Co-ordinates

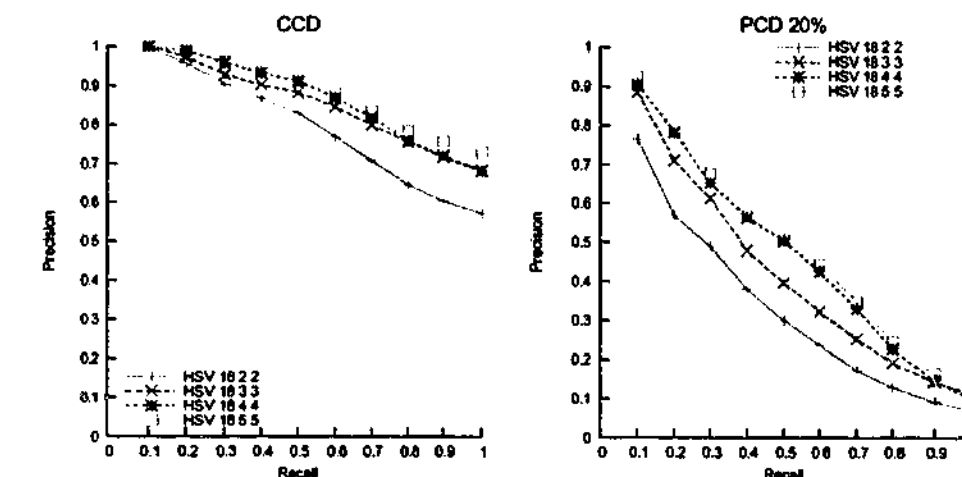
Figure 3.12 to 3.14 on the following pages show the PR graphs for the three colour spaces in polar co-ordinates HSV, pLUV and pLAB, where only the number of quantisation intervals in hue varies, and later, when the number of quantisation intervals in hue was fixed whilst the numbers of quantisation intervals in other axes vary. The graphs show that increasing the number of quantisation intervals for saturation and brightness increases the effectiveness of all colour spaces. The trend is similar in all levels of agreement in PCD, and the complete results for PCD can be found in Appendix B.2 (Fig. B.4 and B.5).

Effectiveness of HSV

From Fig. 3.12(a) on the following page, we can see that HSV $24 \times 3 \times 3$ is most effective, and the gain in effectiveness from $9 \times 3 \times 3$ to $18 \times 3 \times 3$ is most noticeable. In this respect, HSV is less sensitive to the number of quantisation intervals, and therefore more robust. As shown in Fig. 3.12(b), increasing the number of quantisation intervals of S or V increases retrieval effectiveness, and the increase is most noticeable when changing the number of quantisation intervals of S and V from two to three.



(a) Varying the number of quantisation intervals of H



(b) Varying the number of quantisation intervals of S and V

Figure 3.12: Performance of HSV quantised at different number of intervals for H , S and V in CCD and PCD.

Effectiveness of pLUV

It can be seen from Fig 3.13(a) on the following page that pLUV at $9 \times 3 \times 3$ is least effective, and increasing the number of quantisation intervals results in higher effectiveness. Increasing the number of quantisation intervals for hue affects the effectiveness of pLUV more significantly than that of HSV. From Fig. 3.13(b), it is obvious that increasing the number of quantisation intervals of C_{uv} and L_{uv} also increases retrieval effectiveness; for example, pLUV $18 \times 5 \times 5$ is more effective than pLUV $18 \times 3 \times 3$. Likewise, increasing the number of quantisation intervals of C_{uv} from three to five (from pLUV $18 \times 3 \times 3$ to pLUV $18 \times 5 \times 3$) also increases retrieval effectiveness.

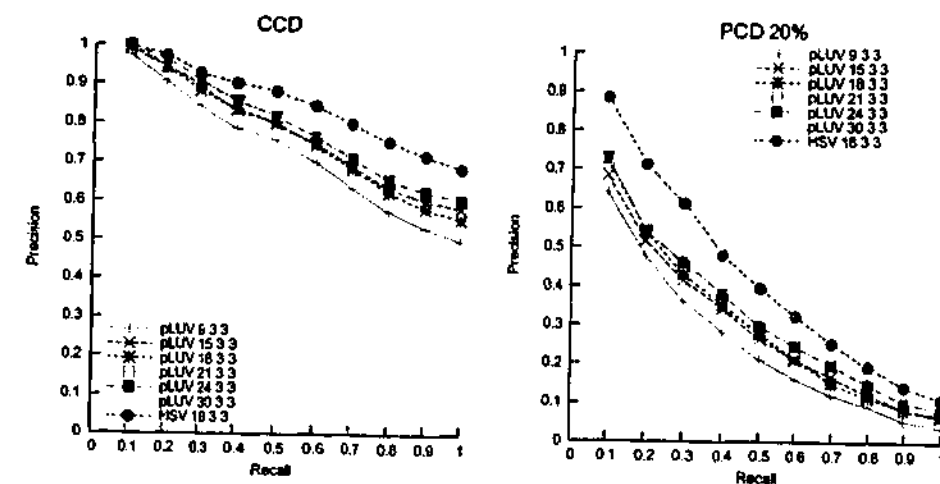
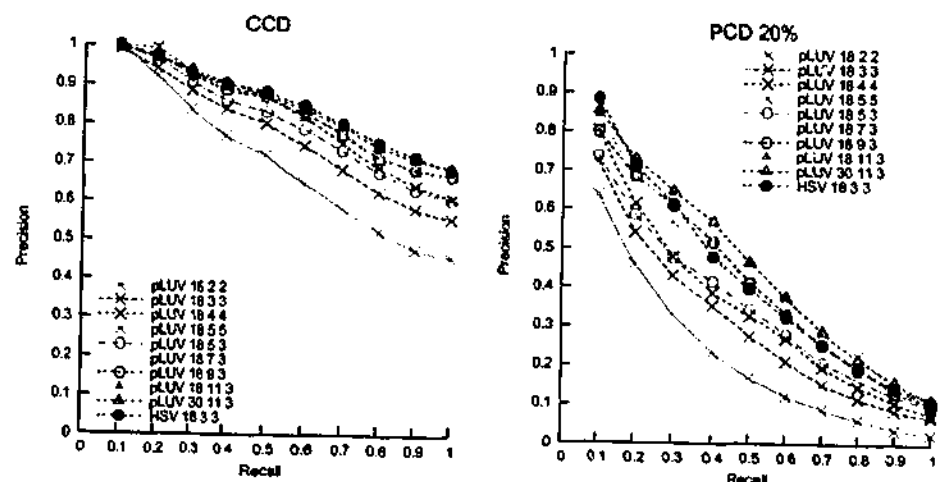
(a) Varying the number of quantisation intervals of H_{uv} (b) Varying the number of quantisation intervals of C_{uv} and L_{uv} , and C_{uv} only

Figure 3.13: Performance of pLUV quantised at different number of intervals for H_{uv} , C_{uv} and L_{uv} , or only C_{uv} in CCD and PCD.

The analysis on pLUV in Section 3.2.2.2 suggests that quantising saturation into three intervals for pLUV results in coarser quantisation than that for HSV; as a result, pLUV $18 \times 3 \times 3$ is less effective than HSV $18 \times 3 \times 3$ (see Fig. 3.13(a)). So, quantising C_{uv} into greater number of intervals may increase retrieval effectiveness. The PR graphs from Fig. 3.13(b) show that increasing the number of quantisation intervals of C_{uv} when fixing the number of quantisation intervals of L_{uv} at three results in increasing pLUV's effectiveness: the effectiveness of pLUV $18 \times 11 \times 3$ is very close to that of HSV $18 \times 3 \times 3$. This result seems to support the analysis made earlier.

Because pLUV $30 \times 3 \times 3$ is more effective than pLUV $18 \times 3 \times 3$, it was interesting to see if finer quantisation of C_{uv} and L_{uv} , or C_{uv} only when H_{uv} is quantised at 30 in-

tervals, is better than when H_{uv} is quantised into 18 intervals. To answer this question, H_{uv} was quantised into 30 intervals, and the same number of quantisation intervals of C_{uv} and L_{uv} when H_{uv} was quantised into 18 intervals were applied. To recap, C_{uv} and L_{uv} were quantised into two, three, four and five intervals (from pLUV $30 \times 2 \times 2$ to pLUV $30 \times 5 \times 5$) and C_{uv} was quantised into five, seven, nine and 11 intervals (from pLUV $30 \times 5 \times 3$ to pLUV $30 \times 11 \times 3$). The complete results can be found in Fig. B.6 of Appendix B.2.

For CCD, there are several quantisation options which are most effective and pLUV $30 \times 11 \times 3$ is one of them; for PCD, the most effective one is pLUV $30 \times 11 \times 3$. Because pLUV $30 \times 11 \times 3$ is at least as effective as others, it was chosen for further comparison. So, pLUV $30 \times 11 \times 3$ was compared with pLUV $18 \times x \times x$ as shown in Fig. 3.13(b). It seems that pLUV $30 \times 11 \times 3$ is only slightly more effective than pLUV $18 \times 11 \times 3$, which is the most effective quantisation option when H was quantised into 18 intervals. This means there is an optimum quantisation option, and using much finer quantisation intervals does not guarantee much higher effectiveness. This observation is consistent with the one made earlier for HSV and the colour spaces in Cartesian co-ordinates.

Effectiveness of pLAB

From Fig. 3.14(a) on the following page, it can be seen that like HSV and pLUV, pLAB $9 \times 3 \times 3$ is least effective. As the number of quantisation intervals increases, the effectiveness increases as well. Changing the number of quantisation intervals affects the effectiveness of pLAB more than HSV. From Fig. 3.14(b), it is clear that like pLUV, increasing the number of quantisation intervals of C_{ab} and L_{ab} also increase the retrieval effectiveness.

Like pLUV, the analysis on pLAB in Section 3.2.2.2 suggests that quantising saturation into three intervals for pLAB results in coarser quantisation than that for HSV, so pLAB $18 \times 3 \times 3$ is less effective than HSV $18 \times 3 \times 3$ (see Fig. 3.14(a)). We recommended that increasing the number of quantisation intervals of C_{ab} when fixing the

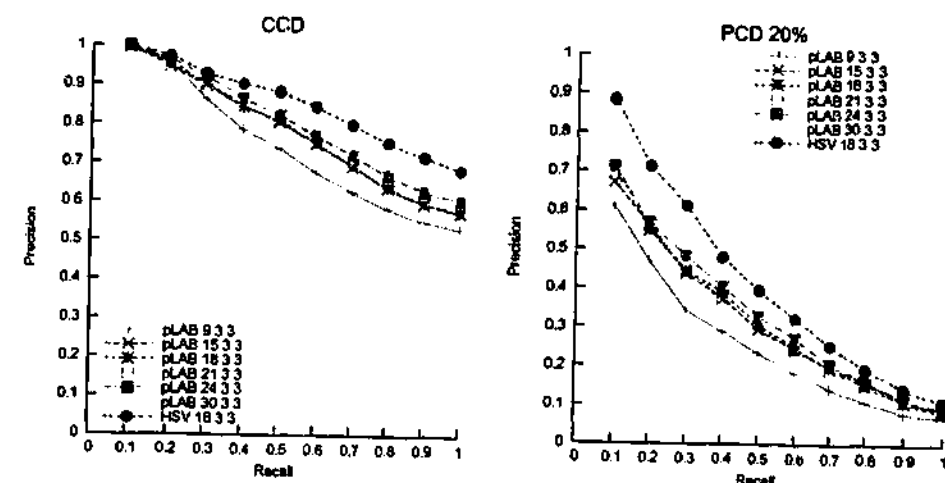
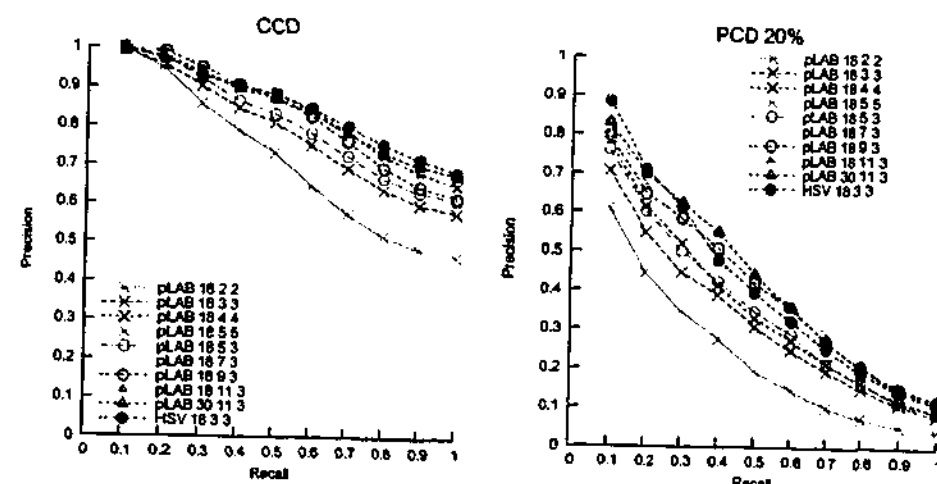
(a) Varying the number of quantisation intervals of H_{ab} (b) Varying the number of quantisation intervals of C_{ab} and L_{ab} , and C_{ab} only

Figure 3.14: Performance of pLAB quantised at different number of intervals for H_{ab} , C_{ab} and L_{ab} , or only C_{ab} in CCD and PCD.

number of quantisation intervals of L_{ab} at three may improve retrieval effectiveness. The PR graphs in Fig. 3.14(b) show that the effectiveness of pLAB $18 \times 11 \times 3$ is now very close to that of HSV $18 \times 3 \times 3$. The result appears to support the analysis made earlier.

Similarly, like pLUV, the quantisation option $30 \times 3 \times 3$ is more effective than $18 \times 3 \times 3$. To find out if finer quantisation of C_{ab} and L_{ab} axes, or only the C_{ab} axis will increase the effectiveness, the C_{ab} and L_{ab} axes were quantised finer. The same approach used in pLUV was also used here; that is C_{ab} and L_{ab} were quantised into two, three, four and five intervals (from pLAB $30 \times 2 \times 2$ to pLAB $30 \times 5 \times 5$) and C was quantised into five, seven, nine and 11 intervals (from pLAB $30 \times 3 \times 3$ to

pLAB $30 \times 11 \times 3$). The complete results can be found in Fig. B.6 of Appendix B.2.

For CCD, there are several quantisation options which are most effective, and pLAB $30 \times 11 \times 3$ is one of them; for PCD, the most effective one is pLAB $30 \times 11 \times 3$. For this reason, pLAB $30 \times 11 \times 3$ was chosen because it is at least as effective as others. So, pLAB $30 \times 11 \times 3$ was compared with pLAB $18 \times x \times x$ as given in Fig 3.14. It appears that for CCD, pLAB $30 \times 11 \times 3$ is only as effective as pLAB $18 \times 11 \times 3$, the most effective quantisation option when H was quantised into 18 intervals. For PCD, it is slightly more effective than pLAB $18 \times 11 \times 3$ at recall value of 0.1. This observation is consistent with the observation made in other colour spaces that much finer quantisation does not always guarantee higher effectiveness.

3.3.1.3 Comparing Retrieval Effectiveness of Six Colour Spaces

The previous sections showed the analysis on retrieval effectiveness of each colour space, whereas this section contains the comparison of retrieval effectiveness of all colour spaces. Figure 3.15 on the following page shows the PR graphs for each colour space at their most effective quantisation options. It can be seen that the retrieval effectiveness for the colour spaces are quite similar, and that only the difference between the best and the worst colour spaces is noticeable. It appears that with the appropriate quantisation options, the effectiveness of all colour spaces are reasonably close. The trend for PCD is the same for all levels of agreement, so only the results for PCD 20% is presented here. The complete results for PCD are given in Appendix B.3.

Discussion and Analysis

From the PR graphs, it is clear that for the CCD images, HSV is the most effective colour space followed closely by LUV and LAB, and lastly, pLUV, pLAB and RGB. For PCD 20%, RGB and HSV are equally effective, followed closely by LUV and LAB, and finally pLUV and pLAB. It is only at PCD 70% that HSV is more effective than RGB (see Appendix B.3). With RGB, although it is impossible to quantise either hue, brightness, or saturation independently, it is still effective with very high number of

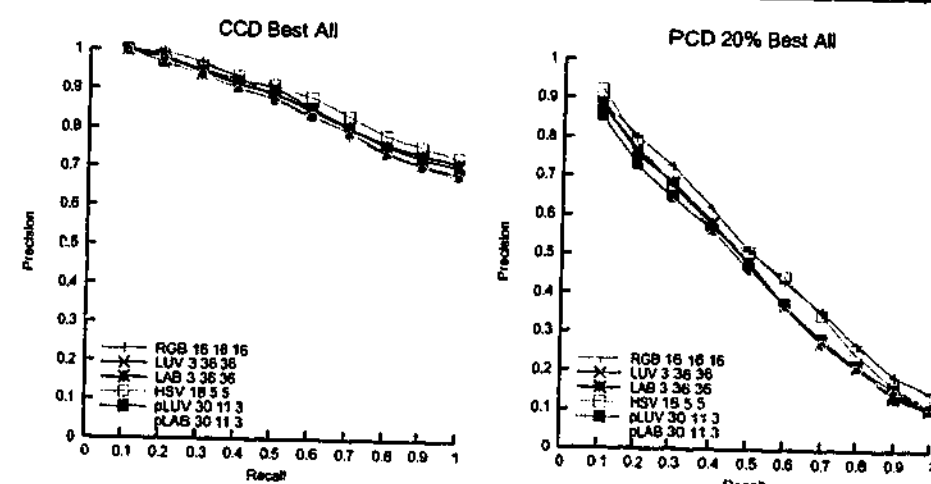


Figure 3.15: PR graphs of RGB, LUV, LAB, HSV, pLUV and pLAB colour spaces at the highest effectiveness in CCD and PCD.

quantisation intervals: RGB $16 \times 16 \times 16$ at 4096 bins.

Recent studies find that LAB is more uniform than LUV, even recommending LUV not be used at all [33]. Any differences between the two colour spaces appear to have no meaningful impact on the effectiveness of colour-based CBIR. This is because both LUV and LAB were created for a very different application in mind. In colour related industries, small colour differences are important, whereas in colour-based CBIR, small colour differences are much less important and colours with small differences are considered the same.

This completes the analysis on the effectiveness of the colour spaces, and we conclude that HSV is, overall, most effective because it is at least as effective as other colour spaces. The next section evaluates the efficiency of the colour spaces for colour-based CBIR.

3.3.2 Retrieval Efficiency

As mentioned before, there are two methods for evaluating the efficiency of the colour spaces: (1) by comparing the size of feature vectors with most similar effectiveness and (2) by comparing the effectiveness of feature vectors with similar size. The evaluations using these two methods are discussed as follows.

3.3.2.1 Method one: comparing the size of feature vectors

The quantisation options given in Fig. 3.15 in the previous section (Section 3.3.1.3) have the most similar retrieval effectiveness, and therefore, they were used for this comparison. As mentioned earlier, given two feature vectors, the one having fewer number of bins is considered more efficient.

Discussion and Analysis

Table 3.1 lists the number of bins for each colour space, sorted in ascending order of number of bins. It is clear that HSV is most efficient given that it has the least number of bins at 450, followed by pLUV at 525, pLAB at 560, LUV at 1759, LAB at 1950, and lastly, RGB at 4096. HSV, in addition to being the most efficient, is also the most effective one. In PCD, the effectiveness of HSV and RGB are very similar, but the size of the RGB feature vector is nearly ten times that of HSV. In other words, HSV is nearly ten times more efficient than RGB.

Colour spaces and their number of quantisation intervals	no. of bins
HSV $18 \times 5 \times 5$	450
pLUV $30 \times 11 \times 3$	525
pLAB $30 \times 11 \times 3$	560
LUV $3 \times 36 \times 36$	1759
LAB $3 \times 36 \times 36$	1950
RGB $16 \times 16 \times 16$	4096

Table 3.1: Evaluating the efficiency of the colour spaces by comparing the number of bins of the colour spaces at their most effective quantisation options. (number of effective bins as some bins are always empty (see Sections 3.2.2.1 or 3.2.2.2 for the definition of effective bins)).

3.3.2.2 Method two: comparing the effectiveness

The purpose of this comparison is to find a compromise between effectiveness and efficiency. Figure 3.16 on the following page shows the PR graphs of the six colour spaces quantised so that they have about the same number of bins as HSV $18 \times 3 \times 3$ (162 bins) - the complete results for PCD can be found in Appendix B.4. The quantisation at $18 \times 3 \times 3$ intervals is not the best quantisation option for HSV, but it was chosen

as a trade off between effectiveness and efficiency. For RGB, the quantisation option closest to 162 bins was chosen, that is RGB $6 \times 6 \times 6$ at 216 bins. For LUV, LAB, pLUV and pLAB, the quantisation options with the number of effective bins closest to 162 were chosen (see Sections 3.2.2.1 or 3.2.2.2 for the definition of effective bin). LUV $3 \times 10 \times 10$ has 170 effective bins, LAB $3 \times 9 \times 9$ has 157 effective bins, pLUV $18 \times 5 \times 3$ has 164 effective bins, and pLAB $18 \times 5 \times 3$ has 177 effective bins.

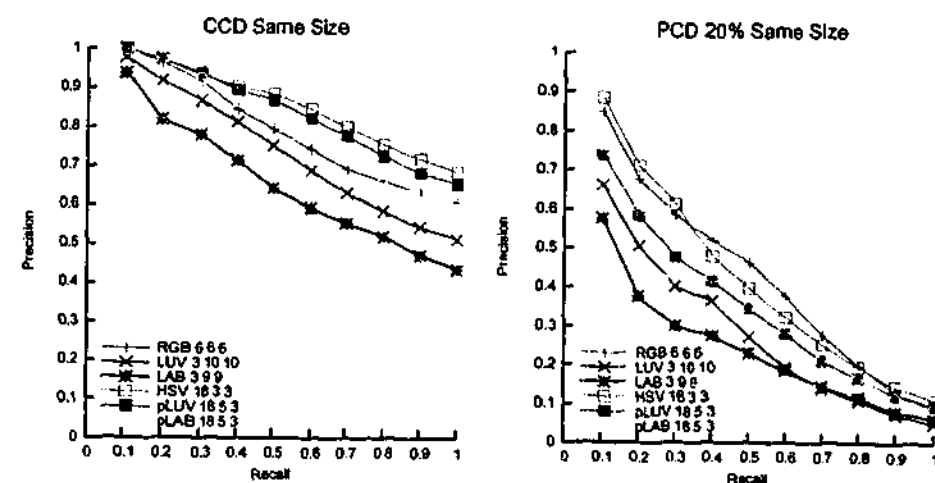


Figure 3.16: Effectiveness of RGB, LUV, LAB, HSV, pLUV and pLAB colour spaces having similar number of bins in CCD and PCD.

Discussion and Analysis

In CCD, it is clear that for the same number of bins, HSV is the most effective, followed by pLUV, pLAB, RGB then LUV, and lastly, LAB. In PCD, HSV and RGB are most effective at 20%, 30% and 50% levels of agreement; it is only at PCD 70% that HSV is clearly more effective than RGB (see Appendix B.4). Surprisingly, pLUV and pLAB were found to be now less effective than RGB. It appears that RGB is more effective in PCD than in CCD. This could be because PCD is a more subjective database than CCD, and some images which have different colours are considered similar. These different colours then happened to be quantised into the same bin in RGB. This would also explain why only in the 70% level of agreement that HSV clearly outperforms RGB. At the 70% level of agreement, only images which are visually very similar have been chosen to be relevant. These images are often the rotation or translation of their original, and therefore they have very similar colours. This is why HSV is clearly far

more effective than RGB in PCD 70%. For the same reason, pLUV and pLAB are more effective than RGB in PCD 70%.

Note that for about the same number of bins, pLUV and pLAB are more effective than LUV and LAB. This is because colour quantisation in polar co-ordinates is more efficient: H , the axis which contributes more to colour differences perceptually was quantised into finer intervals independent of other axes. In their Cartesian co-ordinates, both hue and saturation are controlled by a^* and b^* as well as u^* and v^* , so to achieve fine quantisation of hue both axes were quantised finely.

The literature in colour sciences [33, 120, 156] describes LUV, LAB, pLUV and pLAB as perceptually uniform colour spaces, so in comparison with HSV, why do they require higher number of quantisation intervals? For LUV and LAB, the reason is because the opponent colour axes (u^* and v^* , as well as a^* and b^*) control both the hue and the saturation of a colour; consequently, it is impossible to quantise hue more finely without quantising saturation. Unless they are quantised very finely, perceptually different colours will be quantised into the same bin as demonstrated in Fig. 3.11. For pLUV, the separation of hue from the saturation axis has made it much more efficient than LUV but it is still less efficient than HSV.

To explain why pLUV is less efficient than HSV, we observed the visual differences between these two colour spaces (see Fig. 3.3 for HSV and Fig. 3.4 for pLUV; note that LUV and pLUV have the same geometrical shape). One noticeable difference between both colour spaces is the arrangement of hue and saturation. For HSV when $V = 1$ (brightest), the colours forming the complete spectrum of hue from yellow, green, blue and so on are clearly visible. In addition, the complete range of saturation for different hues is visible when the value of V is high enough; for example, we can see the complete range of saturation for blue when $V = 1$. For pLUV when $L = 1.0$, hardly any colours are visible. When $L = 0.9$, more colours are visible but only colours perceived as bright are visible, and as L decreases, darker colours are visible. As an example, fully saturated blue which is perceived to be darker than fully saturated yellow is only visible at lower values of L . Also, we fail to see the complete range of saturation for different

hues at any value of L ; for example, less saturated blue is only visible at higher values of L and fully saturated blue is only visible at lower values of L . In summary, pLUV does not show the complete spectrum of hue and saturation for a given value of L . It could be these differences of colour arrangement which cause pLUV to be less efficient. It seems that HSV arrangement of hue and saturation are more suitable for colour-based CBIR. The same observation with pLUV can also be seen in pLAB, so this would also explain why pLAB is, relatively, less efficient than HSV.

From the above analysis, we conclude that HSV is, overall, the most efficient colour space for colour-based CBIR. We, however, do not conclude that LUV, LAB, pLUV and pLAB are less useful than HSV, but only suggest that they may be more suitable for other applications.

3.4 Conclusions

This chapter resolved which colour space is most suitable for colour-based CBIR by conducting retrieval experiments and by comparing the effectiveness and efficiency of six colour spaces: RGB, LUV and LAB in Cartesian co-ordinates, as well as HSV, LUV and LAB in polar co-ordinates (pLUV and pLAB). This study also provides insights into why some colour spaces are more suitable than others, and concludes with the following main findings.

With the appropriate number of quantisation intervals, there is little variation between the effectiveness of one colour space over the next best effective one, so only the difference between the most and least effective colour spaces is noticeable. It means that while it is difficult to rank the effectiveness of each colour space individually, it is possible to identify the more effective ones. In both CCD and PCD, HSV is at least as effective as all the other colour spaces; therefore, it is, overall, more effective. In practice, it may not always be possible to use the best quantisation option because of the efficiency issue. For this reason, it is necessary to find a compromise between effectiveness and efficiency. The recommended colour space for colour-based CBIR, if

efficiency is an issue, is HSV at the quantisation intervals of $18 \times 3 \times 3$ as the best compromise between effectiveness and efficiency. This study finds that HSV is, overall, most suitable for colour-based CBIR because it is at least as effective as and more efficient than any of the other colour spaces.

By analysing the results from this study, we found that colour space quantisation in polar co-ordinates is more efficient than in Cartesian co-ordinates because it is possible to quantise hue, the most important axis, independently of other axes. The quantisation of hue in Cartesian co-ordinates requires at least two axes: u^* and v^* in LUV, a^* and b^* in LAB, and all three axes in RGB. In addition, an investigation of the results also suggested that among the polar coordinate colour spaces, pLUV and pLAB are different from HSV in terms of saturation axis quantisation. The saturation axis in pLUV and pLAB needs to be quantised into finer intervals than that for HSV because the valid ranges for pLUV and pLAB are variable, whereas the valid range for HSV is constant.

Further analysis on LAB and LUV suggested that any difference between LAB and LUV has negligible impact in colour-based CBIR despite recent studies in colour science which favour LAB and suggest that LUV should not be used at all. This is because the colour science focuses on small colour differences, whereas colour-based CBIR focuses on large colour differences for reasons of effectiveness and efficiency.

Having resolved that HSV is the most suitable colour space for colour-based CBIR, we now can move forward to improving the effectiveness of colour-based CBIR. As mentioned in the previous chapter, the main problem of basic colour-based CBIR is that they ignore the richness of information provided in the spatial relationships of colours. In fact, the shape and texture features found in an image are provided by these relationships. Unfortunately, the use of these two features requires image segmentation which, to date, is inaccurate when automated and is time consuming when segmented either manually or semi-manually. By incorporating spatial relationships into colour-based feature vectors, we can still harness the richness of the spatial information without performing image segmentation. The next chapter focuses on colour-based feature vectors which incorporate spatial relationships of colours.

Spatial Information for Image Feature Extraction

The main problem with feature extraction methods based on colour distribution is they ignore the rich information provided by the spatial relationships of colours in the image. As a result, their retrieval effectiveness tends to be low because feature vectors of images with similar colour content but different spatial distributions are considered similar although, visually, they can be very different. To address this problem, more recent methods incorporate these spatial relationships and their retrieval effectiveness tends to be higher compared to that of colour distribution methods.

The purpose of the work covered in this chapter is to improve the effectiveness and, to a certain extent, the efficiency of colour-based feature extraction methods which incorporate spatial relationships of colours by first analysing colour autocorrelogram, one of the most promising existing methods, and then proposing a new method. We then compare the proposed method with two other contemporary methods.

4.1 Colour Autocorrelogram

Huang et al. [47] proposed the colour autocorrelogram method to capture the spatial relationships of colours in images. A colour correlogram “expresses how the spatial correlation of pairs of colours changes with distance” [47]. A colour autocorrelogram is concerned with how the spatial correlations of similar colours change with distance.

4.1.1 Feature Extraction

Let \mathcal{I} be an image of $n \times n$ pixels. A colour correlogram is calculated as [47]:

$$\gamma_{i,j}^{(k)}(\mathcal{I}) = \frac{\Gamma_{i,j}^{(k)}(\mathcal{I})}{A_i(\mathcal{I}) \times 8k} \quad (4.1)$$

where $\Gamma_{i,j}^{(k)}(\mathcal{I})$ is the count of colour i when it is at k pixel away from colour j , $k \in [d]$, $[d]$ is the set of pixel distances to be considered, and $A_i(\mathcal{I}) \times 8k$ is the maximum number of neighbours A_i can have. Thus, when A_i is 1 and k is 1, the maximum number of neighbours it can have is 8 and when k is 2, the maximum number of neighbours it can have is 16. A correlogram considers the spatial relationships between any pair of colours, whereas an autocorrelogram considers the relationships between two similar colours ($i = j$). The feature vector of image \mathcal{Q} using an autocorrelogram with $[d] = \{1, 2\}$ and M bins is $(\gamma_1^{\mathcal{Q}1}, \gamma_1^{\mathcal{Q}2}, \dots, \gamma_M^{\mathcal{Q}1}, \gamma_M^{\mathcal{Q}2})$. The size of an autocorrelogram feature vector is therefore $d \times M$.

4.1.2 Distance Calculation for Retrieval

The distance between two autocorrelogram feature vectors is defined by the *Canberra* dissimilarity metric [28]:

$$\text{Canberra}(\text{auto-correlogram}) = \sum_{i \in [M], k \in [d]} \frac{|\gamma_i^{\mathcal{Q}k} - \gamma_i^{\mathcal{I}k}|}{c + \gamma_i^{\mathcal{Q}k} + \gamma_i^{\mathcal{I}k}} \quad (4.2)$$

where c is a constant > 0 , to prevent division by 0. The distance can also be measured using $L1$:

$$L1(\text{autocorrelogram}) = \sum_{i \in [M], k \in [d]} |\gamma_i^{\mathcal{Q}k} - \gamma_i^{\mathcal{I}k}| \quad (4.3)$$

4.1.3 Analysis

Huang et al. observed that a colour histogram describes the probability of colours i occurring in an image by measuring the count of colours i occurring in image \mathcal{I} . Autocorrelogram, in contrast, describes the coherency of colours by measuring the probability of picking colour i in image \mathcal{I} at k pixels away. From (4.1), it can be seen that the value of γ depends on A and Γ . For a fixed value of A , γ could be high or low depending on Γ . This property is useful because γ can be used to differentiate coherent pixels from incoherent ones. A high value of γ_i indicates that colour i is highly coherent and a low value of γ_i indicates that colour i is less coherent [49]. Likewise, for a fixed value of Γ , γ could be high or low depending on A . This property is also useful because a colour with the same Γ but different A (pixel count) will be visually different.

The main problem with autocorrelogram is that γ reaches infinity when $A_i = 0$. Traditionally, in spatial statistics, this is not a problem because there is no need to compare non-existent data. However, when used for CBIR, the dissimilarity metrics defined in (4.2 and 4.3) require a comparison even though $A_i = 0$. Therefore, it is necessary to assign a number to γ when it reaches infinity as it is meaningless to compare any number to infinity. Huang et al assigned it to 0 [49].

The replacement of infinity with zero means that the intended meaning of γ no longer holds; one can no longer assume that a low value of γ is an indication of incoherency. This problem is best illustrated visually using Fig. 4.1 on the following page, which shows four images of 8×8 pixels. Note that the γ_{red} of both image (a) and (c), printed in red, is 0 although image (c) has no red pixels. It also shows their histogram features and autocorrelogram features at $[d] = \{1\}$ and $[d] = \{1, 2\}$ as well as the $L1$ distance between these features from image (a). If image (a) is the query image and $L1$ is the dissimilarity metric, then the ranking of retrieval results with autocorrelogram using $L1$ when $[d] = \{1\}$ is (c), followed by (b) and, lastly, (d). This ranking is incorrect given that (b) is visually closer to (a). Even when using autocorrelogram feature with $[d] = \{1, 2\}$, the ranking remains unchanged. In the next section, we develop a feature vector for overcoming the weakness in autocorrelogram.

		Query Image		auto-correlogram			
		hist.		at k=1	at k=2		
(a)	red	0.125		0.0	0.25		
	yellow	0.875		0.73	0.53		
		hist.		Other Images		autocorrelogram	
				at k=1	at k=2	$d_{L1}(\text{autocorrelogram})$	
						$[d]=\{1\}$	$[d]=\{1,2\}$
(b)	red	0.125		0.13	0.0	0.16	0.42
	yellow	0.875		0.76	0.54		
(c)	red	0.0		0.0	0.0	0.09	0.38
	yellow	1.0		0.82	0.57		
(d)	red	0.125		0.5	0.125	0.57	0.715
	yellow	0.875		0.8	0.55		
						$d_{L1}(\text{hist.})$	
						0.0	
						0.25	
						0.0	

Figure 4.1: Four images with their histogram values and autocorrelogram values when $[d] = \{1\}$ and $[d] = \{1, 2\}$ for illustrating the weaknesses and strengths of autocorrelogram and histogram methods.

4.2 Improving Autocorrelogram

In the previous section, we showed that the spatial description of autocorrelogram is sometimes inaccurate. This problem can be partially solved with the colour histogram which can identify that it is image (b), not (c), that is closer to the query image. We said it partially solves the problem because the histogram method ignores the spatial relationships of colours, and consequently, the distance between images (b) and (d) to image (a) is the same i.e. 0, even though image (b) is visually more similar to (a). On the other hand, autocorrelogram which considers the spatial relationships of colours can correctly identify the distance between image (b) and (d) to image (a).

In summary, colour histogram describes pixel counts and autocorrelogram considers the spatial relationships, so a combination of these two methods can be potentially effective. If $[d] = \{1, 2\}$, the feature vector of image Q is now: $(h_1^Q, \gamma_1^{Q1}, \gamma_1^{Q2}, \dots, h_M^Q, \gamma_M^{Q1}, \gamma_M^{Q2})$.

The distance between any two feature vectors is defined as:

$$w_1 * dis(\text{autocorrelogram}) + w_2 * L1(\text{histogram}) \quad (4.4)$$

where $w_1 + w_2 = 1.0$, and dis could be *Canberra* (4.2) or *L1* (4.3).

We then conducted retrieval experiments to show that the proposed method is more effective than the colour histogram or autocorrelogram alone. As both features have different physical meanings, it is also necessary to determine the value of w_1 and provide an explanation as to why the value is optimum. Theoretically, the sum of a normalised histogram is 1.0 and the maximum *L1* distance between two normalised histograms $L1(\text{histogram})$ is 2. On the other hand, the sum of γ s in an autocorrelogram approaches M , where M is the number of bins - from (4.1), we know that the value of γ approaches 1. We also know that the upper bound distance between two autocorrelograms is when one feature vector has $\frac{M}{2}$ bins of maximum coherency and the other feature vector has the other $\frac{M}{2}$ bins of maximum coherency. In other words, the upper bound of $L1(\text{autocorrelogram})$ approaches M . If w_1 is used to control the contribution of each type of feature vector such that the maximum distance for $L1(\text{autocorrelogram})$ is equal to the maximum distance for $L1(\text{histogram})$ i.e. 2, then the value of w_1 is $\frac{1}{(M/2)}$; however, because the maximum distance between two feature vectors in real images for both $L1(\text{autocorrelogram})$ and $L1(\text{histogram})$ does not normally reach this upper bound, the value of w_1 must be determined empirically. This analysis, however, suggests that w_1 should be set at a much lower value than w_2 , and we will investigate if this analysis is correct. If w_1 is set to 0, the feature vector is the same as colour histogram and if it is set to 1.0, the feature vector is the same as colour autocorrelogram. When $0.0 > w_1 < 1.0$, the feature vector is known as I-autocorrelogram (I-auto), standing for improved autocorrelogram.

To illustrate why I-auto is potentially more effective than either histogram or autocorrelogram, we recalculated the distance between the query image with the other three images in Fig. 4.1 (the distance calculation was based on the dissimilarity metric in (4.4), where dis is *L1*, $w_1 = 0.2$ and $w_2 = 0.8$):

- distance between image (a) and (b) is now 0.032;
- distance between image (a) and (c) is 0.218; and
- distance between image (a) and (d) is 0.114.

Consequently, the ranking of retrieval is now (b), then (d), followed by (c): the results now conform to visual similarity of the four images.

4.3 Experimental Parameters

There are three main purposes of the experiments: (1) to show if the proposed method is more effective, (2) to determine the value of w_1 and (3) to explain why this value is optimum. Besides this, there are other issues which need to be addressed and they will be described as follows.

Huang et al suggested using RGB colour space with uniform quantisation of four intervals of R, G and B, which gives 64 bins, and the *Canberra* dissimilarity metric with $c=1$ and $[d] = \{1, 3, 5, 7\}$ [47]. However, no experiment was carried out to test how these parameters influence the effectiveness of autocorrelogram. The justification for studying the effect of the dissimilarity metric, the choice of $[d]$ and colour space is explained next.

4.3.1 Dissimilarity Metric

$L1$ is used widely in CBIR to measure the dissimilarity between two feature vectors, so we are interested to know if it can also be used for autocorrelogram. We set c in the *Canberra* dissimilarity metric to 1.0 as recommended by Huang et al [49] and evaluated the effect of both metrics.

4.3.2 Choice of $[d]$

Huang et al. recommended using $[d] = \{1, 3, 5, 7\}$, but the influence of $[d]$ on the effectiveness was never studied. Note that as the number of d increases, the size of the feature vector also increases therefore reducing efficiency. Nevertheless, the choice of $[d]$ was never justified. In this experiment, we also studied the effectiveness of autocorrelogram at different levels of d , that is when $[d] = \{1\}$, $[d] = \{1, 3\}$, $[d] = \{1, 3, 5\}$ and $[d] = \{1, 3, 5, 7\}$. The purpose of studying this parameter is to justify the choice of $[d]$.

4.3.3 Colour Space

Huang et al. also recommended using autocorrelogram in RGB with quantisation intervals of $4 \times 4 \times 4$. It was established in Chapter 3 that the effectiveness of colour-based CBIR with uniform quantisation is highly dependent on the colour space and quantisation intervals. We found that RGB $4 \times 4 \times 4$ is less effective than the recommended colour space at the recommended quantisation intervals i.e. HSV $18 \times 3 \times 3$. To be complete, we first evaluated autocorrelogram in RGB $4 \times 4 \times 4$ and HSV $18 \times 3 \times 3$. Then, for a fair comparison, we also evaluated RGB $6 \times 6 \times 6$, which has a similar number of bins to HSV $18 \times 3 \times 3$.

4.3.4 Weighting of w_1

The optimum value of w_1 was obtained empirically by varying the value of w_1 , starting from $w_1 = 0.0$ with an increment of 0.05, 0.1 or 0.2 until $w_1 = 1.0$. If there is an optimum dissimilarity metric and choice of $[d]$, then we only need to vary the value of w_1 at the optimum dissimilarity metric and $[d]$. However, it is necessary to evaluate w_1 at RGB $4 \times 4 \times 4$ and RGB $6 \times 6 \times 6$ as well as HSV $18 \times 3 \times 3$ in order to explain why there is an optimum value for w_1 .

4.4 Experimental Setup and Evaluation Criteria

The experiment setup and evaluation criteria were the same as defined in Section 3.1, but for the evaluation of efficiency, only method one will be used since we are not making any recommendation on quantisation option. To recap, the retrieval experiment was conducted in GIFT using PCD and CCD. The effectiveness was evaluated using PR graphs and the efficiency was evaluated using only method one, that is by comparing the size of feature vectors. Like the previous chapter, in PCD, 50 images were used as queries and in CCD, 32 images were used as queries. The PR graphs for each database are made up of the average PR graphs from all queries in each database.

4.5 Results and Discussion

This section examines the effect of four parameters: the dissimilarity metric, $[d]$, colour spaces and the values of w_1 .

4.5.1 Effect of Dissimilarity Metric

Figure 4.2 on the following page shows the PR graphs with CCD using RGB $4 \times 4 \times 4$ at different values of d using $L1$ and the *Canberra* dissimilarity metrics. It is clear that the difference between the two dissimilarity metrics at all values of d is negligible. This observation is also true with CCD in HSV $18 \times 3 \times 3$, PCD in RGB $4 \times 4 \times 4$ and HSV $18 \times 3 \times 3$ (all graphs can be found in Appendix C.1). The remaining experiments therefore use only the $L1$ dissimilarity metric, as it is simpler.

4.5.2 Effect of $[d]$

Figure 4.3 on the following page shows the PR graphs when $w_1 = 1.0$ for CCD and PCD 20% in RGB and HSV colour spaces. As mentioned before, when $w_1 = 1.0$, the feature vector is the same as the autocorrelogram. With CCD in RGB and HSV, there was only a slight difference at recall value of 0.1 and, later, at recall value of 0.7. With

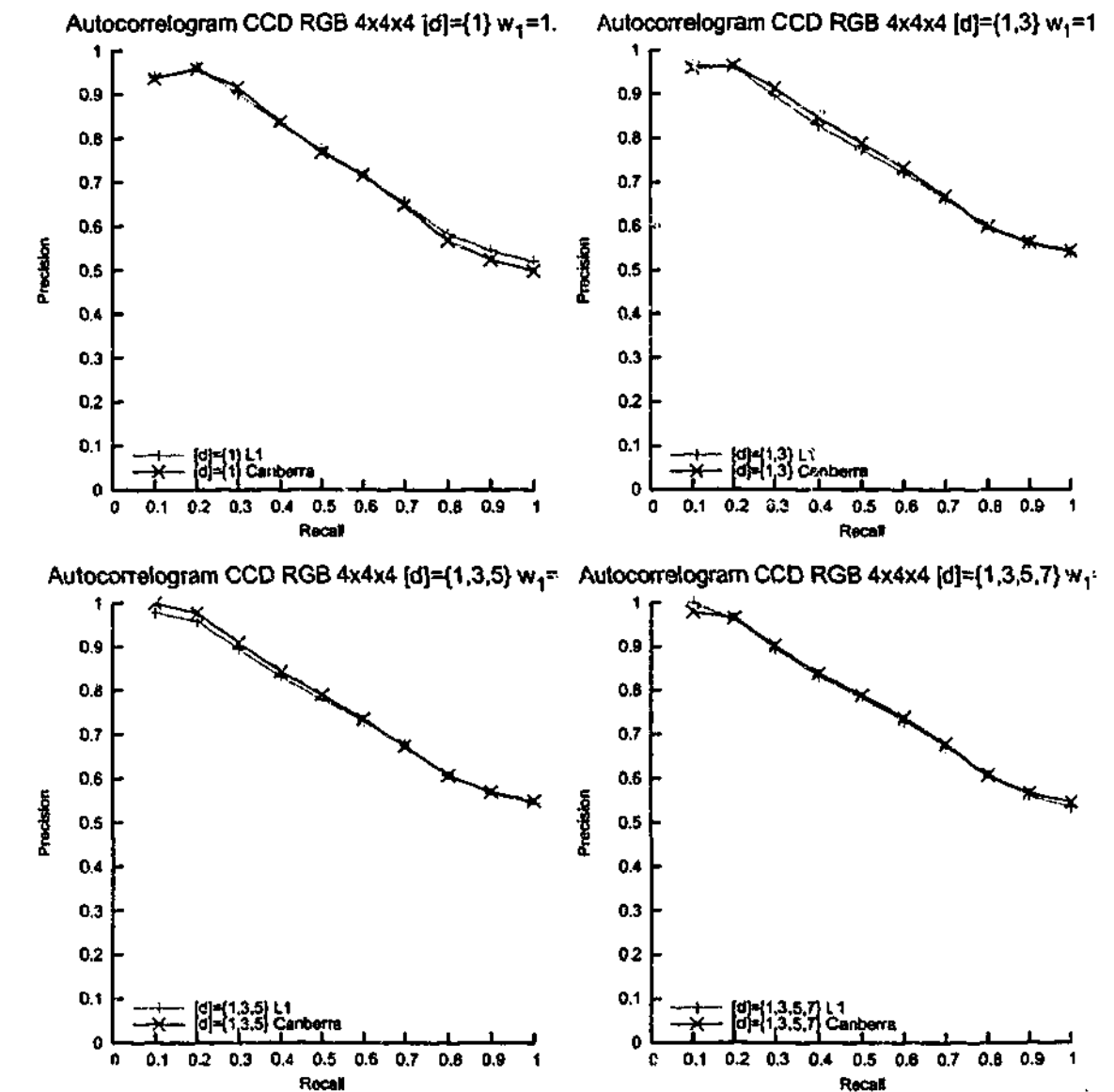


Figure 4.2: Autocorrelogram at $[d] = \{1\}$, $[d] = \{1, 3\}$, $[d] = \{1, 3, 5\}$ and $d = \{1, 3, 5, 7\}$ using $L1$ and *Canberra* dissimilarity metric.

PCD 20% in RGB, the size of $[d]$ had some effect at recall value of 0.1 and 0.3, but in HSV, there was an almost negligible effect (the results with PCD at all levels of agreement can be found in Appendix C.2). To summarise, the effect of the size of $[d]$ on the effectiveness is negligible.

This finding is significant because the size of the feature vector when $[d] = \{1, 3, 5, 7\}$ is four times the size of the feature vector when $[d] = \{1\}$, yet they both have similar effectiveness. Theoretically, small values of k capture spatial relationships in small blocks and large values of k capture spatial relationships in large blocks, so higher values of

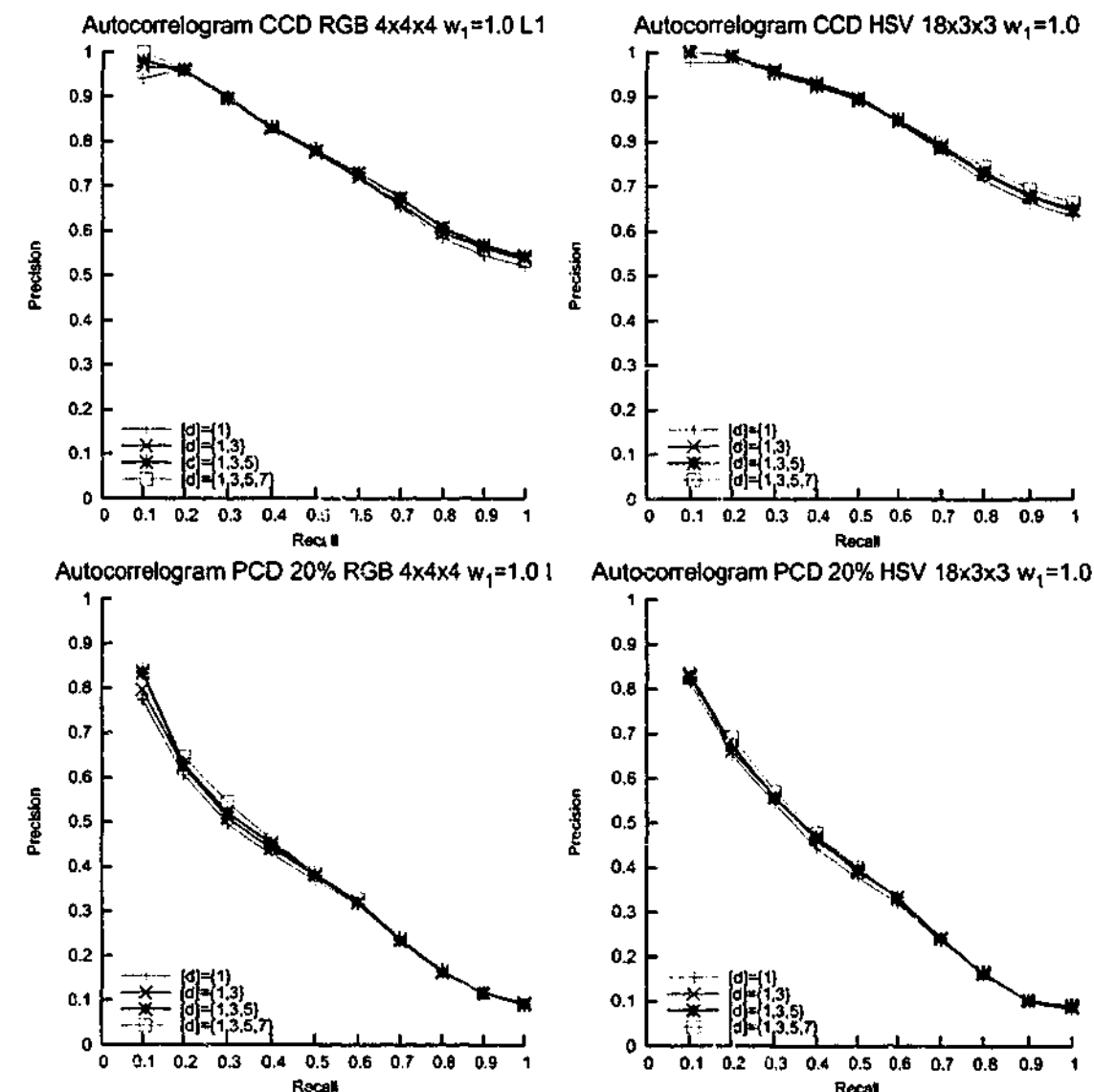


Figure 4.3: PR graphs with CCD and PCD 20% in RGB and HSV colour space at different $[d]$ for autocorrelation ($w_1 = 1.0$).

k can better capture overall colour relationships. In practice, when autocorrelation is used for CBIR, the majority of spatial relationships are already captured when $k = 1$, so the contribution of k when it is large is negligible. Figure 4.4 shows two images of 5 by 10 pixels. The spatial relationships of red in both images are different and this difference, to a certain extent, is already captured by autocorrelation when $k=1$. Although the difference can also be captured when $k = 2$ and $k = 3$, what can be captured at higher values of k is already captured at $k=1$. This means that in most cases, using a higher value of k has little effect on retrieval effectiveness. In the remaining sections, we will only study the results for $[d] = \{1\}$.

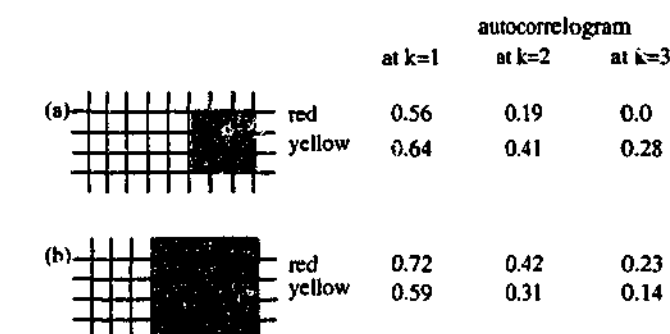


Figure 4.4: Two images at different values of k . The spatial relationships of the colours, to a certain extent, is already captured by autocorrelation when $k=1$.

4.5.3 Effect of Colour Space

Figure 4.5 on the following page shows the PR graphs when $w_1 = 1.0$ with CCD and PCD 20% in RGB 4 × 4 × 4, RGB 6 × 6 × 6 and HSV 18 × 3 × 3 (the complete results with PCD can be found in Appendix C.3). With CCD using RGB, autocorrelation in both quantisation levels is more effective than the colour histogram. In Chapter 3, we showed that in RGB 4 × 4 × 4, many perceptually different colours are quantised into the same bin. In colour histogram, the probability of perceptually different colours having a similar histogram count is high; therefore, the retrieval effectiveness of colour histogram is low. On the other hand, with autocorrelation, the probability of perceptually different colours having similar correlation is lower than that of colour histogram; therefore, autocorrelation is more effective than the colour histogram. For HSV 18 × 3 × 3, it is

hard to say if autocorrelogram or the colour histogram is more effective, as they are more effective than the other at different recall values.

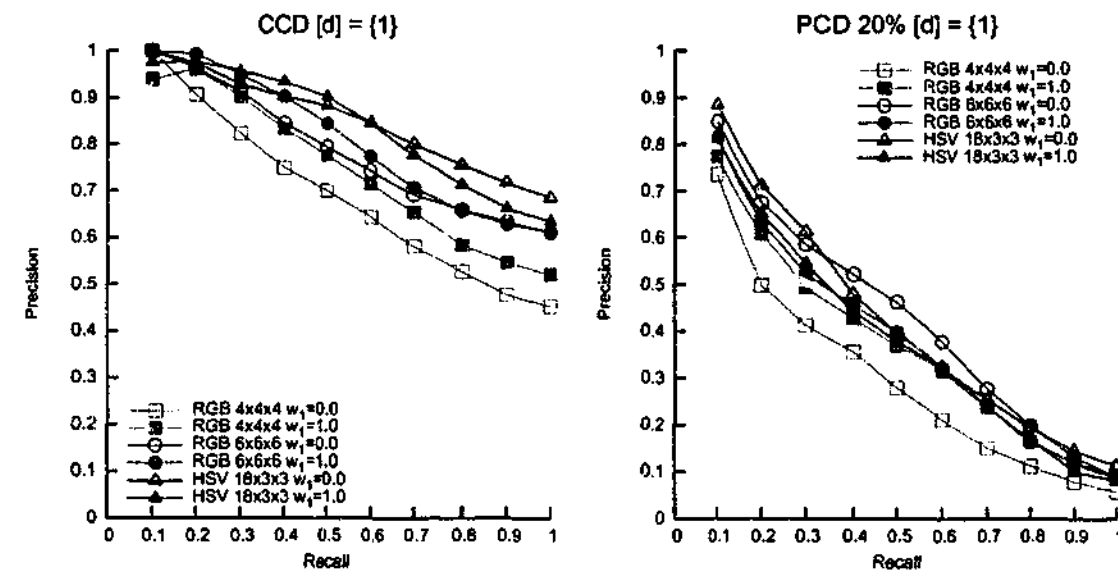


Figure 4.5: PR graphs with CCD and PCD when $[d] = \{1\}$ for colour histogram ($w_1 = 0.0$) and autocorrelogram ($w_1 = 1.0$).

With PCD using RGB, autocorrelogram RGB $4 \times 4 \times 4$ is more effective than the corresponding histogram. On the other hand, with RGB $6 \times 6 \times 6$, it is only at PCD 70% that autocorrelogram is more effective (see Appendix C.3). With PCD 20%, 30% and 50%, it appears that the histogram is more effective. One possible explanation is that because PCD is a more subjective database, it could be that some images have similar colour count but different spatial correlations. This would also explain why only at the 70% level of agreement that autocorrelogram outperforms histogram. This is because at such a high level of agreement, only images which are visually very similar have been chosen to be relevant, so the colours are more likely to have similar spatial correlations.

At HSV $18 \times 3 \times 3$, surprisingly, the histogram is more effective than autocorrelogram. Perhaps in HSV $18 \times 3 \times 3$, the spatial descriptor γ reaches infinity more often than both RGB $4 \times 4 \times 4$ and RGB $6 \times 6 \times 6$. The results of autocorrelogram in HSV $18 \times 3 \times 3$ using CCD and PCD show the importance of testing a new feature vector using several colour spaces. When autocorrelogram was first proposed, the comparison was made only in RGB $4 \times 4 \times 4$, and as a result, it seemed that autocorrelogram was always

better than the colour histogram, but our study shows otherwise.

4.5.4 Weighting of w_1

The value of w_1 was tested using the dissimilarity metrics $L1$ with $[d]$ at 1 because $L1$ is just as good as the *Canberra* dissimilarity metric, and the effect when $[d] > 1$ is negligible. The PR graphs in RGB $4 \times 4 \times 4$, RGB $6 \times 6 \times 6$ and HSV $18 \times 3 \times 3$ with both CCD and PCD 20% are given in Fig. 4.6 on the following page (the complete results with PCD can be found in Appendix C.4).

It can be seen from the graphs that the optimal value of w_1 for RGB $4 \times 4 \times 4$ is 0.2; the optimal value of w_1 for RGB $6 \times 6 \times 6$ is 0.1; and the optimal value of w_1 for HSV $18 \times 3 \times 3$ is $\{0.05, 0.1\}$ (i.e. the results when $w_1 = 0.05$ or when $w_1 = 0.1$ is very similar). In Section 4.2, we predicted that the value of w_1 should be kept much lower than w_2 and the empirical study confirmed this. The reason that the w_1 of RGB $6 \times 6 \times 6$ is lower than that of RGB $4 \times 4 \times 4$ is because there are more bins in RGB $6 \times 6 \times 6$ (216 bins) than in RGB $4 \times 4 \times 4$ (64 bins). The sum of γ s in an autocorrelogram with a greater number of bins will be higher than a feature vector with a fewer number of bins, so w_1 can also be viewed as the normalising factor.

The discussion on the parameters of I-auto is now complete. The next section compares I-auto at the appropriate value of w_1 with autocorrelogram.

4.5.5 Effectiveness of New Feature Extraction Method (I-auto)

Figure 4.7 on the following page compares the effectiveness of the I-auto at the appropriate value of w_1 with autocorrelogram. To make the discussion easier, all I-auto feature vectors calculated at their optimum w_1 are indicated with an asterisk (*); for example, *HSV $18 \times 3 \times 3$ means I-auto calculated in HSV $18 \times 3 \times 3$ where $w_1 = 0.05$. It is clear that for the same colour space in the same quantisation level, I-auto is always better than autocorrelogram. In Section 4.5.3, we found that for HSV $18 \times 3 \times 3$, histogram is, overall, more effective than autocorrelogram. It is interesting to see how

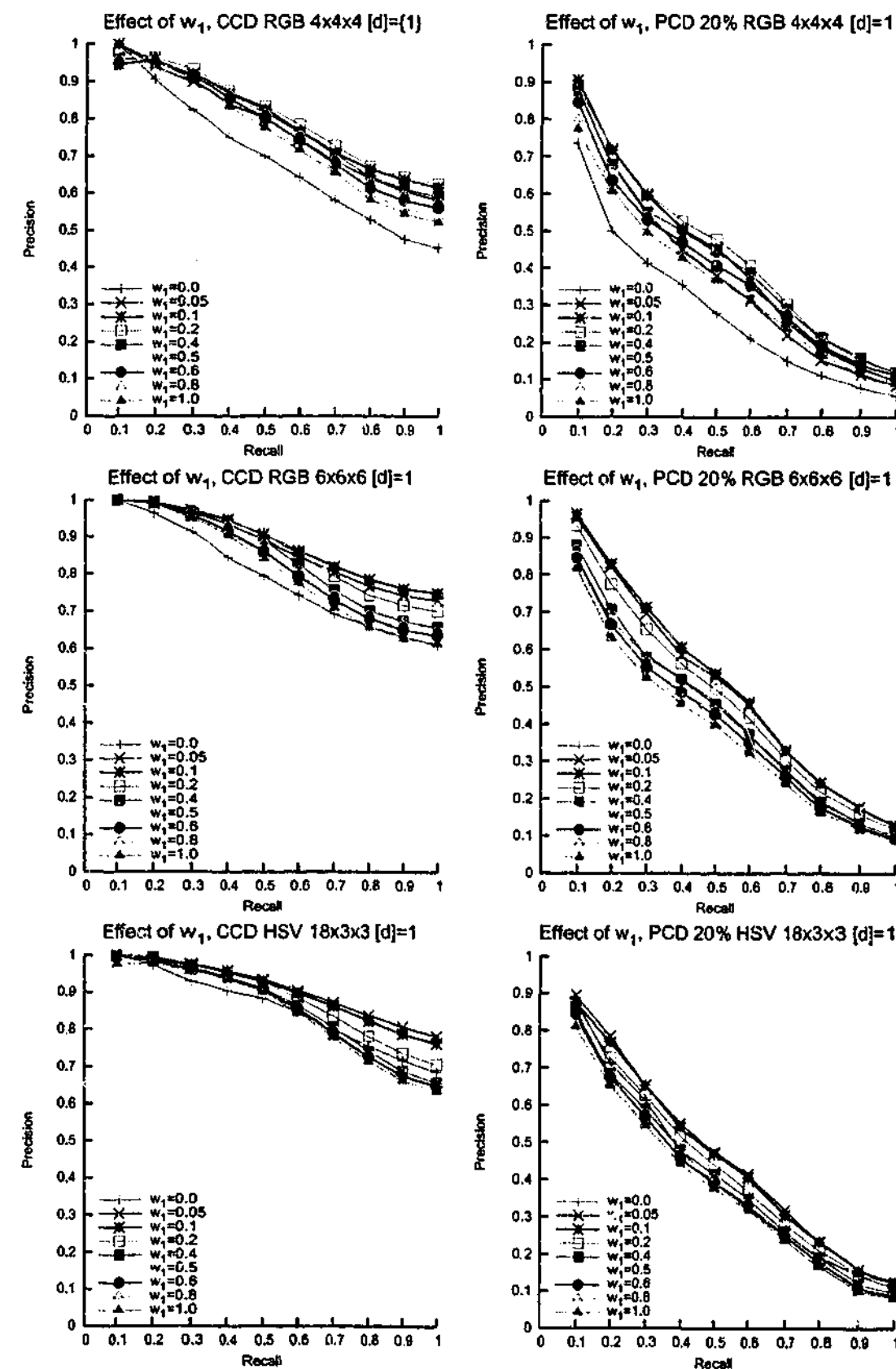


Figure 4.6: PR graphs of the I-auto at different weight for the autocorrelogram in RGB $4 \times 4 \times 4$ using CCD and PCD.

I-auto performs against histogram for HSV $18 \times 3 \times 3$. Figure 4.7 clearly shows that *HSV $18 \times 3 \times 3$ is more effective than HSV $18 \times 3 \times 3$ in both CCD and PCD, so it means that I-auto is more stable than the autocorrelogram.

With CCD, *HSV $18 \times 3 \times 3$ is most effective. With PCD 20%, *RGB $6 \times 6 \times 6$ and *HSV $18 \times 3 \times 3$ are the top two most effective feature vectors with the RGB being slightly better than the HSV; however, *HSV $18 \times 3 \times 3$ is slightly more effective than *RGB $6 \times 6 \times 6$ at PCD 70% (see Appendix C.5). We, therefore, cannot conclude whether *HSV $18 \times 3 \times 3$ is better or worse than *RGB $6 \times 6 \times 6$ but we can confidently conclude that given the same colour space and the same number of quantisation intervals, I-auto is always more effective than the histogram or autocorrelogram alone.

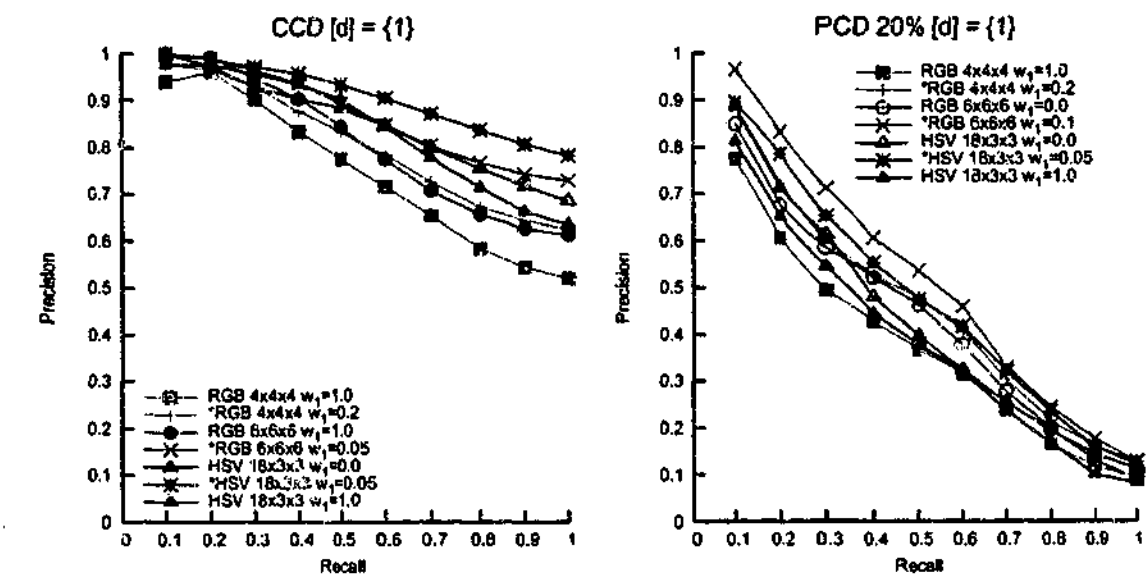


Figure 4.7: PR graphs of CCD and PCD 20% when $[d] = \{1\}$ and different values of w_1 .

This section completes the analysis on I-auto and the comparison of I-auto with histogram and autocorrelogram. The next section discusses the evaluation of I-auto against contemporary methods.

4.6 Evaluating I-auto Against Contemporary Feature Extraction Methods

To show the strength of I-auto, we compared I-auto against other contemporary feature extraction methods which incorporate the spatial relationships of colours i.e. Spectrally Layered Colour Indexing (LCI) and MPEG-7's Colour Structure Descriptor (CSD). The reason for choosing these two methods is as follows.

Autocorrelogram is one of the best colour-based feature extraction methods using uniform quantisation which incorporate spatial relationships of colours and is often compared against other contemporary methods such as colour anglograms and LCI [107, 145]. Tao and Grosky found that the colour anglograms method is more effective than autocorrelogram, and Qiu and Lam found that LCI is more effective than autocorrelogram. In this research, however, we only compare LCI with I-auto. The colour anglograms method was not used because it requires image segmentation and the procedure for segmenting the images appears quite specialised. It appears that the segmentation method will only work for colour images which have distinct large objects with uniform colours.

The other method compared with I-auto is Colour Structure Descriptor (CSD), one of the colour feature extraction methods defined in the (MPEG-7) standard. It was singled out for the comparative study because it is the most effective MPEG-7 colour-based feature extraction method [72]. The next sections describe LCI and CSD in more details.

4.6.1 Spectrally Layered Colour Indexing (LCI)

LCI was reviewed in Chapter 2. It belongs to pixel-based classification methods which capture the spatial relationships of colours by classifying pixels into different categories. LCI classifies the pixels into four categories, and after categorising the pixels, it then extracts the colour feature at each category using colour histogram [107]. The LCI feature vector for an image is the aggregation of the histogram at each category: (f_1, \dots, f_4) ,

where f_i is the histogram of colours at frequency level i . To categorise each pixel into four levels of frequency, we need three thresholds: T_1, T_2 and T_3 . In this experiment, we used the threshold values specified by Qiu and Lam ($T_1 = 6, T_2 = 12$ and $T_3 = 18$). The histogram for each layer was then extracted using HSV $18 \times 3 \times 3$. Qiu and Lam also used the *Canberra* dissimilarity metric (4.2) to calculate the distance between any two LCI feature vectors [107]. To be complete, we also performed the experiment using L1 dissimilarity metric.

4.6.2 MPEG-7 Colour Structure Descriptor (CSD)

CSD captures the colour distribution as well as the spatial distribution in images; it is histogram-like with the additional spatial information. The extraction of CSD feature vectors is described below with the aid of Fig. 4.8:

1. An element of 4×4 pixels which must always be within the image visits every pixel in the image at least once.
2. If the quantised colour is within the element, the histogram of the corresponding bin is incremented by 1.

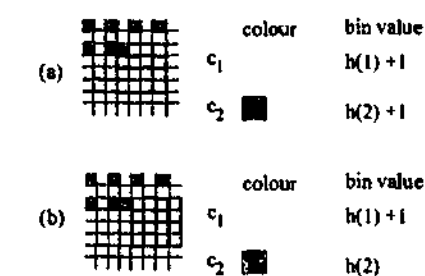


Figure 4.8: Extraction of the CSD feature vector.

For example, in Fig. 4.8(a), both quantised colours are in the element, so the histogram count of both colours are incremented by one. The element is then shifted by one pixel to the right as seen in Fig. 4.8(b), and only the histogram count of yellow colour is incremented because the element contains only yellow pixels. If a colour is coherent, then the histogram count of the corresponding bin will be high, since it will

be found within the element more frequently. On the other hand, if a colour is incoherent, then the histogram count of the colour will be low. The example uses a mask of 4×4 pixels to simplify the discussion, but the actual implementation uses an element of 8×8 pixels, as recommended by the MPEG-7 standard.

The amplitude of the histogram is then normalised to the range [0,1] and non-uniformly quantised into eight bits - the MPEG-7 group obtained the quantisation option from an empirical study. It is believed that the nonlinearity of the quantisation "give[s] the small values greater weight in the similarity measure than they would otherwise have" [72]. The distance between any two CSD feature vectors is calculated using the $L1$ dissimilarity metric. CSD uses the HMMD (*Hue - Max - Min - Diff*) colour space proposed by the MPEG-7 group [72]. It was claimed that HMMD is "closer to a more perceptually uniform colour space" [72], but it is not widely used even within the MPEG-7 community. The HMMD colour space is then non-uniformly quantised into 32, 64, 128 or 256 bins, with 256 bins being the most effective. For this reason, we used the quantisation option of 256 bins in this study.

4.6.3 Experimental Setup and Evaluation Criteria

To evaluate which feature extraction method is more effective, several retrieval experiments were conducted in CSD, LCI, and I-auto HSV $18 \times 3 \times 3$ ($w_1 = 0.05$). The experiment setup and evaluation criteria were the same as in Section 4.4.

4.6.4 Results and Discussion

Figure 4.9 on the following page shows the PR graphs with CCD and PCD 20%. The trend using PCD is similar at all levels of agreement, and the complete results can be found in Appendix C.6. With CCD, it can be seen that I-auto and CSD are most effective, followed by LCI. It is also clear that in LCI, there is not much difference between using $L1$ or the *Canberra* dissimilarity metric. With PCD 20%, I-auto and LCI are most effective followed by CSD. The next sections will analyse the results of comparing I-auto with CSD and then with LCI.

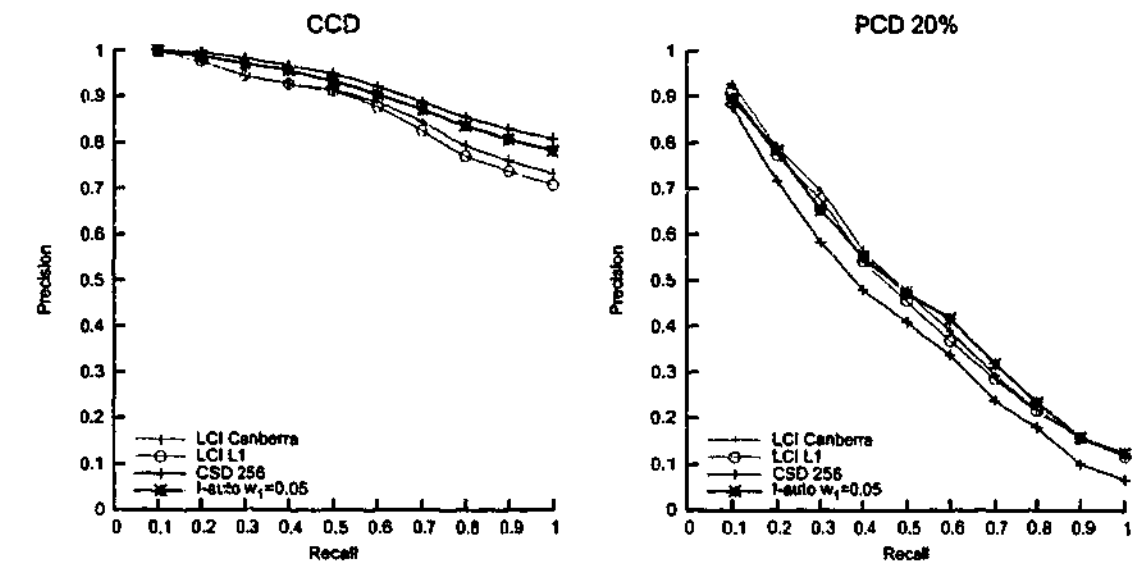


Figure 4.9: PR graphs for LCI, CSD and I-auto with CCD and PCD.

4.6.4.1 Analysing Results of I-auto and CSD

Figure 4.9 shows that CSD is slightly more effective than I-auto in CCD but less effective than I-auto in PCD. CSD was tested using CCD and the parameters (the quantisation intervals of HMMD and the normalisation of the histogram amplitude) were selected to maximise its effectiveness. It could be that the parameters are tuned only for CCD, so when it is used to extract feature vectors from a different database, the recommended parameters are no longer optimum. Having to customise the parameters makes the use of CSD in real world applications less practical because these parameters have to be recalculated for each database, and they may change as new images are added to the database. In contrast, the parameters in I-auto (the number of quantisation intervals of HSV and the value of w_1) are generic, so the same parameters can be used for other databases. For this reason, I-auto is preferred over CSD.

4.6.4.2 Analysing Results of I-auto and LCI

With CCD, I-auto is more effective than LCI because I-auto's spatial measurement is more accurate than that of LCI. LCI uses a pixel classification method which can be viewed as a quantisation of the spatial correlation. In contrast, I-auto captures the exact spatial correlation, so it can retrieve more relevant images. LCI, which quantises the

spatial correlation, uses the same quantisation level to describe many more images, and, as a result, it retrieved more irrelevant images. With PCD, I-auto is just as effective as LCI. Recall that the relevant images in PCD are obtained using a subjective test, so relevant images may have different spatial relationships. Because of this, the use of LCI is sufficient and the advantage of a more accurate spatial description is less evident.

In terms of efficiency, LCI captures the relationships of colours by pixel classification, and the methods using this approach is highly inefficient. In contrast, I-auto captures the spatial relationships using a spatial descriptor, which is a more efficient approach than pixel classification: the size of I-auto is half the size of LCI.

In summary, I-auto is, overall, more effective than LCI. Given that the size of I-auto is half the size of LCI, it is therefore also more efficient than LCI.

4.7 Summary

To show that I-auto - the proposed method - is most preferred, we compared I-auto with two contemporary methods which incorporate spatial relationships of colours (CSD and LCI) by conducting retrieval experiments. It was found that I-auto is preferred over CSD because I-auto's parameters are more generic than those of CSD, and therefore, they can be used for other image databases. The parameters for CSD, that is the quantisation for the colour space and normalisation of amplitude, not only have to be customised for each database but they also need to be recalculated as new images are added to the database. This study also found that I-auto is at least as effective as LCI but twice as efficient. We, therefore conclude that I-auto is preferred over CSD and LCI for extracting colour features incorporating spatial relationships.

4.8 Conclusions

An analysis of autocorrelogram suggests that colour histogram and autocorrelogram complement each other, so it is natural to use both methods to extract features from an

image. Because both features have different physical meanings and ranges of values, it is necessary to control the contribution of each feature to the overall dissimilarity using a weighting factor in the dissimilarity metric. The value of this factor was determined empirically, by carrying out retrieval experiments. We found that there is an optimum value for this factor and explained why. The proposed method is known as I-auto, standing for improved autocorrelogram. Experimental results showed that I-auto is more effective than the autocorrelogram and colour histogram.

I-auto is then evaluated against LCI and CSD, two contemporary colour-based feature extraction methods which incorporate spatial relationships of colours, by carrying out retrieval experiments. This study found that I-auto is preferred over CSD because I-auto is more generic than CSD. It also found that I-auto is at least as effective as LCI but twice as efficient. LCI captures the relationships of colours by pixel classification, which is a highly inefficient approach. In contrast, I-auto captures the spatial relationships using a spatial descriptor, which is a more efficient approach than pixel classification. We thus conclude that I-auto is preferred over CSD and LCI for extracting colour features incorporating spatial relationships.

This chapter concludes the research on feature extraction. The rest of the thesis will now focus on browsing large scale image databases.

Browsing Large Scale Image Databases

The research in the previous chapters is concerned with image retrieval, that is, given a sample image, find a set of similar ones. Unlike image retrieval, where we look for specific sets of images, when browsing, we "examine in a casual way" [26]. The notion of browsing is that viewers inspect large collections of objects, hoping to *discover* items of interest. People browse in their daily life i.e. in shops, supermarkets and libraries. In these places, the items are arranged systematically so that shoppers or visitors can find what they want with relative ease. Browsing also implies interaction; for example, if an item is partially obscured, it is possible to shift or remove the offending item.

Image browsing is far less understood than image retrieval, so we need to establish what it means. In this thesis, it is similar to browsing in daily life. A computer program which facilitates image browsing must replicate the two functions that enable browsing in daily day life i.e. systematic arrangement and interaction. One systematic arrangement of images is to group them by visual similarity. A program which displays images in such a layout enables users to *visualise* the database, that is, to obtain a quick overview of the content of the entire database. The program must also provide a set of tools so that users can interact with (*navigate*) the database. In summary, image browsing in this thesis refers to the ability to *visualise* and *navigate* image databases. Browsing is another search mode, and it is an important natural complement to retrieval. In fact, their roles are so complementary that they can be integrated to support one another. However, the research in image browsing is relatively new in

comparison to image retrieval.

The aim of the research described in this chapter is to formulate an image browsing framework appropriate for browsing large image databases by providing users an overview of the entire database and intuitive navigation tools. To be successful, the framework must enable users to transfer their daily browsing behaviour into browsing images. A review of current work finds that no existing system supports browsing. The proposed framework not only supports browsing but also allows users to find an appropriate sample image to initiate a visual query. In a visual query, users submit a sample image in order to retrieve a set of relevant images (query-by-example).

5.1 Previous Work

The visualisation of image databases is made possible by proximity visualisation in which the location of images are dictated by their visual similarities and dissimilarities. Faloutsos and Lin were the first to describe image database visualisation [35]. They first proposed "FastMap", an algorithm for reducing high dimensional feature vectors into two or three dimensions (2D or 3D). Then, they plotted the reduced feature vectors as points, not images, into a 2D or 3D space. The purpose for creating the visualisation display was to reveal potential clusters and other features useful for data-mining, not to support browsing.

More recent systems attempted to support browsing by displaying the images [18, 46, 117, 118, 119, 121, 122]. Some of these systems can display only several hundred images, for instance, Santini and Jain described the El Niño system, which displays images in proximity visualisation [121, 122]. They report that "for practical reasons, an interface can't present more than a small fraction of the images in the database" [122]. El Niño displays at most 300 images, so it clearly does not support browsing of a complete image database. Other researchers claimed that their systems can display thousands of images but unfortunately, the images in most of these systems are overlapping, very small or both, so it is impossible to view them properly. Hence, the systems

are less useful than what they could be [115]. In trying to make the systems more useful, some researchers rearranged the layouts to reduce or even eliminate the overlapping, while attempting to preserve the mental map of the original layouts [84, 117, 150].

Rodden et al. eliminated all overlapping by using a proximity grid to restrict the location of the images to regular grids [115]. Figure 5.1 illustrates the difference between a proximity visualisation and a proximity grid [115]. Although the proximity grid is effective in removing overlapping, it is less efficient in terms of screen real estate usage. As a result, only a small number of images, in the order of 100, can be effectively displayed.

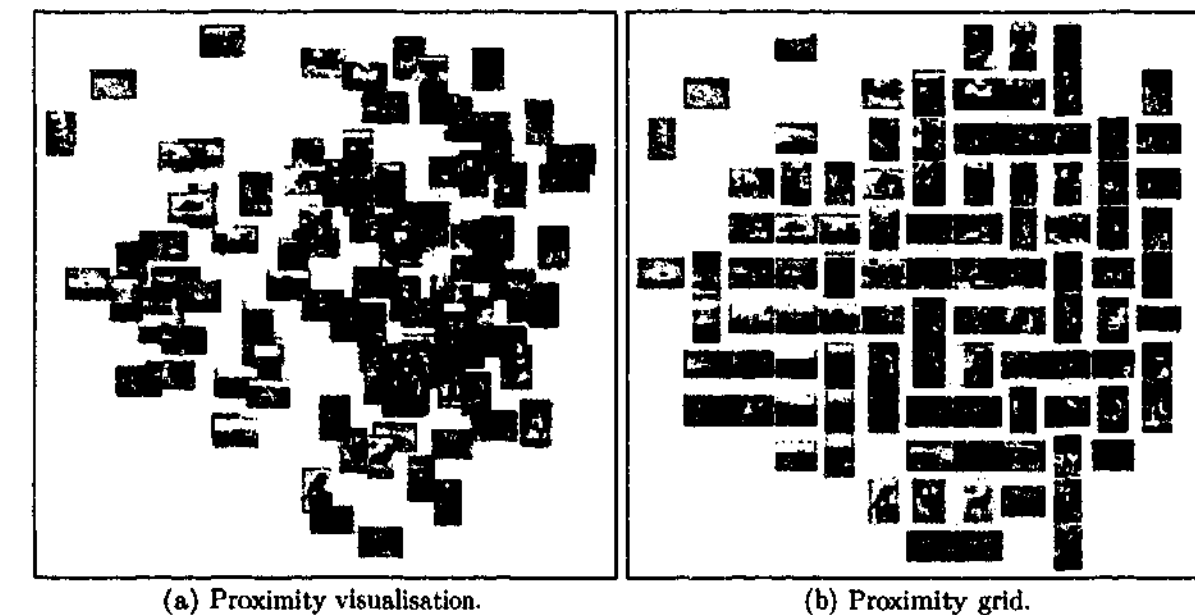


Figure 5.1: An example of using proximity grid to eliminate overlapping.

Moghaddam et al. reduced the overlapping using an algorithm they described as "optimised PCA splat" [84, 82, 83]. It is a non-linear constrained optimisation algorithm: the goal is to reduce overlapping and the constraint is to preserve the mental map of the layout. An example of a layout generated using the "optimised PCA splat" is given in Fig. 5.2 on the following page; note that to further reduce the overlapping, the images in the optimised PCA splat layout are smaller than those of the original layout.

Reorganisation of layouts to reduce or remove node overlapping but still maintain-

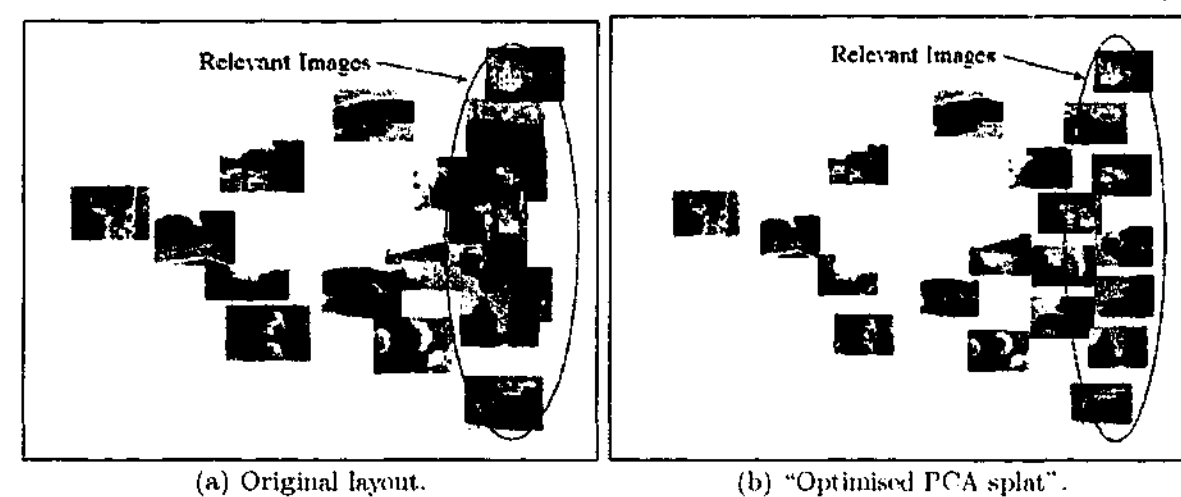
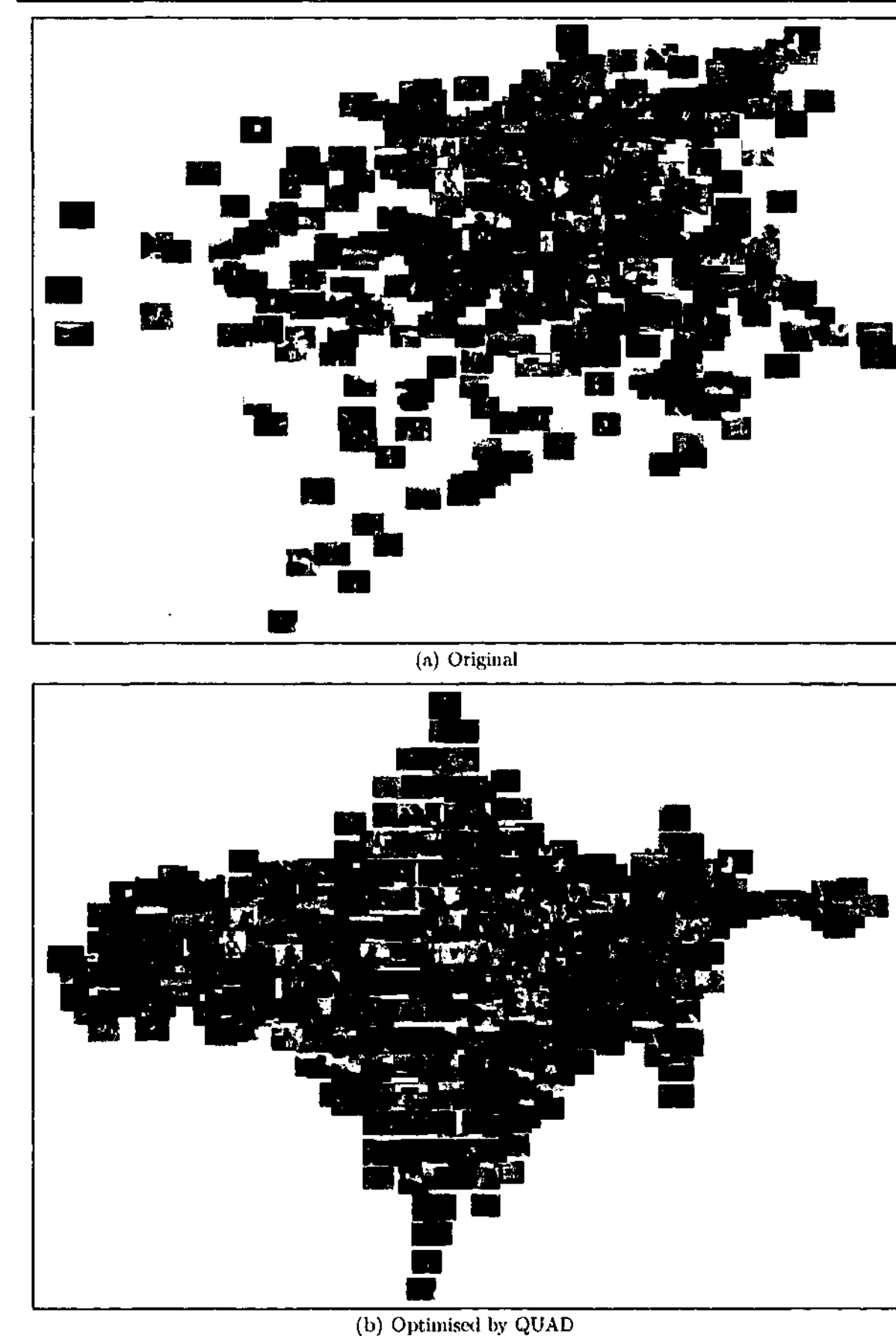


Figure 5.2: An example of using "optimised PCA splat" to reduce image overlapping.

ing the layouts' mental map is an active research area in graph layout [55, 75, 81]. The contemporary algorithm for solving this problem is QUAD, proposed by Marriott et al. [75]. However, neither overlapping reduction nor removal is an ideal solution for solving the overlapping problem because as the number of images increases, the size of the images to be displayed must further decrease; it is meaningless if the images are too small to be useful to users. To demonstrate this, we requested Prof. Marriott's QUAD source code and used it to optimise a layout of 472 images we had generated. The original layout and the QUAD optimised layout are given in Fig. 5.3 on the following page. As expected, fitting the optimised layout into the screen results in some overlapping and to reduce all overlapping, the images have to be even smaller. It is clear that image overlapping removal is not the optimal solution for displaying large databases.

To manage large databases, other researchers use tree or pyramid structures which have multiple hidden levels. At each level, a parent node points to a small number of child nodes, and the number of parent nodes at each level is restricted to 100 [19, 51, 77]. Initially, the systems display only the top most parent nodes. At first, such an approach seems intuitive as it is similar to the *divide and conquer* technique so often employed when dealing with large amounts of data. In the context of visualisation, however, this approach is inappropriate for two reasons. First, such systems fail to provide an overall view of the *entire* database; therefore, it does not support effective browsing. Second,



(b) Optimised by QUAD
Figure 5.3: Visualisation of 472 images.

navigation is confusing as users have to backtrack if they wish to visit the children a different parent.

The system proposed by Pečenović et al. is by far the most complete and powerful [102], as it can display more than several hundred images at a time (Figure 5.4 shows two screen shots of their system). This system has two windows: one for visualisation (in the green box) and another for displaying the selected images in the visualisation window. Users are expected to navigate the database by using the red box in the visualisation window. The images within the red box are then displayed in a separate window and users can zoom in to see the images in more detail. Yet, this system remains unsatisfactory for browsing for the following three reasons. First, it uses a tree structure and the use of this structure is unsuitable for the reasons given above. Second, the images in the visualisation view are too small to be useful for browsing. The use of such a view is common in image editing programs for panning an image, such as in Gimp or ImageMagick [1, 2]. However, it is less appropriate to use this technique for visualisation because the images are too small for navigation. Third, it is a multi window system which lacks an overall context because there is no continuity between windows [131].

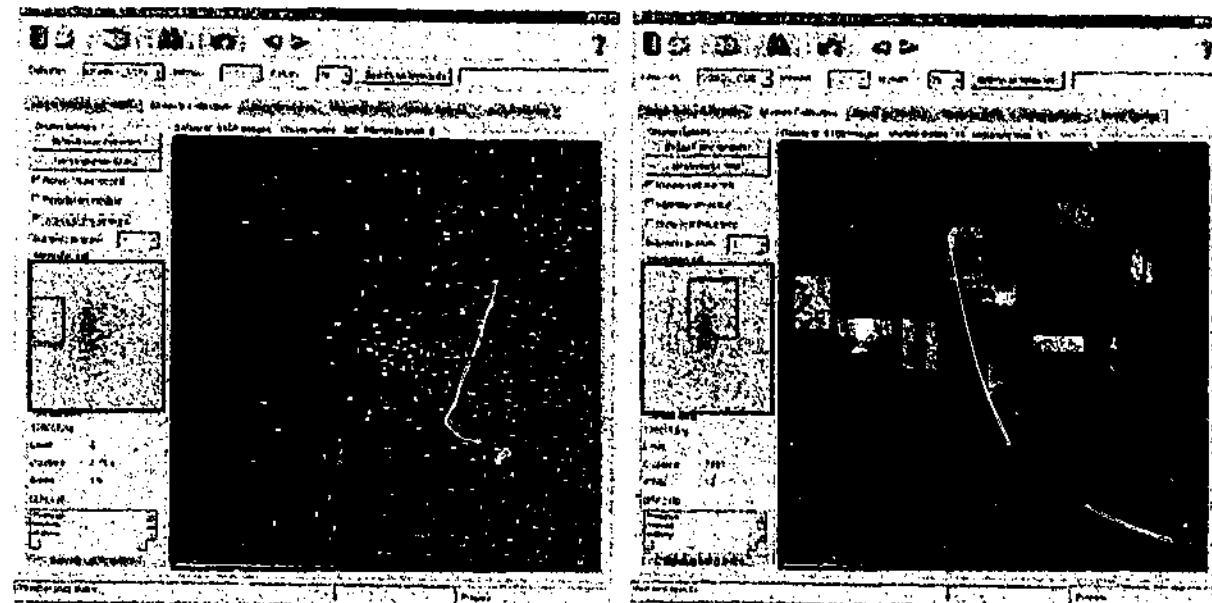


Figure 5.4: Screen shots of the system described by Pečenović et al. Users can zoom and pan to see the images in more details. However, the images in the visualisation view (in green box) are too small to be useful.

5.2 Towards Browsing Large Scale Image Databases

Existing systems are clearly inadequate for browsing large image databases. They all fail because of the degree of overlapping and the techniques used to resolve the overlapping are undesirable for browsing. The design of a visualisation system suitable for browsing large databases requires a more systematic approach, which could be divided into three steps:

1. The use of an algorithm for generating layouts for image databases (Section 5.3).
2. The selection of image features to generate a suitable layout. This is important to ensure that the generated layout is useful for browsing because a random display of images does not facilitate browsing.
3. The introduction of innovative techniques to support browsing of large image databases (Section 5.4).

The first step can be completed using readily available algorithms, whereas the second and third steps are previously unanswered research questions, and therefore, are the research contributions of this thesis. The research into feature selection in the second step is dependent on the type of images (such as colour or texture images), and because we are formulating a browsing framework which must be independent of the type of images, feature selection will only be discussed in Chapter 6 for browsing general colour image databases and Chapter 8 for browsing grey-scale texture image databases. This chapter only concentrates on the first and third step, and it is assumed that a suitable feature for browsing already exists.

5.3 Proximity Visualisation

The very first task to visualise an image database is to select an algorithm for generating a layout required for visualisation. The following sections describe how to achieve this.

5.3.1 Basic Concept

As mentioned earlier, proximity visualisation is the core of image database visualisation. This section explains this concept in more detail. Proximity visualisation, in general, is a type of display in which the proximity of objects is dictated by their similarities and dissimilarities. To use proximity visualisation, the objects must first be represented in feature vectors so that the distances between the feature vectors can be calculated using a dissimilarity metric. The distance between two objects in the display is their Euclidean distance, and it should reflect the dissimilarity of the objects they represent. After selecting the features and dissimilarity metric, it is possible to generate a layout for the objects using dimension reduction algorithms such as Multidimensional Scaling (MDS), Principal Component Analysis (PCA), or optimisation algorithms such as genetic algorithms. They all are equally valid choices [9], but in this research we used MDS algorithms because they are more versatile [54, 157, 158].

5.3.2 Deriving Layouts for Proximity Visualisation Using Multidimensional Scaling (MDS)

The use of MDS algorithms can be described using the following example. Figure 5.5 on the following page shows the location of six points (A to F) on a two dimensional map; finding the distances between all points is easy. Using a ruler, Table 5.1, which lists the distance between any two points, can be easily produced. However, if the question is reversed, given the distances between all points find the layout of the points, the task is no longer easy. MDS algorithms are used to answer such a question, and the layout it produces can then be used to draw a map similar to that of Fig. 5.5, a proximity visualisation. We say similar because the generated layout may have a different orientation to the map in Fig. 5.5 but the relationships of all points in the generated layout are the same as the points in the map.

MDS algorithms find a layout by minimising *Stress*. Several definitions of *Stress* have been given in the literature [9, 25, 28, 88, 117], but in this thesis, it is defined as

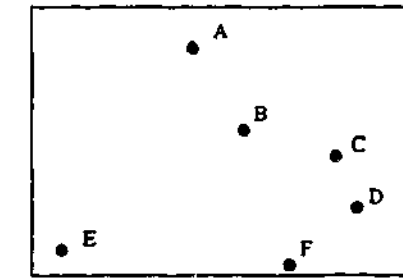


Figure 5.5: A map of point A-F.

	A	B	C	D	E
B	1.0				
C	2.0	1.0			
D	2.5	1.5	0.6		
E	2.5	2.4	3.2	3.4	
F	2.5	1.5	1.3	1.0	2.5

Table 5.1: The distances between all points in Fig. 5.5.

in [25]:

$$Stress = \left[\frac{\sum_{i,j} (d_{ij} - g_{ij})^2}{\sum_{i,j} d_{ij}^2} \right]^{1/2} \quad (5.1)$$

where d_{ij} is the distance between objects i and j in the original dimension and g_{ij} is the Euclidean distance between point i and j in the layout. *Stress* measures how well the layout in the low dimension represents the distances between objects in the high dimension. A lower value of *Stress* is more desirable, with the ideal value being 0 which indicates a perfect representation. *Stress* as defined in (5.1) penalises mismatches of objects with short distances more heavily than mismatches of objects with long distances. In other words, it is more important to represent objects with short distances more faithfully than objects which are far apart. On the choice of the MDS algorithm, classical MDS is effective but has high complexity i.e. in the order of $O(N^3)$. More recent MDS algorithms are incremental and therefore more efficient [25, 88]. In this research, we used the hybrid method described by Morrison et al. which has an overall complexity of $O(N)$ [88].

In the hypothetical example given above, the distances between the features (the points in the table) are in high dimension, while the distances between the points on

the map or layout are in low dimension. This means the number of dimensions for the points in the high and low dimensions happens to be the same, so it is possible to generate a layout which can faithfully represent the distances between all points. In real world applications, the number of dimensions for the features are often higher than those in the layouts, so it is impossible to represent the distances faithfully in the layout, as illustrated in the following example. If there are only three equidistance objects, they can be represented in a two dimension layout but if there are four equidistance objects, it is impossible to represent the distances faithfully in two dimensions: at least three dimensions are needed. This problem is not unique to MDS algorithms; it is common to all dimension reduction algorithms. The more important issue is how to select a suitable set of features so that the MDS algorithm can generate a meaningful layout. The choice of the right features is therefore of prime importance in determining the usefulness of the display.

Because feature selection is dependent on the type of image databases and we are proposing a framework independent of the type of image databases, the issues related to feature selection will be explored in Chapter 6 for browsing general colour image databases and Chapter 8 for browsing grey-scale texture image databases. For now, we assume that a feature suitable for browsing an image database already exists and a layout has been generated using the process described above. In the next section, we investigate techniques suitable for browsing large scale image databases.

5.4 eyeMap, an Image Browsing Framework for Large Image Databases

Browsing large scale image databases are challenging for two reasons. First, many images need to be drawn on a limited screen area, thereby introducing more image overlapping. Second, the system's response time is proportional to the number of images: the more images, the longer response time. For users to feel that their actions have direct impact, the system must respond within 100 ms [126]. Previously developed

systems mainly displayed only a small number of images, about 200, so image overlapping and response time were not major issues. Other systems that do display more images provide no useful visualisation because either there is too much overlapping or the images are too small, only several pixels wide.

The central principle of Shneiderman's visual design guidelines is "overview first, zoom and filter, then details on demand" [126]. The proposed image browsing framework, eyeMap, is designed with this guideline in mind to ensure that it is useful for browsing large scale image databases. It is possible to address the two problems described above and still adhere to the guideline by:

- clustering the images to reduce the number of images to be displayed at any one time;
- removing image overlapping; and
- allowing users to hide images which are definitely not the target image.

The following sections describe each strategy in detail. The approach taken in eyeMap for dealing with large numbers of images is innovative because it is efficient in using screen real estate and intuitive for users to manipulate.

5.4.1 Image Clustering

To facilitate the browsing of large scale image databases, eyeMap reduces the number of images to be displayed at any one time by clustering the images. The image closest to the centroid of the cluster is considered the representative image, and initially only these images are displayed. Note that this approach is different from the tree or dynamic structures which use multiple hidden levels. eyeMap uses only one hidden level, and the user interface described in the next section makes eyeMap easy and intuitive to use. In summary, the steps for producing the final visual layout are:

1. Extract feature vectors from images.

2. Cluster the images using the k-means clustering algorithm.
3. Find the representative image of each cluster (the image closest to the cluster's centroid).
4. Create a visual layout for the representative images.
5. Find the visual layout for non-representative images for each cluster. These images are simply placed around the representative image of each cluster.

Initially, eyeMap only displays the representative images and, only if the representative images are in the area of interest (focal region) it displays the corresponding non-representative images. Users can intuitively specify the focal region using Distortion Oriented Displays, which will be explained in the next section.

5.4.2 Removal of Image Overlapping

The second strategy to enable browsing of large scale image databases is to provide intuitive tools so that users can remove image overlapping. This section describes how eyeMap can effectively remove the image overlapping and still maintain efficient use of screen real estate using two related approaches: using Distortion Oriented Displays (DOD) and displaying a small number of images linearly in a separate window, with the help of DOD.

5.4.2.1 Distortion Oriented Displays (DOD)

Apart from removing image overlapping, DOD allows users to *look at the trees without losing the sight of the forest* by displaying different levels of detail on the same screen using the concept of focal and context regions. The focal region is the area of interest; it is the *focus* or the trees, and therefore shows more detail. The context region is the *overview*; it is the context or the forest, and therefore shows less detail. A typical use of DOD is in geographical information systems (a research area involving electronic maps), where the spectrum of details from the overview to the focus region progresses

from state boundaries, then town boundaries and, finally, street boundaries as shown in Fig. 5.6 [131]. To adapt DOD for eyeMap, we modified this spectrum of details to progress from the display of representative images in smaller size to the display of corresponding non-representative images in bigger size.

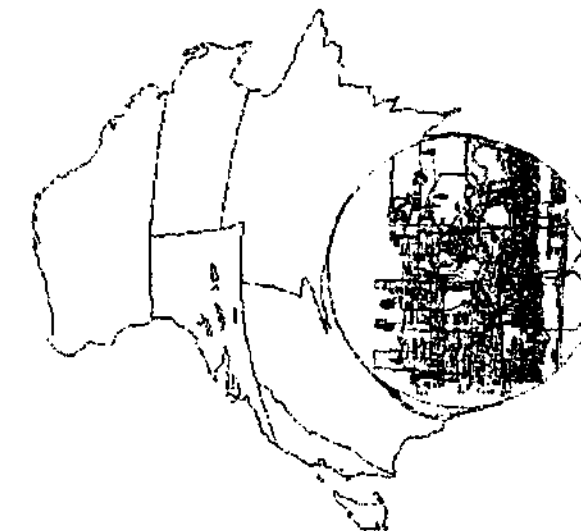


Figure 5.6: A sample of a geographical information system showing the map of Australia. In the context region, only the state boundaries are visible, whereas in the focal region (within the circle), town and street boundaries are visible.

Spence and Apperley proposed the first DOD system using the one dimensional bifocal display in 1982 for interactive computer applications. It was one dimensional because the view could only be magnified or demagnified in one dimension [132]. Figure 5.7 on the following page shows a circle data set (a) undistorted and (b) distorted using a bifocal display with a distortion factor of four. The red cross is the focal point, which is simply the centre of a focal region; the region within the red rectangle is the focal region; and the region outside the rectangle is the context region. A distortion factor of four means that the distances between the circles in the focal region are now four times as far as they would normally be had they been undistorted. For a predefined window size, the magnification in the focal region forces demagnification in the context region to maintain the same overall size. Since the focal region is magnified, it is now possible to show more detail, which in this case simply means bigger circles.

Leung extended the bifocal display into two dimensions for displaying the London underground map [59], and an example of 2D bifocal display using the circle dataset

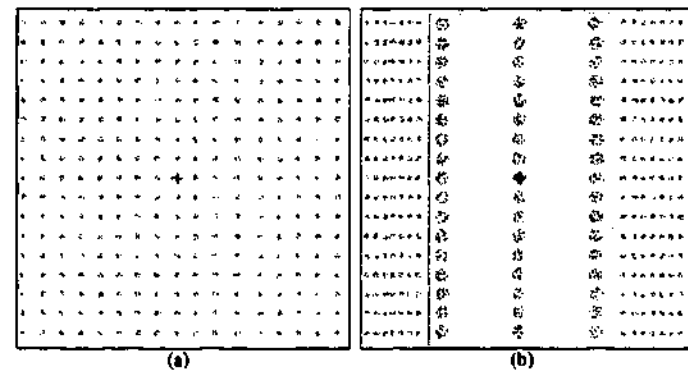


Figure 5.7: (a) Undistorted display of circle data set (b) Bifocal Distortion display of the data set.

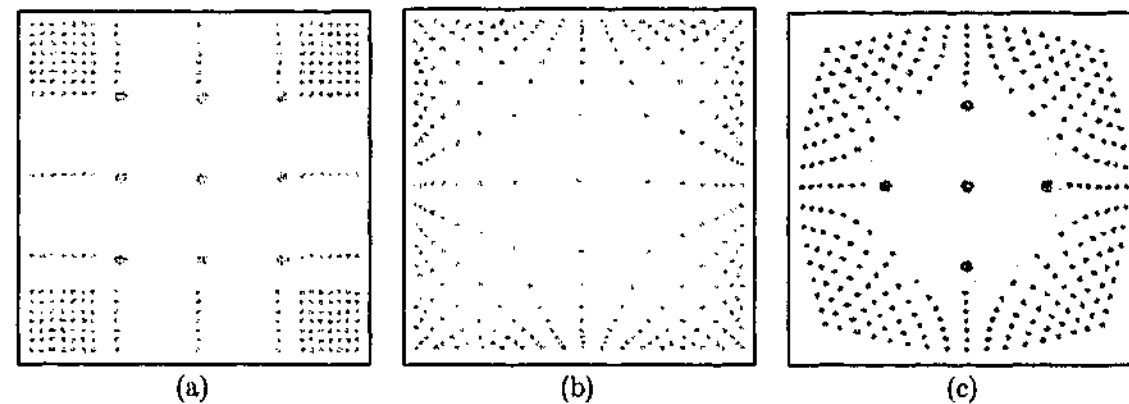


Figure 5.8: The circle dataset in (a) 2D bifocal (b) fisheye and (c) frustum.

is shown in Fig. 5.8(a). The area within the red rectangle is the focal region, and the circles within this area are bigger than the circles outside. Unfortunately, this distortion also magnifies the context region. The context region to the east and west of the focal region is magnified in the Y axis, while the region to the north and south of the focal region is magnified in the X axis. This contradicts the idea behind DOD because the purpose of the context region is to provide an overview, and any magnification wastes screen real estate. Because of this unnecessary magnification, it is difficult to maintain a useful context when using bifocal display. In eyeMap, it is important to use a DOD technique which can maintain a meaningful context because when the size of the database is large, many images will overlap and a high distortion factor (up to sixty) is needed to reduce the overlapping.

Other DOD techniques in the literature include fisheye and its many variations, frustum, perspective wall, document lens, table lens and hyperbolic browser [41, 56, 70, 109,

112, 123, 131]. Leung and Apperley provide a detailed review of DOD techniques [60]. Of all these techniques, fisheye is most popular but it has a major problem: its context region cannot maintain a useful overview with large distortion factors, typically ten or over [131]. This problem arises because in fisheye, there is no clear distinction of when the focal region ends and when the context region starts. As a result, even areas which are not of interest to users are unnecessarily expanded, so the context region becomes very small. An example of the fisheye display with a distortion factor of four can be seen in Fig. 5.8(b), which was implemented in polar coordinates by transforming (r, θ) into (r', θ') [123]:

$$r' = r_{max} \frac{(d+1) \frac{r}{r_{max}}}{d \frac{r}{r_{max}} + 1} \quad (5.2)$$

where r is the original distance from the focal point, r' is the transformed distance from the focal point, d is the distortion factor, and r_{max} is the maximum possible value of r in the direction of θ . Note that the value of θ remains unchanged, $\theta' = \theta$.

Another type of DOD in the literature is the frustum technique. The display of the same dataset and same distortion factor in frustum is given in Fig. 5.8(c). Unlike fisheye and 2D bifocal, in frustum only the focal region is magnified. The calculation of the frustum transformation is in polar coordinates, and the transformation from (r, θ) to (r', θ') is expressed as:

$$r' = \begin{cases} r \cdot M_{Rf} & \text{if } r \leq R, \text{ focal region} \\ R \cdot M_{Rf} + (r - R) M_{Rc} & \text{otherwise, context region} \end{cases} \quad (5.3)$$

$$\theta' = \begin{cases} \theta & \text{if } r \leq R, \text{ focal region} \\ \frac{r'}{r \cdot M_{Rc} - 1} & \text{otherwise, context region} \end{cases} \quad (5.4)$$

where

- R = focal area radius
 M_{Rf} = focal area magnification
 M_{Rc} = context area demagnification
 r = original radial distance to the focal point
 r' = transformed radial distance to the focal point

Unlike fisheye, frustum distinctly separates the focal region from the context region, so the context region still provides a useful overview even at a very high distortion factor (up to one hundred) [131].

The original frustum projects the dataset onto a square display but monitors are not square, so some screen real estate is wasted. To efficiently use screen real estate, the display is scaled, and as a result, the focal region is now elliptical rather than circular as shown in Fig. 5.9.

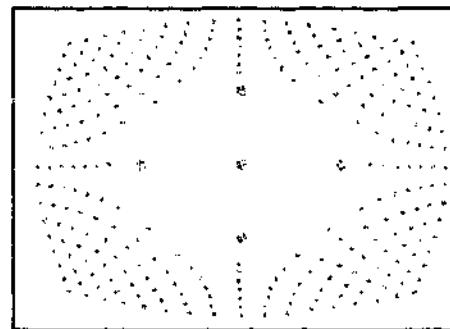


Figure 5.9: A frustum display of the circle dataset scaled to make full use of the screen real estate.

Most image formats are not scalable: the size of the decoded image is the same as the size of the coded image. To obtain a smaller image, it is necessary to subsample after a complete decoding. This means that the processing time is longer if we require a smaller image size than the one coded. JPEG 2000 is a scalable image format: the size of the decoded image may differ from the size of the coded image. If we require a smaller image, then we decode only up to the required size [146]. In other words, the processing time is now proportional to the required image size. For this reason, it is recommended an implementation of eyeMap uses the JPEG 2000 image format.

5.4.2.2 Linear Display

The second method of removing image overlapping is by displaying a small number of images linearly. The use of frustum has another advantage. Because it distinctly separates the focal region from the context region, it is possible to display all the images in the focal region linearly in a separate window with the overlapping removed. In this window, users can also enlarge an image so that they can see the image in full size. The number of images displayed in this window is relatively small in comparison to the size of the entire database, so users only need to linearly search from a small number of images. For this reason, displaying these images linearly should have no effect on browsing, search effectiveness and efficiency.

5.4.3 Searching for a Target Image by Elimination

The last strategy used by eyeMap to deal with large scale image databases is to allow users eliminate irrelevant images. Users can hide the images they have seen and display the hidden images again if necessary. By eliminating images they have seen, users can reduce the number of images to be displayed and avoid looking at images which they have inspected; therefore, this functionality helps them to further narrow down their area of search.

5.4.4 Summary of eyeMap Specification

The specification for eyeMap is now complete and includes the following:

- Cluster images based on visual similarity; each cluster has a representative image.
- Calculate a layout for all representative images, then assign the positions of non-representative images based on the coordinates of their representative images.
- Display the images based on the layout using DOD.
- Remove overlapping by either changing the distortion factor of the DOD or by opening a separate window.

- Hide and unhide images.

5.5 Applications of eyeMap

This section describes two potential applications for eyeMap: browsing and initiating a visual query. Because eyeMap is a browsing framework, it may be difficult to imagine why an implementation of eyeMap is suitable for these two tasks. It is, at this stage, appropriate to show a layout generated using an implementation of eyeMap for general colour images developed in Chapter 6 (see Fig. 5.10 on the following page). It can be seen from the layout that the arrangement of images is systematic, in that they are grouped by colour similarity. We can also see that the context region displays only the representative images in smaller size, whereas the focal region (within the ellipse) displays the representative and the corresponding non-representative images in bigger size.

5.5.1 Browsing

The use of eyeMap for browsing is obvious. eyeMap can display the whole database and users simply browse through the collection by going directly to the area of interest. As they browse, they can change the distortion factor to remove the overlapping in the focal region or display all images in the focal region linearly without any overlapping at all. Of course, there is a limit to the size of the image database that can be displayed; for instance, it would be difficult to display all images in the Internet. In fact, when it comes to searching images on the Internet, it is a great challenge even for image retrieval.

eyeMap can also be useful for displaying search results where historical text annotations are already available. It is a practice within the concept-based image retrieval community to coarsely annotate the images and to then issue textual queries [6, 74]. Because of the coarse categorisation, a query is likely to return many images which are often displayed linearly; thus, browsing is inefficient. To facilitate efficient browsing,

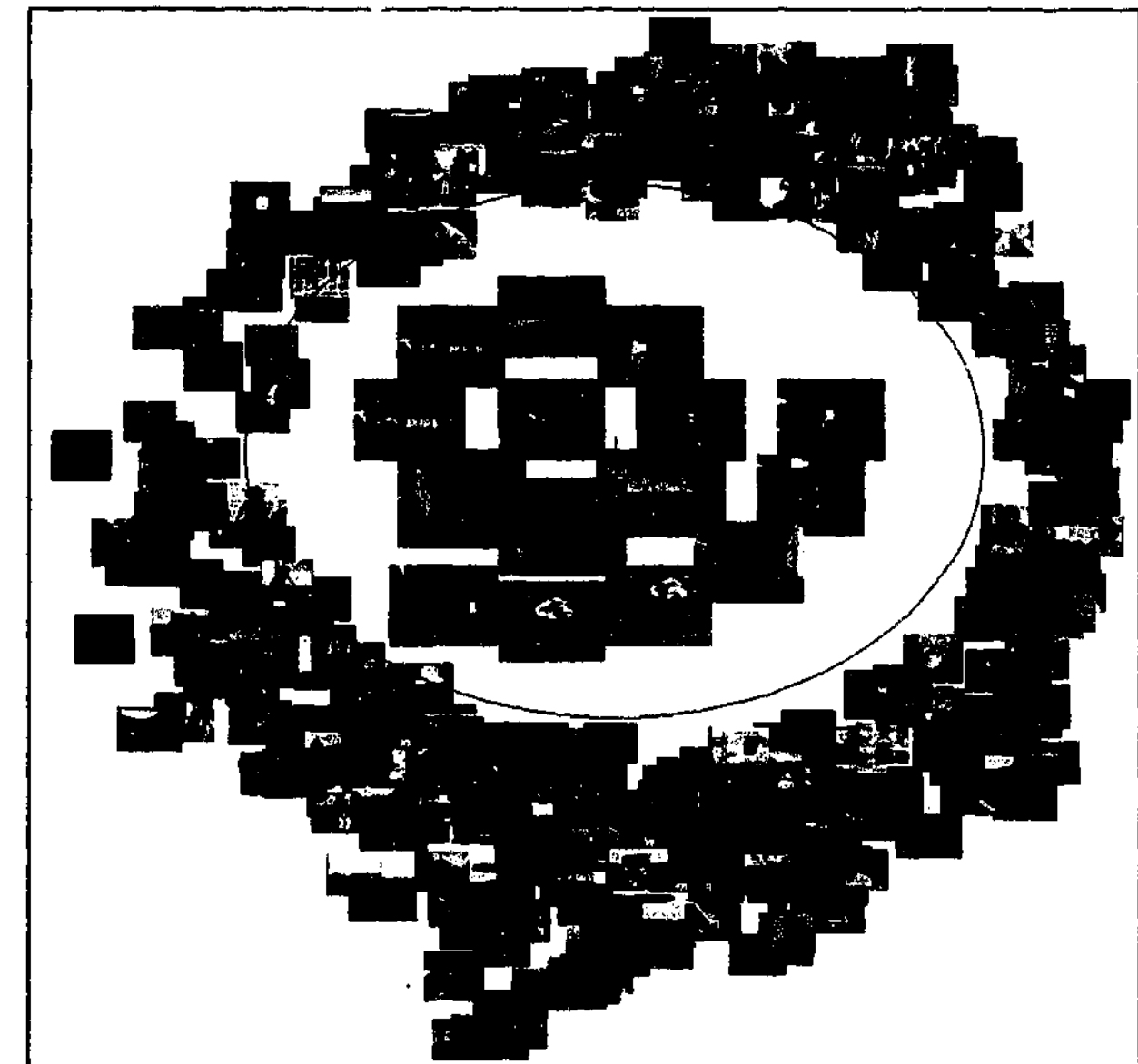


Figure 5.10: A visualisation of the CCD database using frustum display with a distortion factor of 52. Images within the focal region, the red ellipse, are displayed in more detail (larger). This layout is generated by colour-based eyeMap.

the images could be displayed using eyeMap. Note that some systems reviewed earlier, such as the one described by Rodden et al., can also be used for this purpose but because of the inefficient use of screen real estate, the number of images to be displayed at a time is limited to at most one hundred [115]. With eyeMap, the only limitation is the time taken to find a layout. The efficiency issue is beyond the scope of this thesis but the research conducted in Chapter 6 and Chapter 8, to a certain extent, contribute to speeding up this process.

The implementation of eyeMap is a powerful browsing tool but it can also be integrated with a CBIR: eyeMap as the browsing engine and the CBIR as the retrieval engine. The application of eyeMap is then to find a sample image to initiate a visual query. The next section illustrates this application.

5.5.2 Finding a Sample Image to Initiate a Visual Query

As previously stated, CBIR has been an active research area in computer vision for more than a decade yet it has been of little practical use mainly because there is no convenient way of initiating a query. It is also known as the Page 0 problem.

Existing methods of initiating a query include text searching of annotated images, specifying the colours (and the percentage of each colour), sketching and query-by-example (QBE) [11, 153]. They fail for the following obvious reasons. Annotating the images defeats the purpose of a CBIR system. Specifying the mix of colours is most unnatural as that is not how humans remember images. Sketching a query is impractical as users are not only expected to sketch but also have an intimate knowledge of the internal functions of the system, including the feature extraction algorithms, to sketch successfully. QBE is by far the most accurate and popular method but is useless when users do not have a sample image. Locating a sample image is time consuming as users have to go through the images in the database either linearly or randomly until a target image is found.

The design of eyeMap is modular in that both the browsing and retrieval engines are independent of each other, so the retrieval engine can be added or even changed without

affecting the browsing engine. Integrating eyeMap with a CBIR system means we can now solve the Page 0 problem because users can first browse the area of interest, and then issue a visual query using the sample image they found. The following scenario illustrates how eyeMap could be used for solving the Page 0 problem.

Alice is looking for an image and she has a pretty good idea what type of image she wants. She starts using eyeMap which shows a proximity visualisation display of the database. Now, Alice has an idea what the database looks like, and can use the display like a map. The images are small (though visible enough) and are overlapping, but she can still eliminate areas which look totally different from the image she has in mind. So, she moves the focal region away from those areas and starts exploring areas which look more promising. When the representative image is in the focal region, it is enlarged and other images in the cluster are displayed as well. To remove the overlapping, Alice increases the distortion factor or opens up another window which will display all images in the area she is interested in with the overlapping removed. Alice is sure that the image she is interested in is not there, so she hides them. She continues exploring and "bingo", that's it. Alice sees the image she wants and several similar images nearby. To find out if there are more similar ones, she issues a visual query using the image she has found as an example.

The above scenario was written based on a real user's experience when looking for an image in the MPEG-7 CCD of 5466 images using eyeMap integrated with a CBIR.

5.6 Conclusions

In this chapter, we proposed and formulated eyeMap - an image browsing framework - for browsing large image databascs. The philosophy behind the design of eyeMap was to enable users to transfer their browsing behaviour in daily life, such as browsing merchandise items, into browsing images. Daily browsing activities are possible and, to a certain extent, enjoyable because someone has organised the items systematically and shoppers can access the items. The development of eyeMap closely followed this daily

browsing paradigm which dictates that the items must be organised systematically and the embedded interaction is intuitive.

To satisfy the first requirement of this paradigm, it is essential to ensure that the images are organised systematically by studying the layouts generated. The evaluation of the layout depends on the type of image databases but because eyeMap is a browsing framework independent of the type of image databases, it will be covered later in this thesis. In this chapter, however, we describe Multidimensional Scaling (MDS), a technique useful for deriving a layout for browsing. Chapter 6 discusses the evaluation of colour features to establish which one is more suitable for browsing colour image databases, while Chapter 8 establishes which texture descriptor is more suitable for browsing grey-scale texture image databases.

The second requirement in the paradigm demands that users can interact with the images intuitively. To comply with this requirement, we first solve the problems associated with browsing large image databases by clustering the images. Then, with an intuitive user interface, users can access all images by using a DOD technique known as the frustum display. DOD ensures that users can *look at the trees without losing the sight of the forest* by displaying the images of interest in the focal region and other images in the context region (overview). In addition, by using the focal region, users can select some images to be displayed linearly (with the overlapping removed) in a separate window or to be hidden.

The specification for eyeMap is complete and an implementation of eyeMap will result in a fully functional system. A complete implementation of eyeMap is a powerful browsing tool, but when integrated with a CBI system, it can be used to solve the Page 0 problem i.e. the problem of initiating a visual query without a sample image. The next chapter discusses an implementation of eyeMap for browsing colour image databases (colour-based eyeMap) including the selection of a feature suitable for browsing.

Colour-Based eyeMap: Browsing Colour Image Databases

The previous chapter describes eyeMap as a concept for browsing any type of large scale image databases on the assumption that suitable layouts for the image databases already exist. As mentioned in the previous chapter, the issue of finding a suitable feature for browsing image databases is unresolved and this is the contribution of the research in this chapter. The purpose of this chapter is to establish which colour feature is suitable for browsing general colour images by evaluating several colour features. After resolving which features are more suitable for browsing, we develop a colour-based eyeMap, a full implementation of eyeMap for browsing large scale colour image databases. The colour-based eyeMap is integrated with a CBIR system so it can also be used to find a sample image to initiate a visual query.

6.1 Feature Selection: Evaluation Method and Design

In generating layouts for colour image databases, it is essential that they are meaningful to users because a random display does not promote browsing. As defined in the previous chapter, browsing is made up of two parts: visualisation and navigation. Visualisation is the part of browsing responsible for the layout of the images. The purpose of this section is to determine which feature is more suitable for visualisation by studying the layouts generated from different features using objective and subjective evaluations. All layouts studied in this chapter were generated using the MDS algorithm

as described in the previous chapter.

General colour images could be described using their colour, shape and texture features; however, the use of shapes or texture requires image segmentation which is unreliable when automated and time consuming when segmented either manually or semi-manually. As mentioned in Chapter 2, colour-based feature extraction methods can effectively extract useful features from an image without image segmentation, and for this reason they remain popular. Rogowitz also found that colour features are often sufficient to capture semantic information of images [116]. It was found in Chapter 4 that I-auto, a feature vector which incorporates the spatial relationships of colours, is more effective than colour distribution methods. Nevertheless, the experiments in this section were restricted to colour distribution features. I-auto is a complex feature vector made up of more basic feature vectors, so the findings from the experiments using basic feature vectors were used as a guide for deciding whether to use more sophisticated feature vectors.

Unlike previous chapters, all experiments in this chapter were conducted in only CCD. This is not a disadvantage because the research from previous chapters suggest that most results in CCD can be generalised to PC³.

6.1.1 Evaluation Criteria

A layout is useful for visualisation if perceptually similar or relevant images are located close to each other and perceptually different ones are located far apart. The first condition can be measured using a variation of PR graphs called spatial PR graphs but the second one can only be evaluated qualitatively by visual inspection. In retrieval, the effectiveness of a feature only requires a comparison of PR graphs. In visualisation, evaluation includes spatial PR graphs and a visual inspection.

Stress (5.1) is the function that the MDS algorithm optimises, so at first it seems ideal to use *Stress* as another objective measurement. Often, it is used to measure the quality of two layouts [25], but for this study it must not be used, as *Stress* is only appropriate for measuring the quality of layouts generated by *different algorithms*

using the *same feature*. This study evaluated layouts generated using *different features*, so a comparison of *Stress* values is meaningless.

6.1.1.1 Spatial Precision and Recall Graphs (Spatial PR Graphs)

PR graphs described in Chapter 3 are frequently used to measure the retrieval effectiveness of a CBIR system and can be adapted to measure how closely relevant images are clustered in the two dimensional (2D) layout [57, 114]. The adaptation process involves first of all, generating a linear ranking from the 2D layout by calculating the Euclidean distance between all images to the reference image, which is equivalent to a query image in retrieval. Then, the images are sorted with increasing distance from the reference image. Finally, the precision at each recall value is calculated: it is now the same as calculating PR graphs (3.1). To differentiate these graphs from the traditional ones, they are called spatial PR graphs.

The interpretation of spatial PR graphs for visualisation is similar to that of PR graphs for retrieval in that the feature with the higher precision at the same recall value is more desirable. Because of this, initially it would seem that the feature with the highest *retrieval* effectiveness is the most suitable one for visualisation. The validity of this assumption was evaluated by comparing the spatial PR graphs and PR graphs generated using the 50 images (same as the previous chapters) and all their relevant images as references or queries respectively, in total 387 images.

The reason for using all 387 images is that *spatial* PR graphs are very sensitive to the location of the reference image, as illustrated in Fig. 6.1 on the following page. Both displays have exactly the same layout, and differ only in the reference image: image 1 is the reference image in layout A, but image 3 is the reference image of layout B. However, the values of the spatial precision at the same recall for both layouts are very different; for example, when recall is 0.5, the spatial precision for layout A is $\frac{2}{3}$ but, for layout B, it is $\frac{2}{6}$. This problem can be avoided by using each relevant image in turn as the reference image, so the spatial PR graphs must include all the relevant images as references. Because of this, the PR graphs must also include these images as queries.

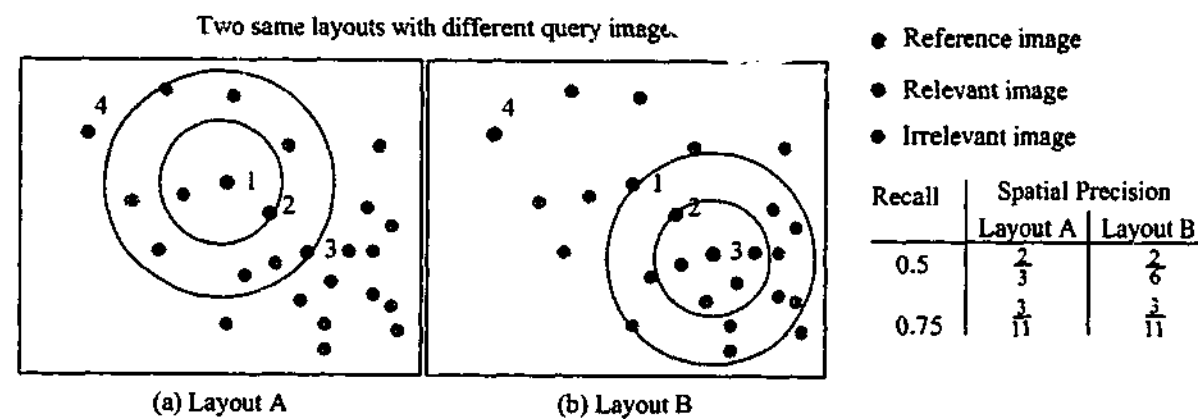


Figure 6.1: The spatial PR graphs are highly sensitive to the location of the reference image.

6.1.1.2 Visual Inspection

Research in proximity visualisation often requires a visual inspection to find out if the generated layout has a meaningful interpretation [9, 88]. Spatial PR graphs can only capture how relevant images are organised, but not how the irrelevant or dissimilar ones are organised. To visualise a collection of images, it is equally important to see how dissimilar images are organised; however, this information can only be gathered using visual inspection. A layout has a meaningful interpretation if it is not random. Because the feature vectors capture colour content, the generated layout is not random only if it, overall, conveys useful information on the colour arrangement. It means that given two layouts, the one which appears less random is considered more contextually meaningful, therefore, more suitable for visualisation.

6.1.2 Generating Layouts: Colour Features and Their Parameters

To determine which colour feature is more suitable for visualisation, we evaluated four colour features: colour histogram, colour moments, EMD-based colour signature (EMDcs) and cumulative histogram. (Please see Chapter 2 for a description of these features.) All except cumulative histogram have been used for visualisation - EMDcs in [117], colour moments in [84, 102, 122] and colour histograms in [46, 102, 113]. The evaluation of histogram has another importance because it indirectly indicates if I-auto is suitable for visualisation: I-auto also uses histogram. Because the size and complexity of cumulative histogram is comparable to that of histogram, it was interesting to

see if cumulative histogram could be used to generate a meaningful layout.

In database visualisation, the layouts will mostly be generated off-line but it was interesting to see if it is possible to generate a layout more efficiently without sacrificing the quality. EMDcs has been extensively studied in [117], and it was found that the computational cost of this feature prohibits its use for generating the layout on-line. Finding a layout is a computationally expensive process because the MDS algorithm requires many iterations and a single iteration involves $\frac{N(N-1)}{2}$ evaluations of the dissimilarity metric, where N is the number of images.

All colour features were implemented in the HSV colour space. For histogram and cumulative histogram, the colour space was quantised into 162 bins ($18 \times 3 \times 3$) as recommended in Chapter 3. For colour moments, no quantisation is necessary and the weight used was the one described in Chapter 2.

6.2 Results and Discussion

To determine which colour features are more suitable for visualisation, this section discusses the results of the four layouts using the two evaluation criteria: spatial PR graphs and visual inspection.

6.2.1 Spatial PR Graphs

The spatial PR graphs of the four layouts are given in Fig. 6.2 on the following page. To show that the image arrangements in all layouts were not organised by chance, they were also compared with a randomly generated layout. We can clearly see that the image arrangements in these four layouts did not happen by chance, as they all have higher spatial precision than the randomly generated layout. We can also see that that cumulative histogram and colour moments are most effective i.e. more relevant images are located closer to each other.

As stated earlier, it seems fair to assume that features rated highly in the spatial PR

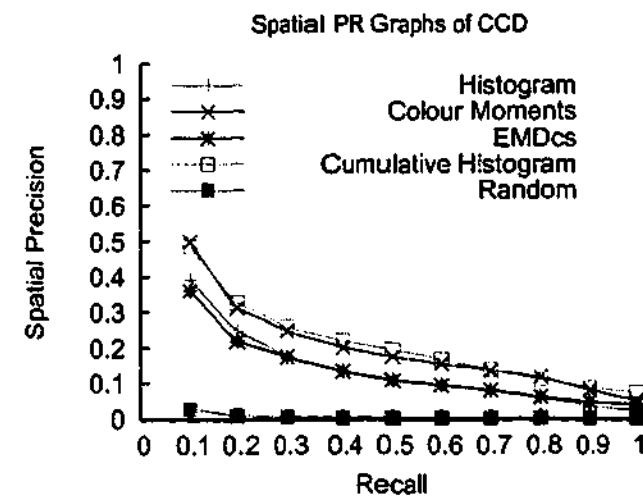


Figure 6.2: The spatial PR graphs of the four colour features on CCD and a randomly generated layout.

graphs should also be rated highly in the PR graphs. Figure 6.3 shows the PR graphs of the four features. Surprisingly, cumulative histogram and colour moments were rated poorly in the PR graphs. This shows that the assumption is untrue: it appears that the advantage offered by colour histogram during retrieval is not transferred to the 2D layout. The conclusion above is counter intuitive but we can explain it by discussing the relationships between PR graphs and spatial PR graphs.

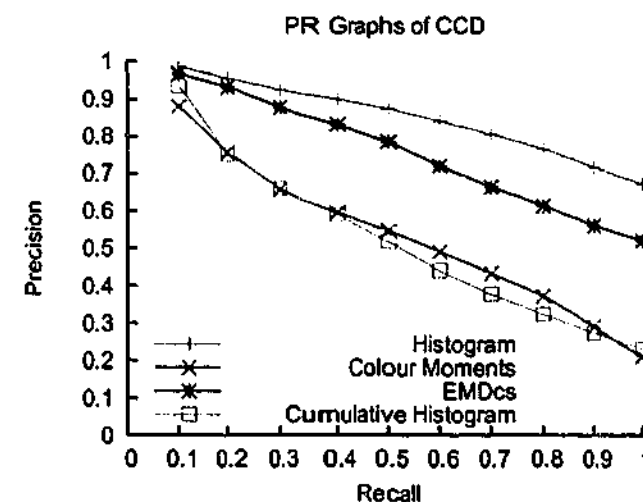


Figure 6.3: The PR graphs of the four colour features on CCD.

6.2.2 Relationships between PR Graphs and Spatial PR Graphs

The relationships between both types of graphs can be established by understanding the similarities and differences of retrieval and visualisation (see Table 6.1 below). Their

Similarities	
1	The feature vectors of the images are the same.
2	The distances between the feature vectors are calculated using the same dissimilarity metric.

Differences		
	<i>Retrieval</i>	<i>Visualisation</i>
1.	Purpose To retrieve n most relevant images, where $n \ll N$.	To browse N (all) images.
2.	Process The retrieval engine only needs to calculate the distance between the query image to every other image in the database. It is a <i>one to many</i> relationship and the degree of relationship is $N - 1$.	The MDS algorithm needs to calculate the distance between every image in the database with every other image in the database. It is a <i>many to many</i> relationship and the degree of relationship is $\frac{N(N-1)}{2}$.
3.	Information The retrieval engine only needs to know how similar two images are and if the images are different it is immaterial how different they are.	The MDS algorithm needs to know how similar two images are and if they are different, how big the difference is.

Table 6.1: Similarities and Differences of Retrieval and Visualisation.

similarities are obvious and, on the surface, they indicate that results from the PR graphs should correspond to the spatial PR graphs. However, experimental results show otherwise. This finding is explained by examining their differences.

In a layout, the degree of relationships for an image with other images in a database is defined as $\frac{N(N-1)}{2}$ (many-to-many). Because spatial PR graphs are calculated by transforming the 2D distances in a layout into a linear ranking, the implication of a many-to-many relationship is that spatial PR graphs are influenced by not only the distance between the reference image to other images but also by the distances between every image to every other image. In contrast, in retrieval, the degree of relationships for an image to other images in a database is defined as $N - 1$ (one-to-many). Thus, PR graphs are only influenced by the relationships of a query image to every other

image in the database. This explains why the results from traditional PR graphs are not directly translated into spatial PR graphs.

6.2.3 Visual Inspection

Figures 6.4(a) to (d) on the following pages show the layouts generated using the four colour features. For colour histogram, the layout appears random in many areas; for example, green images are found in several parts of the layout. Besides that, the display also appears cluttered around the central region.

For colour moment (see Fig. 6.4(b)), the layout appears somewhat random; for example, blue images can be found on the top right and across the centre of the display. It is interesting to note that colour moment is rated highly in the spatial PR graphs. This confirms the observation made earlier that spatial PR graphs do not capture all information about contextual meaningfulness.

For EMDs (see Fig. 6.4(c)), the layout appears less random; dark blue images are concentrated mainly in one corner, green images in another corner. It is also more spread out compared to colour histogram but the centre of the display has a big hole, thus wasting screen real estate.

For cumulative histogram (see Fig. 6.4(d)), the layout appears less random than the histogram and colour moments. However, when compared with EMDs, it appears a little cluttered. In short, the layouts generated from EMDs and cumulative histogram are less random.

Visually inspecting a layout generated from large databases is less than ideal because the degree of overlapping is so great that most images are invisible. This problem can be avoided by also inspecting layouts generated using a small subset of the database. To ensure that there is a wide range of images with enough visual similarity, the 387 images used for generating the spatial PR graphs are used for generating another four layouts (one using each feature). These four layouts are shown in Fig. 6.5 on the following pages.

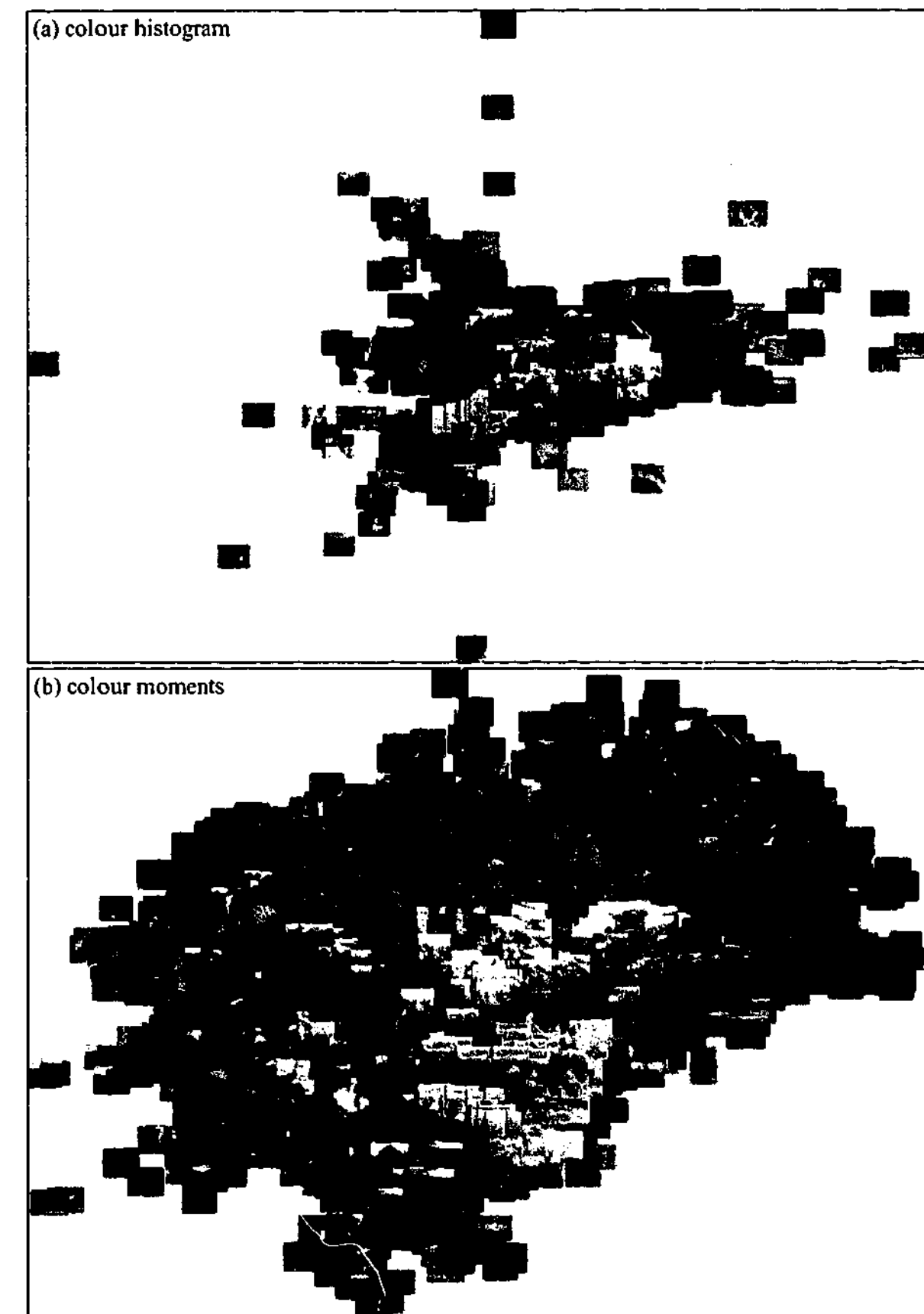


Figure 6.4: Generated layouts using all images in CCD - 5466 images...continued on next page.

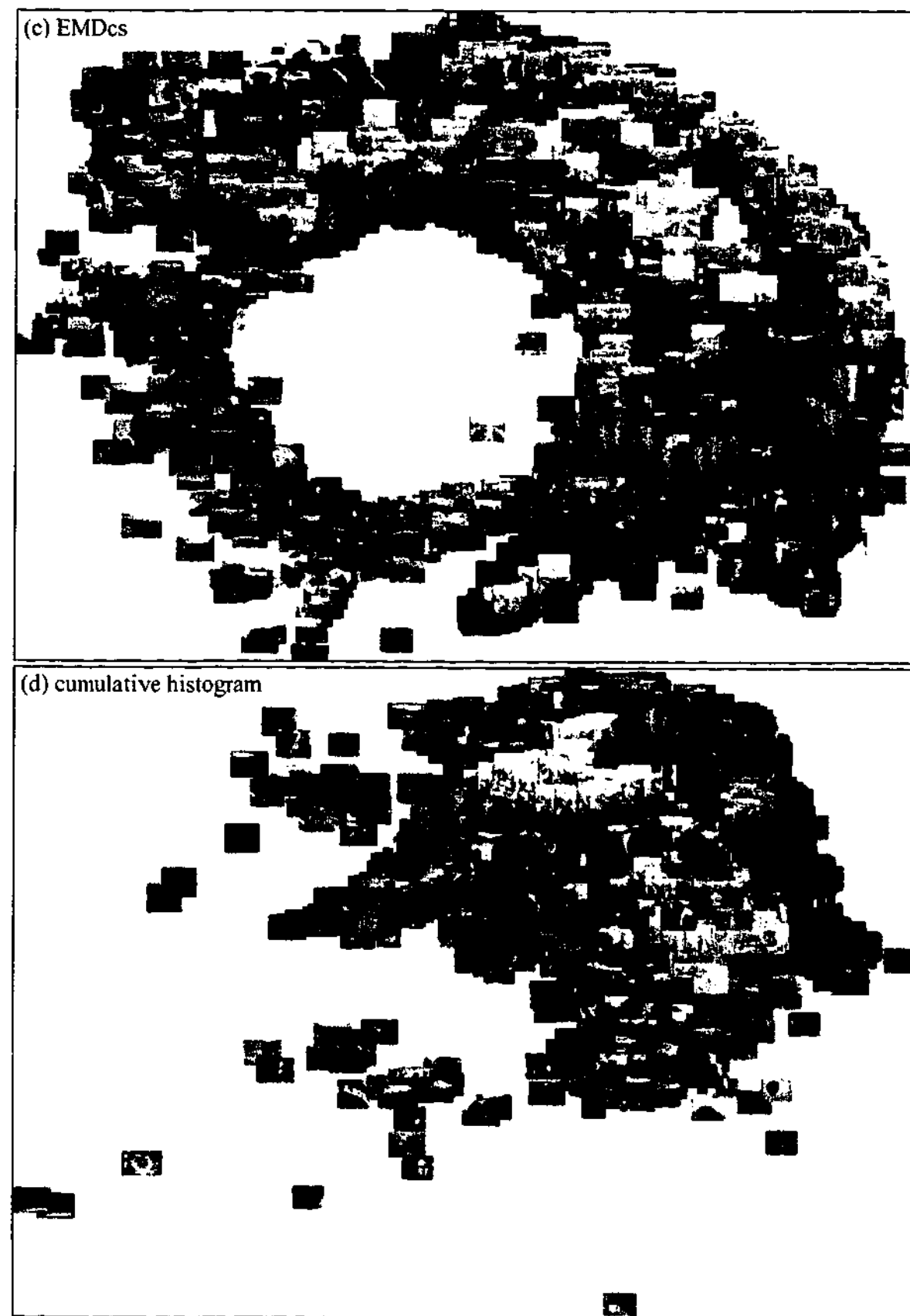


Figure 6.4: ... continued from previous page.

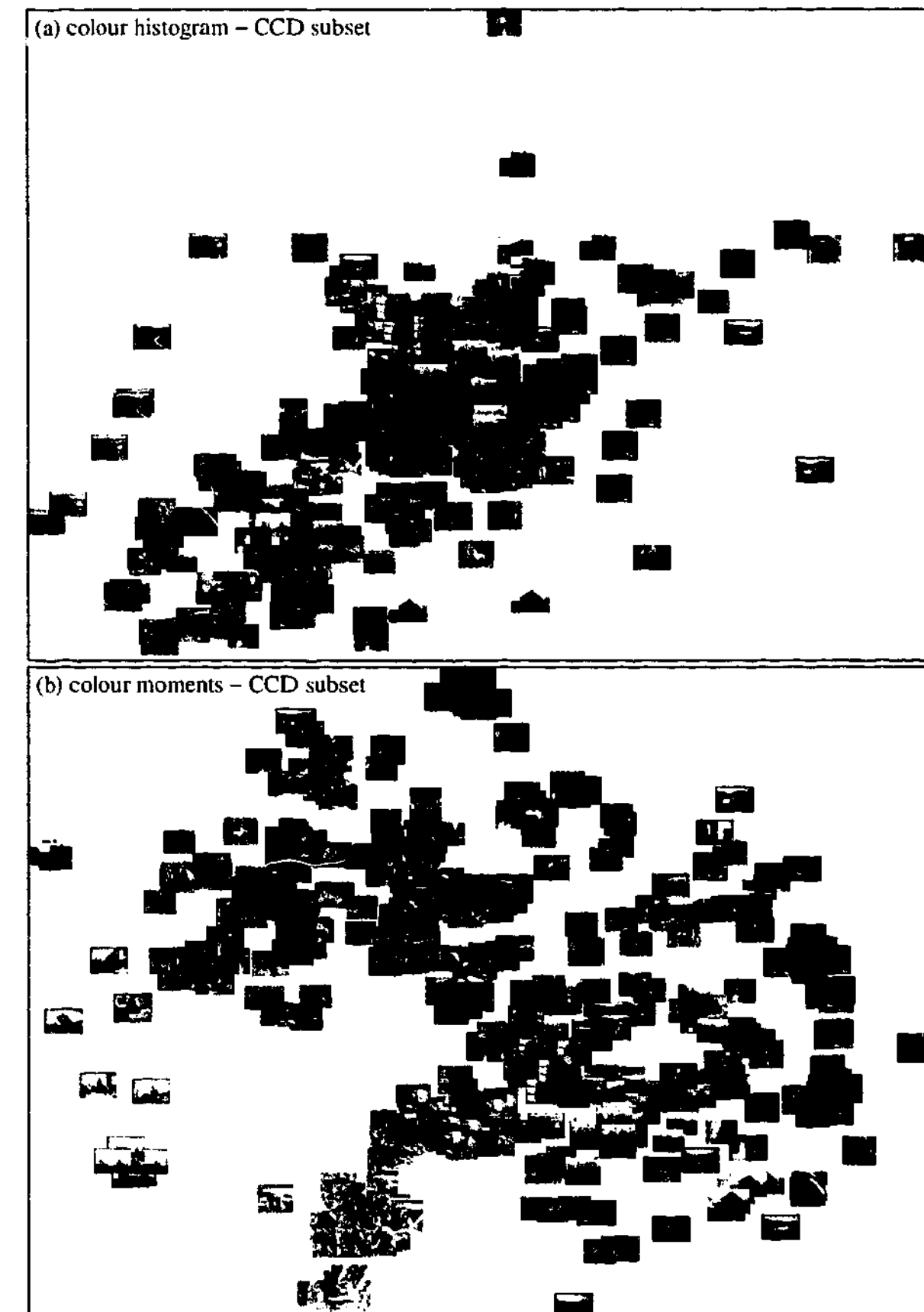


Figure 6.5: Generated layouts using a subset of CCD made up of the 50 CCQ and all the relevant images - 387 images. ... continued on next page.

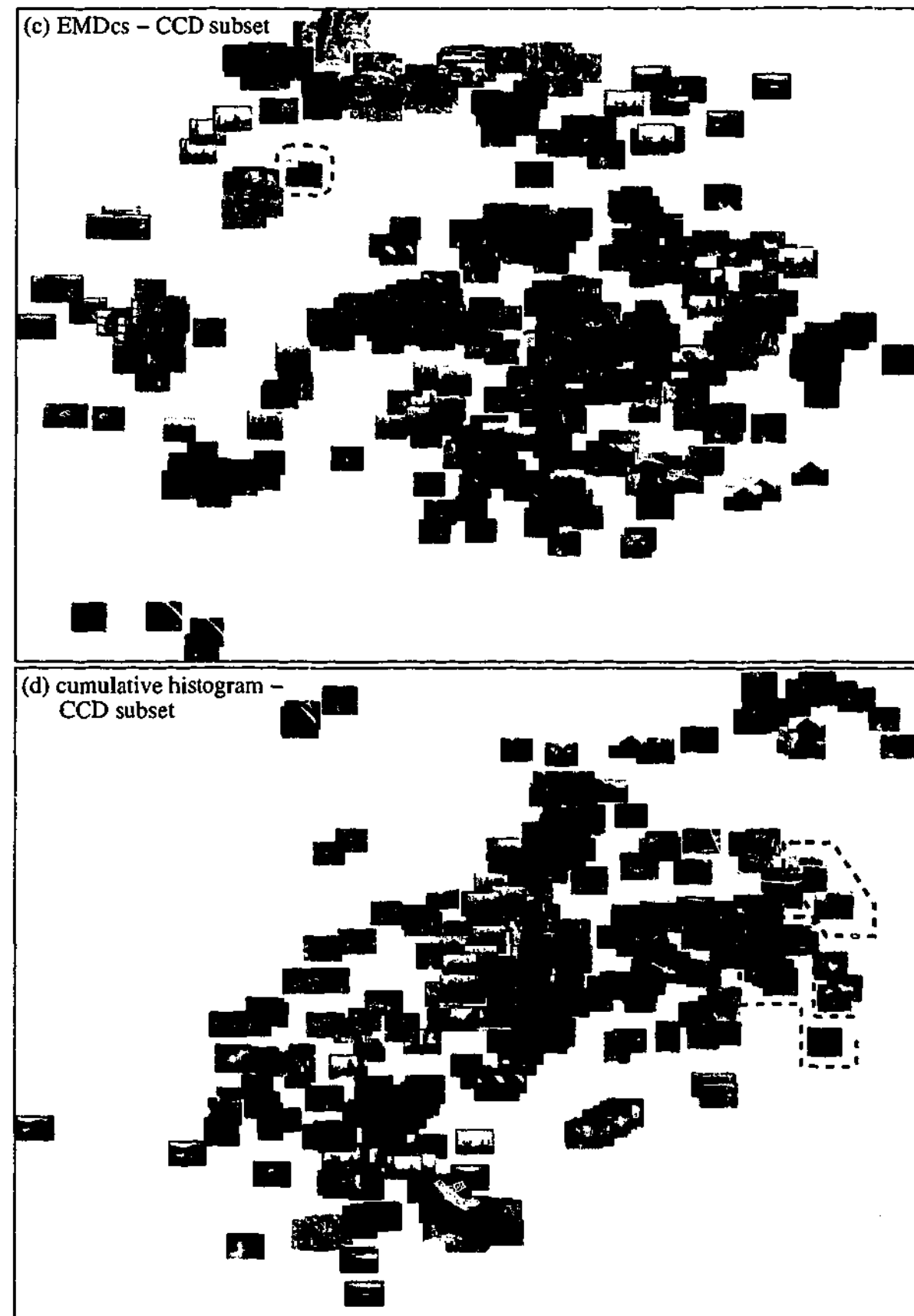


Figure 6.5: ...continued from previous page.

For colour histogram in Fig. 6.5(a) and colour moment in Fig. 6.5(b), the layouts still appear somewhat random. In colour histogram, predominantly blue, green and red images can be found in many sections of the display. In colour moment, predominantly blue, green and red images can be found in almost every quadrant of the layout.

For EMDcs (see Fig. 6.5(c)), the layout appears less random because predominantly blue, green and red images each are concentrated only in a section of the display. However, notice that the two *castle* images indicated in the red polygons are located quite far apart from each other. This is because it is often impossible to represent the distances faithfully in the low dimension, therefore some images appear to have been misplaced (see Section 5.3 for more details). However, the number of misplaced images is so few that EMDcs is, overall, still contextually meaningful.

For cumulative histogram (see Fig. 6.5(d)), the findings for EMDcs are also applicable: the display appears less random and some similar images are located quite far apart. It can be seen that some predominantly red images indicated by the red polygons are located in two parts of the display but it is unfair to conclude that it is more random than EMDcs because the *castle* images indicated by the green polygon which were misplaced in the EMDcs are correctly placed here. In short, like EMDcs, the number of misplaced images are so few that it is, overall, still contextually meaningful.

Cumulative histogram, however, has an advantage over EMDcs because it is more efficient than EMDcs: it is simpler and faster to generate and evaluate. On an Intel P4, 1.4GHz PC running Linux, extracting a cumulative feature vectors takes 130ms, and calculating the distance between two feature vectors needs a negligible 0.004ms. Extracting features from the same image in EMDcs requires 620ms, and calculating the distance between two feature vectors takes between 0.5ms to 6ms. This means that the use of cumulative histogram will speed up the time taken to find a layout.

6.2.4 Further Discussions on Contextual Meaningfulness

In this section, further analysis on each colour feature and the *Stress* function reveals why some colour features are more contextually meaningful or less contextually mean-

ingful than others. Colour moment is less contextually meaningful because extracting colour features in each colour channel independently compromises the accuracy of the colour description (see Section 2.1.1.3). The feature vectors, therefore, may not reflect perceptual similarity of colours when there is more than one colour in an image.

From Section 5.3, we know that the MDS algorithms find a layout by optimising the *Stress* function (5.1). *Stress* measures how well the layout in the low dimension represents the distances between objects in the high dimension, therefore smaller values are more desirable. Colour histogram is less contextually meaningful because it only captures the similarities and dissimilarities of corresponding bins, not of different bins. It was established earlier that contextual meaningfulness depends on how dissimilar images are organised in the layout, so an MDS algorithm has to know how big the differences are between two dissimilar images so that these differences can be reflected in (5.1), the function it tries to optimise. For this reason, it is also important to know the differences between different bins, otherwise the MDS algorithm fails to correctly arrange the visually dissimilar images. Here is a very simple example to illustrate this problem.

If we give the colour histogram feature vectors of four different single-coloured images to MDS to find a layout, then, the MDS optimises the value of *Stress*, and it will eventually converge to a layout with the lowest value of *Stress*. This, however, does not imply this layout is contextually meaningful. Figure 6.6 on the following page shows two layouts with four single colour images: (a) red, (b) dark red, (c) green and (d) dark green. The colour in each image belongs to different bins, so the distance between any two images is at its maximum i.e. 2. These original distances are reflected well in the generated layout but not in the preferred layout; however, the generated layout is less contextually meaningful compared to the preferred layout. In the preferred layout, dark red is closer to red than it is to green or dark green: the arrangement is more contextually meaningful than the generated layout.

The EMDcs feature, unlike histogram, can capture information from different colours. For this reason, its layout is more contextually meaningful. For cumulative histogram,

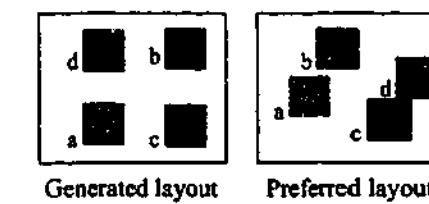


Figure 6.6: The problem of using colour histogram feature for proximity visualisation. The preferred layout is more contextually meaningful than the generated layout.

although it is very similar to histogram, the layout generated using cumulative histogram is less random than colour histogram. This occurs because cumulative histogram considers the perceptual similarities of colours as explained in the following simple example. Figure 6.7 shows the feature vectors of three single colour images; in cumulative histogram, the distance between the red and magenta images is 1 and the distance between the red and cyan images is 3. In colour histogram, the distances between all colour images would be 2. The layout generated using the cumulative histogram feature is, therefore, less random than the layout generated using colour histogram.

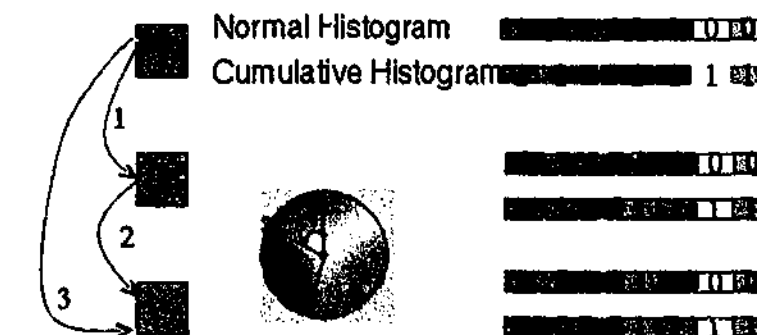


Figure 6.7: In cumulative histogram, the distances between the feature vectors reflect the perceptual similarity of colours; the distance between red and magenta images is one while the distance between red and cyan images is three. In histogram, the distances between all images are two.

To test the validity of the hypothesis that it is important to capture dissimilarities from different colours, artificial colour images were created and three layouts were generated based on histogram, EMDcs and cumulative histogram. Each colour image contains only one colour so that it is easier to compare the randomness of the layouts. The hue (H) was varied from 0 to 360 in increments of 5 degrees, the saturation (S) and brightness (darkness or V) were each given values of 0.3, 0.6 and 1 giving a total of 657 images. Three layouts were then generated using each colour feature and the

results are given in Fig. 6.8 on the following page.

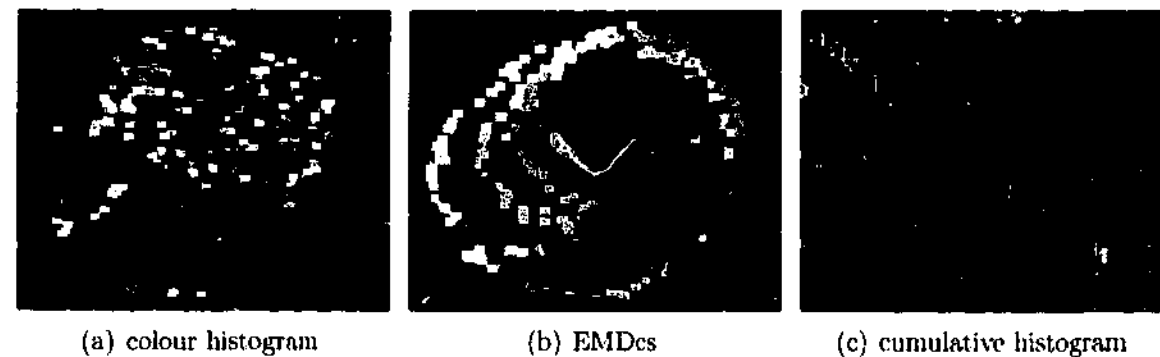


Figure 6.8: Proximity visualisation using artificial images. The layout generated using colour histogram is random, while the other two layouts are contextually meaningful.

It is clear that the layout generated using histogram appears random while others appear less random. The artificial images in the layout generated using EMDs are organised into several sets of circles, and each circle is made up of images of different hues having the same saturation and brightness. The layout from cumulative histogram appears to suffer from two problems. First, it has only one dimension and second, the most perceptually similar colours are at both ends of the line: the cumulative histogram fails to capture the circular nature of hue. Because of these two problems, initially, it appears that cumulative histogram is unsuitable for visualisation; however, these problems are unnoticeable when used for real colour images as shown in previous sections. In this display, it has only one dimension because there is only one colour in each image. General colour images use more than one colour, and as a result, the layouts for general colour images are comparable to those generated using EMDs.

6.3 Implementations of Colour-Based eyeMap

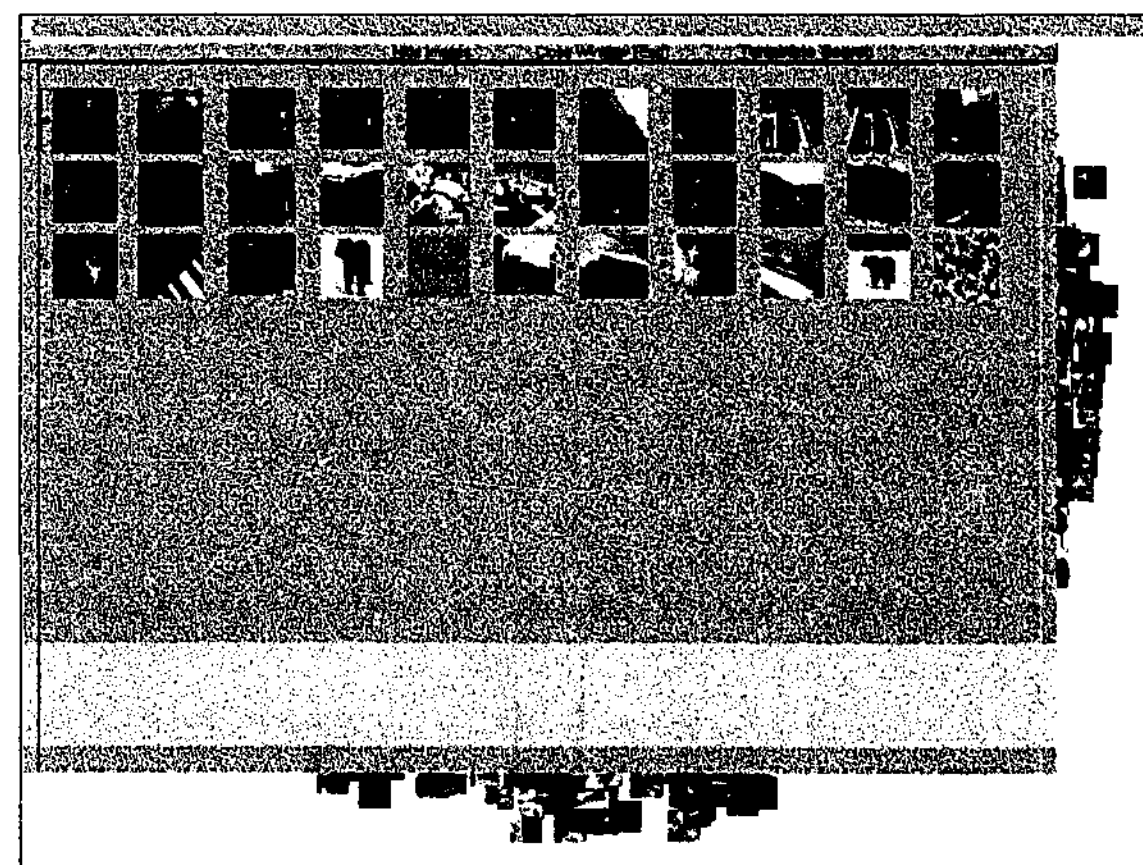
Having established a suitable feature for browsing colour images, it is now possible to implement colour-based eyeMap. eyeMap specification in Chapter 5 requires the images to be clustered first before generating the layout. The question is which feature should be used for clustering? In Section 6.2, we demonstrated that the more effective feature vector for retrieval may not be the most suitable one for visualisation, but in

clustering, we should use such a feature vector since it is more likely for the relevant images to be in the same cluster. To solve these two conflicting requirements, we use two feature vectors: I-auto for clustering and cumulative histogram for visualisation.

Colour-based eyeMap was fully implemented on a GNU/Linux Debian using C++ and FLTK graphical user interface [133]. Also, as recommended in the previous chapter, the image format used in colour-based eyeMap is JPEG 2000. Screen shots of colour-based eyeMap can be seen in Fig. 6.9 on the following page. The arrangement of images is systematic because they are grouped by colour similarity. The context region displays only the representative images in smaller size, whereas the focal region (within the ellipse) displays all images in bigger size. Users can also display all images within the focal region linearly without any overlapping in a separate window as demonstrated in Fig. 6.9(b). Colour-based eyeMap is also integrated with a retrieval engine developed using I-auto, which was described in Chapter 4.



(a) The initial screen



(b) With linear display

Figure 6.9: Screen shots of colour-based eyeMap. Images in the focal region, within the red ellipse (a), can be displayed linearly in a separate window (b).

6.4 Conclusions

It is important to evaluate the suitability of colour features for visualisation because random display of images does not facilitate browsing. In this chapter, we established which colour feature is more suitable for browsing colour image databases by evaluating four colour features for visualising colour image databases: colour histogram, colour moments, EMDs and cumulative histogram. Experimental results showed that cumulative histogram is most appropriate for visualisation because the location of relevant images are close to each other and the layout is contextually meaningful. In addition, the distance calculation between any two cumulative histograms is efficient, so using this feature will speed up the process of finding a suitable layout for browsing.

This research also showed that the feature most suitable for retrieval may not be suitable for visualisation. This finding has three implications. First, I-auto, although found to be more effective for retrieval, is unsuitable for visualisation because it is based on the colour histogram technique. Second, in the implementation of colour-based eyeMap, I-auto was used for clustering the images whilst cumulative histogram was used for generating the layout. The final implication is the retrieval engine integrated with colour-based eyeMap should be built using I-auto because it has the highest retrieval effectiveness.

Colour-based eyeMap has been implemented and is a fully functional system. Initial experience, as described in the previous chapter, shows that eyeMap is promising as a tool for browsing and solving the Page 0 problem. Its usefulness for these two tasks was confirmed by evaluating an implementation of eyeMap against existing systems. The next chapter discusses this evaluation of colour-based eyeMap against existing systems.

Usability Study of Colour-Based eyeMap

Research in visualisation has largely centred on the development of new techniques. It was only recently that the evaluation of the techniques was considered important [17]. The usability study undertaken in this research pioneers the evaluation of visualisation for image browsing. Rodden and Combs et al. [26, 115] did not, strictly speaking, evaluate image visualisation systems because visualisation implies displaying large amounts of data and in relation to image databases, large numbers of images. They evaluated systems which display only one hundred or at most two hundred fifty images, while eyeMap displays thousands of images. Thus, the main contribution of this chapter is the testing of large image databases which has never been done before.

This chapter describes the evaluation of colour-based eyeMap and existing systems for browsing and solving the Page 0 problem. The purpose of the study is to show if eyeMap is better at solving the problems it is designed for, by comparing the performance of colour-based eyeMap and existing systems in a usability study. Recall that eyeMap is a browsing concept, so only the implementations of eyeMap (such as colour-based eyeMap) can be tested and the success of the implementation indicates the success of eyeMap. As mentioned in the previous chapter, the philosophy behind the design of eyeMap as a concept for image browsing is to enable users to transfer their daily browsing activity into image browsing; therefore, if eyeMap is better, then users have successfully transferred that experience into image browsing. In addition, the study provides insights into how humans search for images, and the findings are

useful for designers of any image browsing or search programs.

7.1 Evaluation Methods and Design

Figure 7.1 on the following page shows different types of usability evaluation methods and techniques [8]. Only some of these methods require users' participation, and they include direct and indirect field studies; observation, attitudinal, and experimental methods in a laboratory setting; and collection of log files and market performance for statistical analysis. Other methods are conducted without users' participations and they include cognitive walkthrough, heuristic, and predictive models. To evaluate eyeMap, we conducted a usability study on colour-based eyeMap using the attitudinal and experimental methods (printed in blue).

For the attitudinal method (see Fig. 7.1), we collected users' attitude towards the programs whilst in the experimental method, we collected their performance in using the programs. The instructions for the evaluations were carefully controlled to ensure that all volunteers received the same description by reading the instructions from a written manuscript. The next sections describe the evaluation criteria, the systems tested (existing ones and colour-based eyeMap) and the experimental design.

7.1.1 Evaluation Criteria

To evaluate if a program is useful for solving the Page 0 problem, we asked users to search for a target image using the program. A program is considered useful for solving the Page 0 problem if users:

- can find the target image and;
- find it quickly;
- rate the program highly; and
- like to use the program.

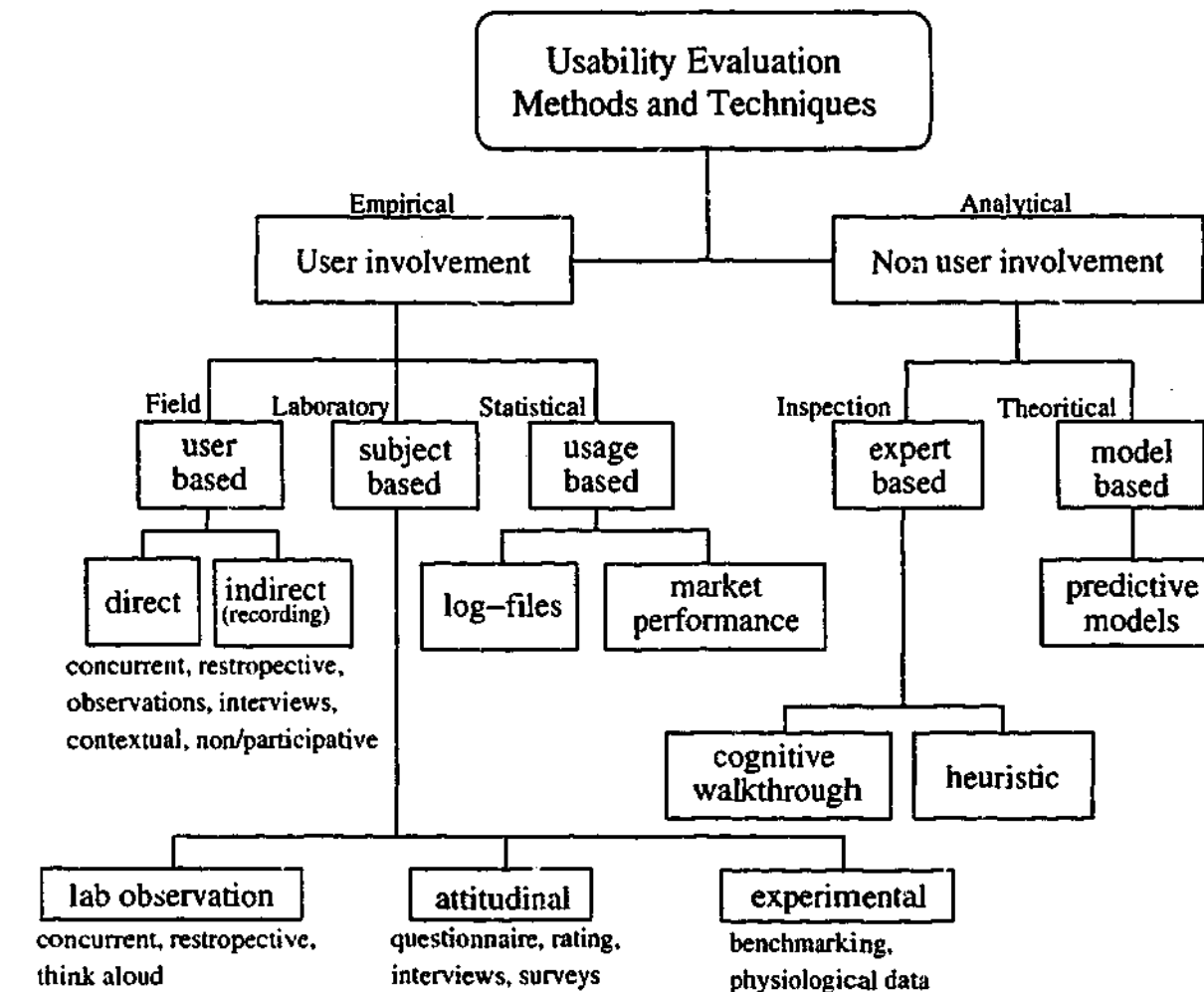


Figure 7.1: Different types of evaluation methods and techniques. Techniques and methods printed in blue were used for evaluating eyeMap.

The first two criteria measure participants' performance in using a program, while the last two criteria measure participants' perception of the program. The data for the last two criteria was collected from a post experiment questionnaire, and a sample of the questionnaire can be found in Appendix D. More discussion on the questionnaire will be given in Section 7.1.3.1 when we cover the experimental procedures. The above four evaluation criteria are also appropriate for measuring the usefulness of eyeMap for browsing, so the same criteria will be used for that purpose as well.

To properly evaluate eyeMap, four programs were developed: two were based on eyeMap and two were based on the traditional linear display. One eyeMap program did not have visual query facility but the other one did. To use the visual query facility, users performed a query-by-example by submitting a sample image to retrieve other

images; this type of query is more commonly known as a visual query. The other two programs developed for this study were based on the traditional linear display. One program in this display did not have visual query whilst the other one did. The taxonomy of these four programs are given in Fig. 7.2. The programs using linear layout, that is P1 and P2, are the existing methods for browsing and for finding a sample image to initiate a visual query. The only difference between P1 and P2 is P2 has visual query. The other two programs, P3 and P4, are based on eyeMap, and the difference between these two programs is P4 has visual query. These programs are the combinations of two factors: layout (eyeMap or linear) and existence of visual query (with or without).

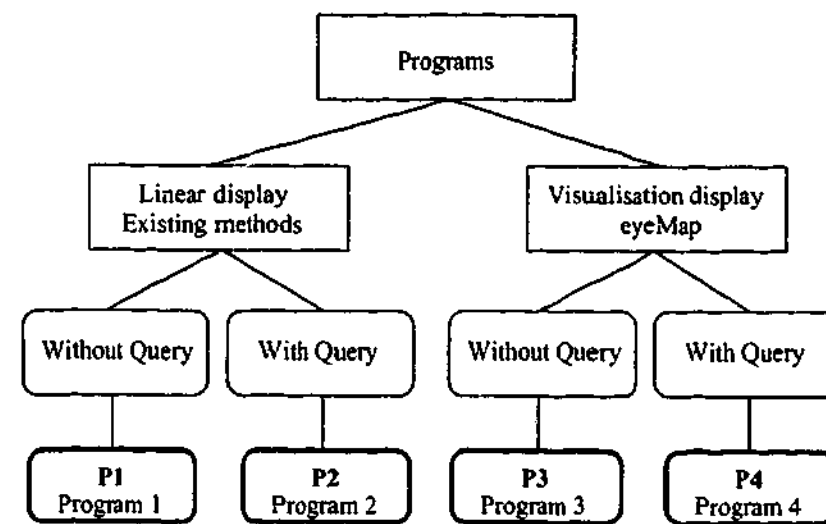


Figure 7.2: Four programs developed to carry out the usability study. Two programs use linear display (P1 and P2) and two programs use eyeMap (P3 and P4).

To resolve which systems are better for either of the two tasks, we compared which factor is more useful in helping users in completing the tasks, and performed a statistical test to establish if the differences were statistically significant. It means that for the Page 0 problem, if P3 and P4 are found to be more useful, and if only the layout factor is significant, then P3 is better than P1, and P4 is better than P2. This means eyeMap with or without visual query is the best system for solving the Page 0 problem. If both factors are significant, then only P4, eyeMap with visual query, is the best system for solving the Page 0 problem.

The same principles were also used for browsing, that is if P3 and P4 are found to be more useful and if only the layout factor is significant, eyeMap with or without visual query is the best system for browsing. If both factors are significant, then only P4, eyeMap with visual query, is useful for browsing.

The description of the evaluation criteria is now complete. The next sections describe the functionalities of each program.

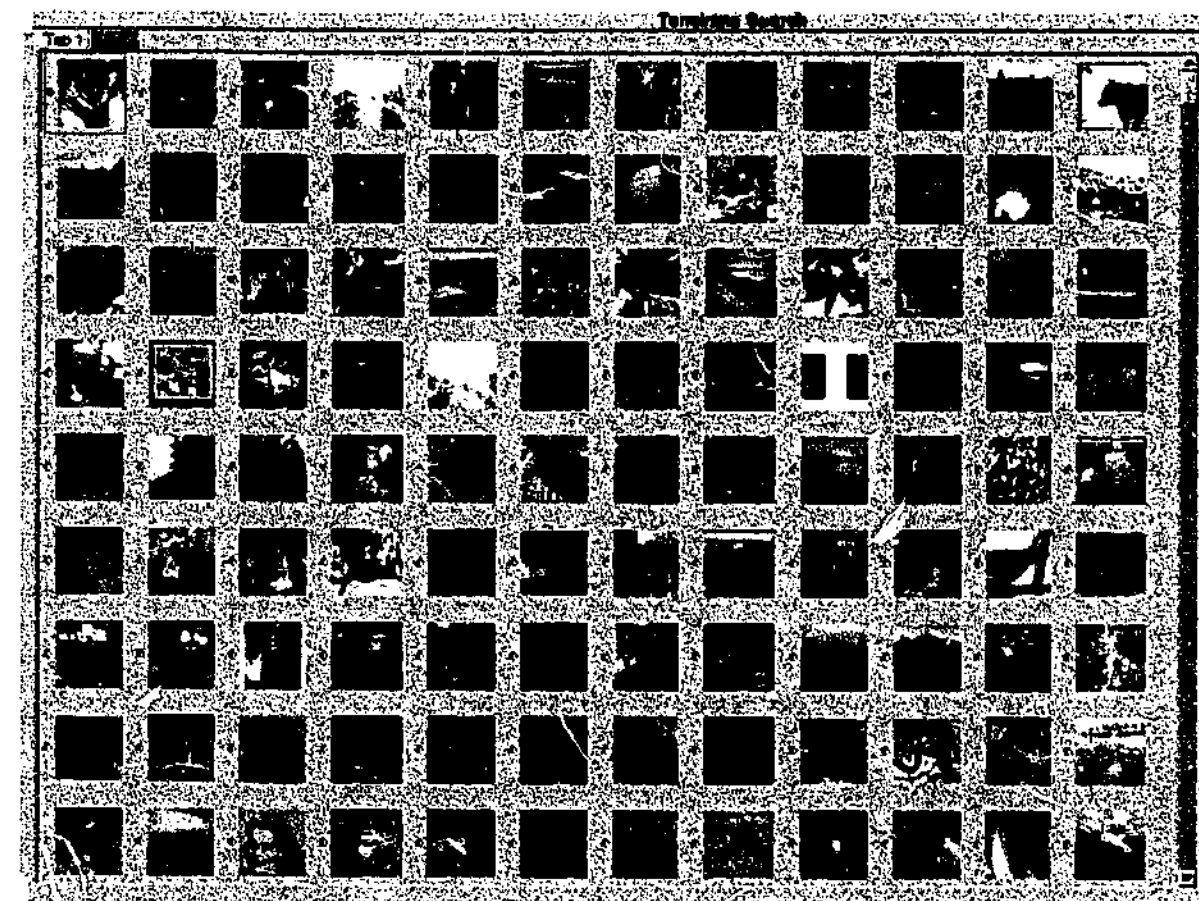
7.1.2 Descriptions of Existing Methods and Colour-Based eyeMaps

The Program - P1

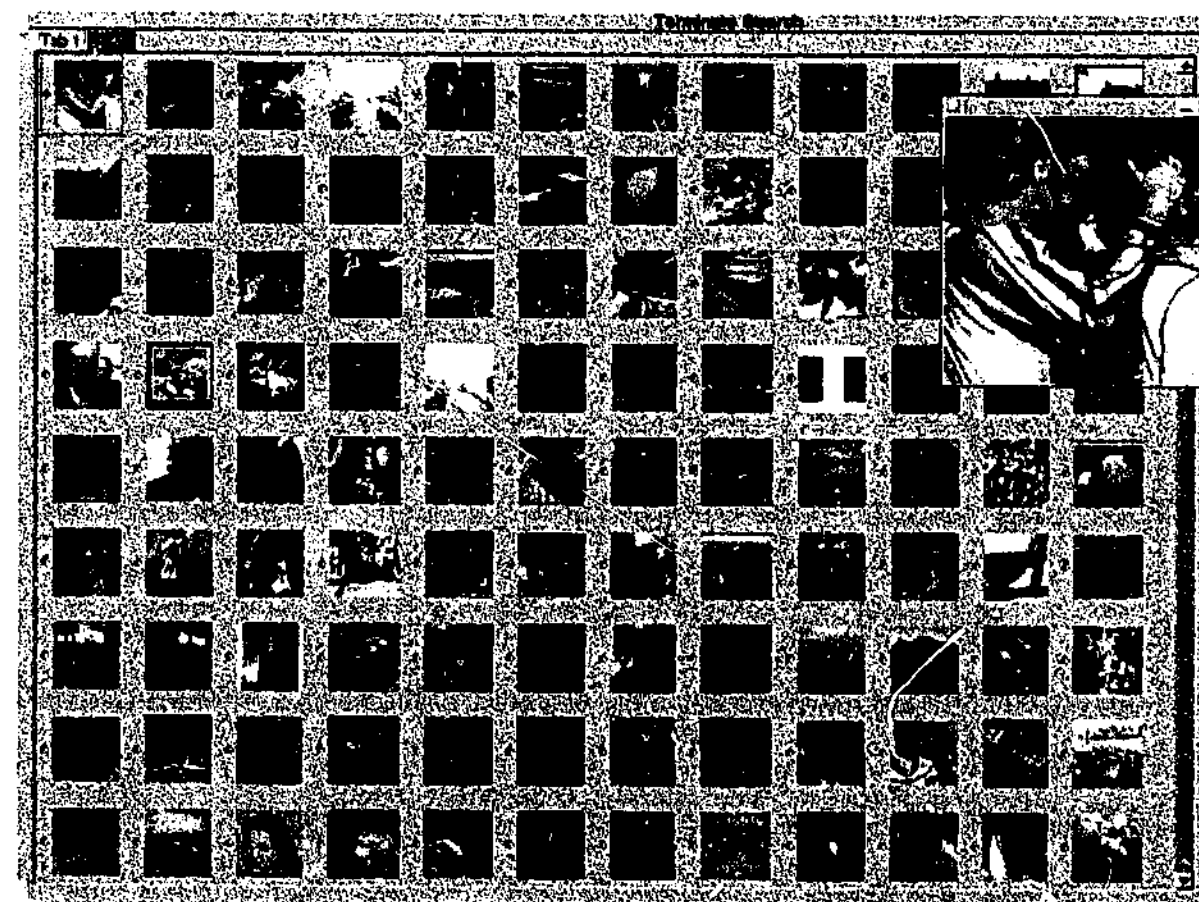
A screen shot of P1 is given in Fig. 7.3 on the following page. The width and height of the display covers the entire viewing area, and one screenful contains 108 thumbnail images of 64 by 64 pixels. The images are displayed linearly, one after the other, and their order in the display is random. To view more images, users simply use the scroll bar located on the display's right hand side; alternatively, they may use the scroll wheel commonly found in the more modern mouse input device. They can also view the images one at a time in full size by clicking on it and a separate window will display the image in full size as seen in Fig. 7.3(b).

The Program - P2

As mentioned earlier, P2 is P1 with an additional visual query function, so apart from the visual query, it is exactly the same as P1. To use the visual query, users choose a sample image by clicking on one of the thumbnails. Then, to activate the visual query, they select the visual query function using either a mouse to click on the pull-down menu or a keyboard stroke as the shortcut. Figure 7.4 on the following page shows screen shots of P2 before and after the visual query. Initially, the visual query function returns 40 most relevant images to the query image but at any time, users can modify the number of images to be displayed easily by using the pull down menu or a keyboard stroke.

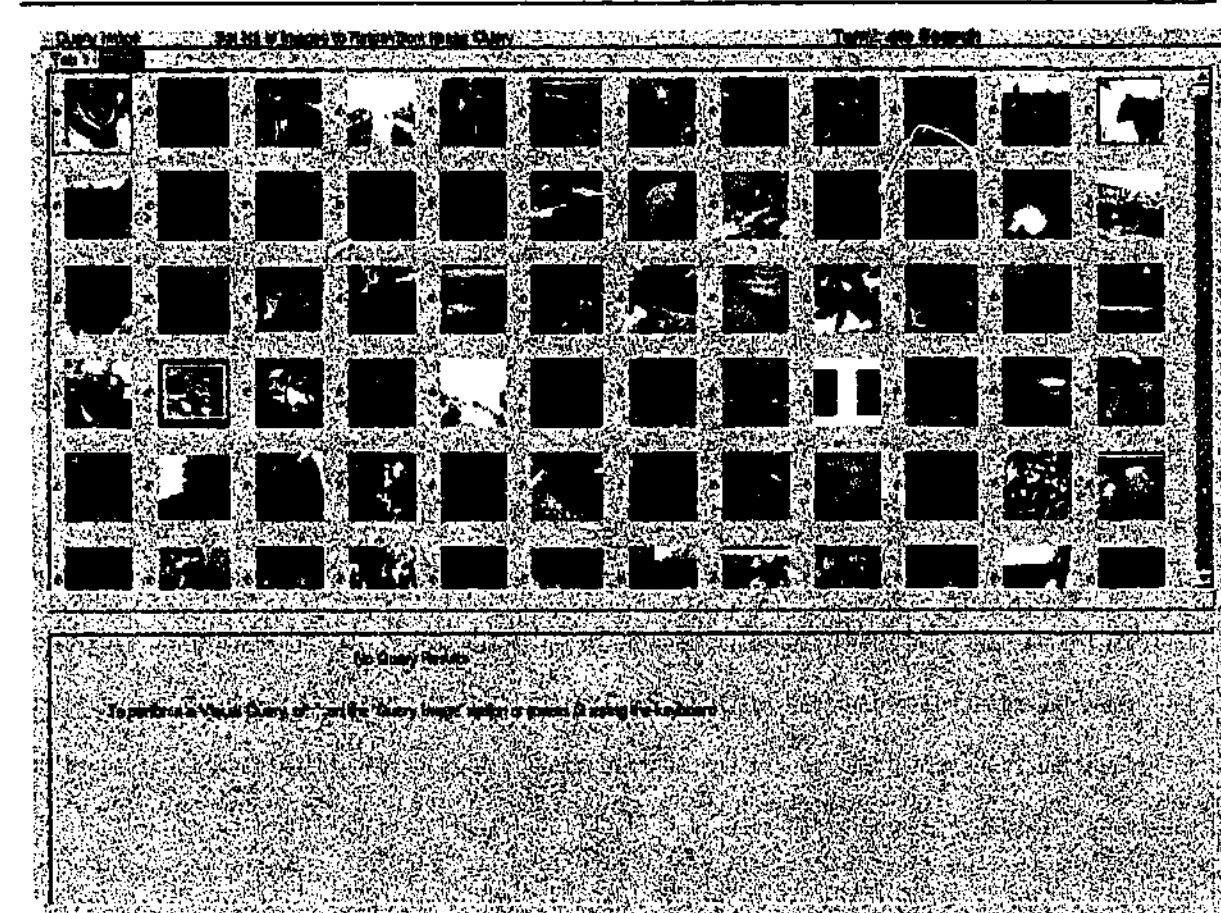


(a) Linear display of images

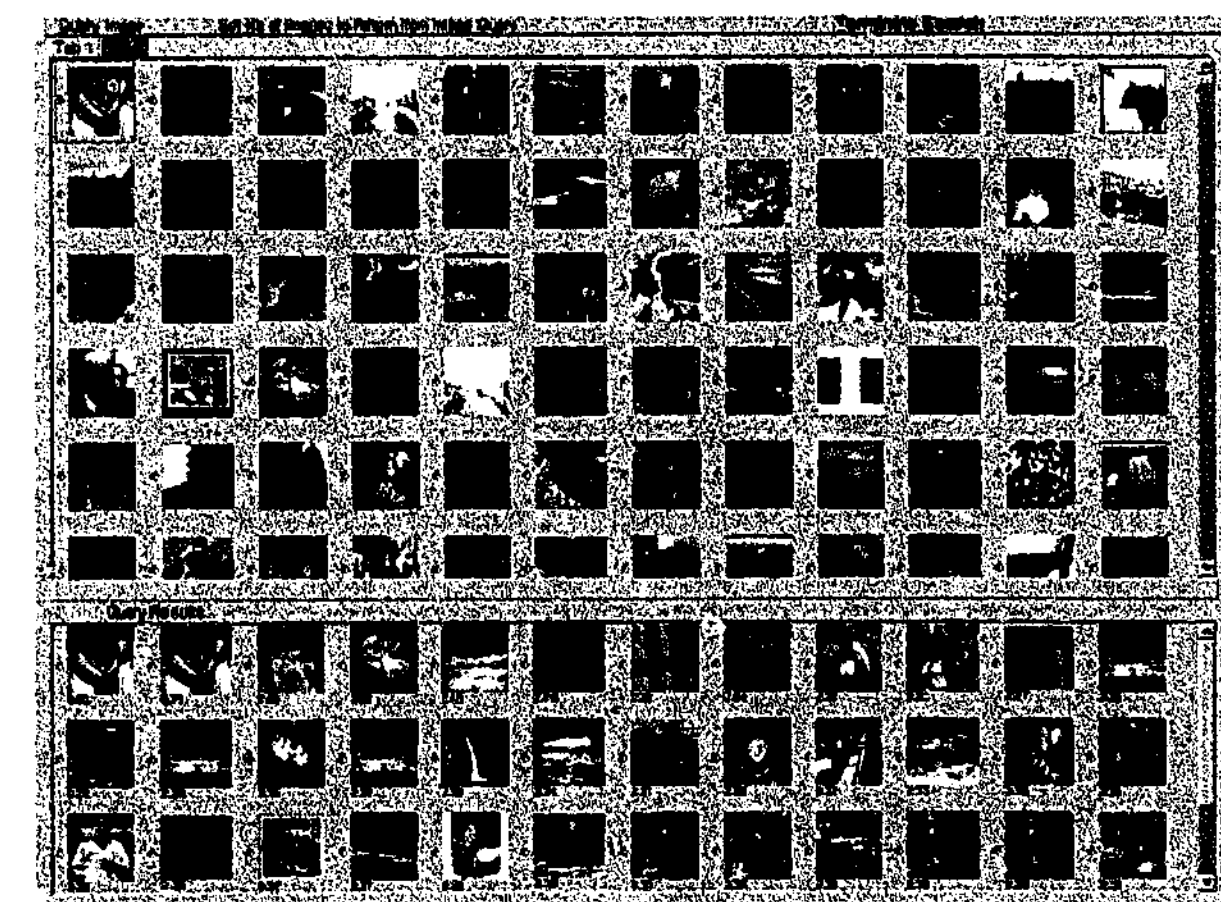


(b) One image displayed in full size

Figure 7.3: Snapshots of P1. The images are displayed one after the other, and the order of the images is random. Users scroll the display to see more images.



(a) Linear display before visual query



(b) After visual query

Figure 7.4: Snapshots of P2. It is exactly the same as P1 but it has a visual query function.

The Program - P3

The functions of this program were described in the previous chapter, and the screen shots for this program can be seen in Fig. 6.9 on page 126. Like P1 and P2, the display also covers the entire viewing area. In summary, the functions and their implementations are given as follow:

- Visualisation display with focal and context regions, and the images are grouped. For each group of images, a representative image is selected. In the context region, the size of the images is 32 by 32 pixels and only the representative images are displayed. In the focal region, the size of the images is 64 by 64 pixels, and the non-representative images are displayed as well. In this study, the focal region is initially located at the centre of the screen and none of the target images is within this focal region.
- Moving the focal point to the area of interest with the mouse either progressively by dragging, that is left or right press and move the mouse, or directly by clicking on the area of interest.
- Removing of image overlapping can be done by:
 - changing the distortion factor in either of two ways: firstly, by rolling the mouse-wheel, then a small window pops up for two seconds to display the current distortion factor; and secondly, by using the pull down menu.
 - displaying the images in the focal region in a separate window (see Fig. 6.9(b) on page 126). Users can also hide all images in this window so that they will not be shown in the visualisation display. To show these images in the display again, they click on an option in the menu bar. Similar to P1 and P2, in this window they can view the images in full size one at a time by clicking on the thumbnail.
- Showing hidden images. Show all images which have been hidden.

The Program - P4

P4 is an enhancement of P3, as it has all the functionalities in P3 as well as the visual query function. This function is accessible from a separate window which displays images in the focal region linearly. The screen shots for this program can be seen in Fig. 7.5 on the following page. This figure shows the linear window before visual query and after visual query. Like P2, the visual query function initially returns the 40 most relevant images to the query image but users have the freedom to change this number in the same way as in P2, that is, by either using the pulldown menu or a keyboard stroke.

Summary of Programs

P1 is the simplest as it has the least number of functions. In contrast, P4 is the most complex as it has the most number of functions and most unfamiliar user interface. Another way of looking at these programs is that a given program is the enhancement of another. The programs can be enhanced in two ways, in terms of the image layout or the visual query function. In other words, P3 is the visual enhancement of P1, while P4 of P2; likewise, P2 is the functionality enhancement of P1, while P4 of P3. These two enhancements constitute the following two factors: image layout (eyeMap or linear) and existence of visual query (with or without).

7.1.3 Experimental Design

The experimental design is Latin square or within-subjects, meaning that all participants use all four programs to search for the same *set* of images. The independent variables, variables fixed by experimenters, are target images and the sequence of the programs. The randomisation of these two variables are given in Table 7.1 on the following page in which each row is unique. Programs are coded from P1 to P4 and target images in letters, so P1p means use P1 (program 1) to practice, P1a means use P1 to search for image (a). The arrangement of images for all programs and target images were generated off-line to ensure that all participants who use the same program to

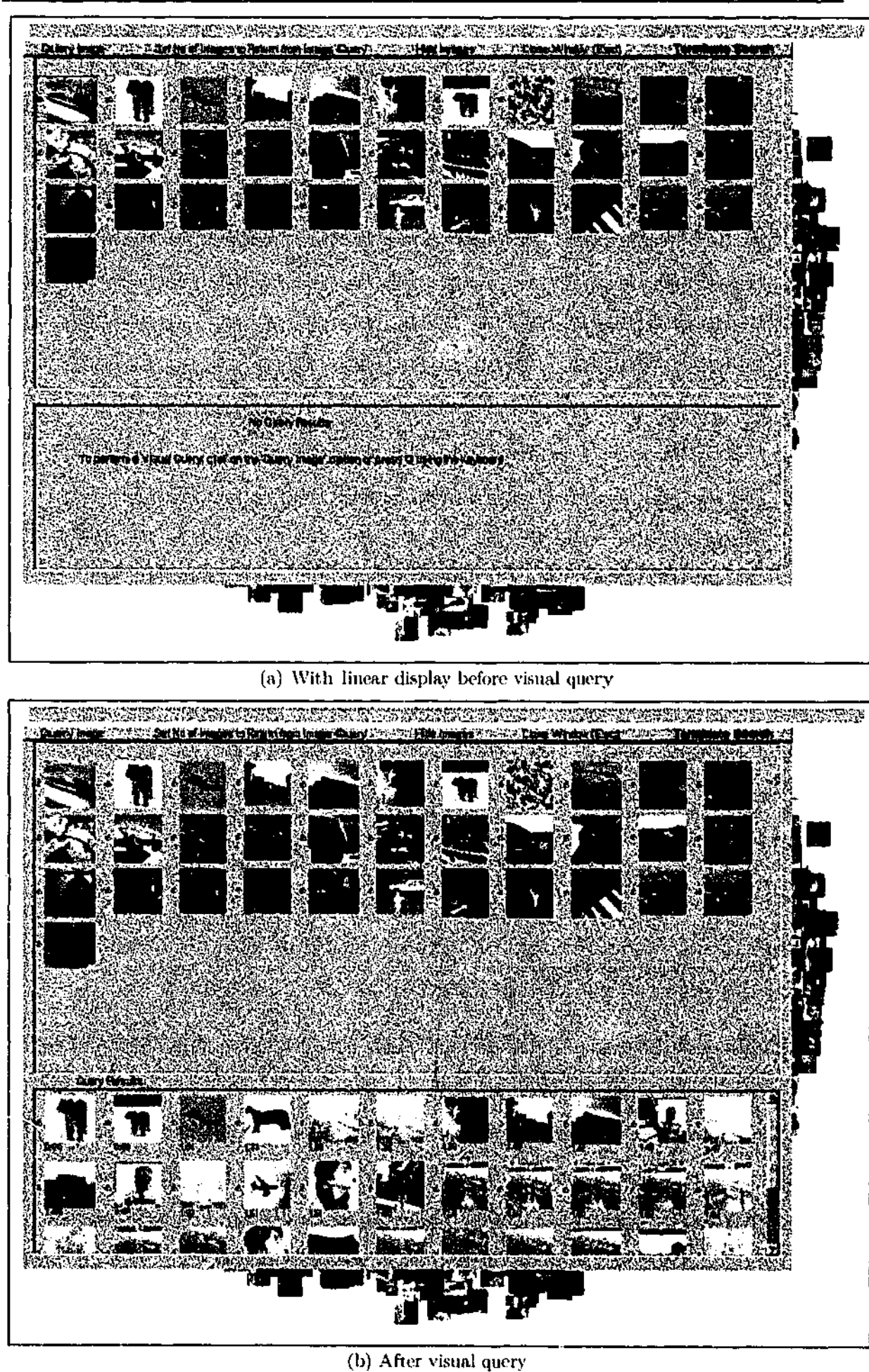


Figure 7.5: Snapshots of P4. The only difference between P3 and P4 is that P4 has a visual query function.

search for the same image see exactly the same image layout. All target images are given in Fig. 7.6: (p) is the target image for practising and (a) to (d) are the four target images for the tasks. The four target images (a) to (d) were chosen to ensure that they have different characteristics such as dark or bright colours, objects, animals or humans. To choose these four images, we first randomly selected n number of images from both CCD and PCD. Then, we chose the four images out of these randomly selected images to satisfy the criteria mentioned above.

Sequence of Programs and Target Images

Program and image sequence					Program and image sequence				
1	P1p P1a	P2p P2b	P3p P3c	P4p P4d	13	P3p P3c	P1p P1a	P2p P2b	P4p P4d
2	P1p P1b	P2p P2c	P4p P4a	P3p P3d	14	P3p P3d	P1p P1b	P4p P4a	P2p P2c
3	P1p P1c	P3p P3a	P2p P2d	P4p P4b	15	P3p P3a	P2p P2d	P1p P1c	P4p P4b
4	P1p P1d	P3p P3b	P4p P4c	P2p P2a	16	P3p P3b	P2p P2a	P4p P4c	P1p P1d
5	P1p P1a	P4p P4d	P2p P2b	P3p P3c	17	P3p P3c	P4p P4d	P1p P1a	P2p P2b
6	P1p P1b	P4p P4a	P3p P3d	P2p P2c	18	P3p P3d	P4p P4a	P2p P2c	P1p P1b
7	P2p P2d	P1p P1c	P3p P3a	P4p P4b	19	P4p P4b	P1p P1c	P2p P2d	P3p P3a
8	P2p P2a	P1p P1d	P4p P4c	P3p P3b	20	P4p P4c	P1p P1d	P3p P3b	P2p P2a
9	P2p P2b	P3p P3c	P1p P1a	P4p P4d	21	P4p P4d	P2p P2b	P1p P1a	P3p P3c
10	P2p P2c	P3p P3d	P4p P4a	P1p P1b	22	P4p P4a	P2p P2c	P3p P3d	P1p P1b
11	P2p P2d	P4p P4b	P1p P1c	P3p P3a	23	P4p P4b	P3p P3a	P1p P1c	P2p P2d
12	P2p P2a	P4p P4c	P3p P3b	P1p P1d	24	P4p P4c	P3p P3b	P2p P2a	P1p P1d

Table 7.1: Programs are coded in numbers and target images in letters: p - practice, a to d - the four target images (see Fig. 7.6). P1p means use P1 (program 1) to practice, P1a means use P1 to search for image (a), that is fruits and flowers.

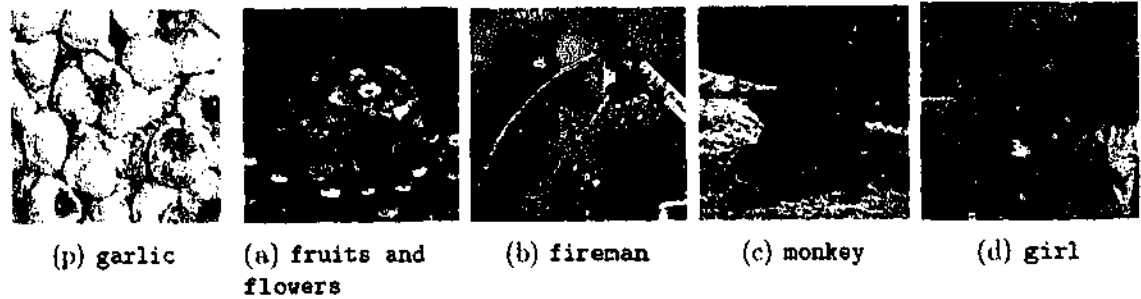


Figure 7.6: The target image used in the experiments. (p) is only used in the practice trials and (a) to (d) are used in the experimental trials. The names of the images are to aid discussions only, participants were unaware of them during the experiment.

7.1.3.1 Procedure

The experiment was conducted using a 17 inch CRT monitor. The task in the experiment was to use the four programs to search for four target images after viewing the images for ten seconds. By showing the target image for ten seconds, we hoped to simulate long term image memory [94, 136, 137]. The volunteers in this study used each program once to search for a different target image. The reason for conducting only one search in each program was to keep the total experimental time to about an hour, because searching for two target images could take about two and a half hours. This would not only discourage participation but would also affect the results due to fatigue. The sample size was increased by using a better alternative, that is by involving more volunteers. We decided that 24 participants were sufficient as each image would have been searched for at least six times using each program. It was also adequate to ensure that any variability between the images was averaged out. Note that 24 participants is a large sample size compared to most usability studies [96, 111, 115, 116, 140]. Another aspect that requires consideration is the manner in which the instruction is given to participants.

Participants in software usability studies often feel that their abilities are being judged and are consequently nervous [95]. To reduce this side effect in this study, we decided to allow users to perform the task as long as they wanted to and to be able to give up at any time. In addition, they were also told that while they were performing the task, they were not being visually observed. To further increase their confidence, they were told that the purpose of the study was to evaluate the software programs, not their abilities in using the programs.

At the start of each search, the target image was at first simultaneously displayed for ten seconds in full size (256 by 256 pixels) and in thumbnail (64 by 64 pixels) because participants, in general, prefer to look at target images in full sizes and in thumbnails before searching [113]. The data collected included:

- the number of successful searches;

- the time taken to find the target image or the time taken before they gave up in case they did not find the target image; and
- all functions' usage, including the time at which they were invoked.

All participants received the same instruction, tried each function and had equal opportunity to practise. Nevertheless, there were some differences in the search techniques or strategies because they were told to use the programs in any way they liked, that there was no right or wrong way.

After finishing all trials, they completed a questionnaire and had the opportunity to give further clarification verbally. A sample of the questionnaire can be found in Appendix D. The answers to the questionnaire and comments are both quantitative and qualitative. The quantitative data is useful for evaluating users' perception of the programs and the qualitative data for explaining the quantitative results. The questionnaire includes questions such as how they performed the search, program ratings using a five point Likert scale ranging from "strongly agree" to "strongly disagree", differences in images and, lastly, the preferred searching method *if there was any*. The last question was carefully phrased so that no one felt there had to be a preferred method.

To evaluate the usefulness of eyeMap for browsing and solving the Page 0 problem, the participants only needed to complete one task, that is, to find the target image. The evaluation for browsing differed only in the definition of successful search and the time taken to complete the successful search. For the Page 0 problem, successful search meant being able to find the target image, and the search time was the time taken to find the image. For browsing, on the other hand, successful search meant being able to find an image *similar* to the target, and the search time was the time taken to find the *similar* image. The programs logged all functions and the time when each function was invoked, so after the experiment, we checked which images they had looked at to determine if they had found a similar image and when.

The description of the experimental design and procedures is now complete. The

next section describes how to obtain the sets of images for performing the experiments.

7.1.3.2 Image Sets

The results from this experiment are meaningful only if the number of images is large enough so that a manual search is tedious; however, there is no guideline on how many is sufficient due to the lack of research in this area. Ideally, one would use a huge image database of at least half a million images, but this is impractical because a large image collection is not readily accessible to the majority of CBIR researchers. Collecting images from the Internet is infeasible, for they are either copyrighted or too small to be useful.

The combined size of the two image databases introduced in Chapter 3 i.e. CCD and PCD, is about 16,000 images, and although bigger than most research in CBIR, it is still too low for this experiment for the following reason. To ensure that no participant is advantaged from being familiar with the images, it is essential that the image set is unique for all trials, including the practice trial. With five trials, one practice trial and four experimental trials, five image sets are required. This means that each image set will have only 3,200 images. As a tradeoff between uniqueness and image set size, the number of images for each set was increased to 5,000 using the following approach.

First, we reserved 7,500 images so that each set had 1,500 unique images. Then, the remaining images were distributed across all five sets until each set reached 5,000 images. This distribution scheme ensures that each image in the database is used at least once and no image appears in all sets. The number of images taken from each database, unique and repeated, is proportional to the size of the database; for example, PCD is about twice the size of CCD, and therefore, the number of unique images drawn from PCD is about twice the number drawn from CCD. Finally, after creating the image sets, the images in CCD require some preprocessing for the following reason.

The images in CCD are of different sizes, and they could be any of the following: 240×320 , 256×384 , 320×240 , 352×288 or 384×256 pixels. On the other hand, the size of all images in PCD are 256×256 pixels. To guarantee that users search behaviour

is independent of the image size, the images in CCD were cropped to the size of images in PCD as follows. It is fair to assume that the most important information is at the centre; hence, the edges were removed as illustrated in Fig. 7.7. The width of the horizontal edges to be cropped = $\max(\frac{X-256}{2}, 0)$ and the height of the vertical edges to be cropped = $\max(\frac{Y-256}{2}, 0)$, where X is the width of the original image and Y is its height. The final image sizes are now either the same or very close to those of PCD i.e. 256×256 , or for images which are only 240 pixels wide or high: 256×240 or 240×256 pixels - these images are slightly smaller but the differences are unimportant because they are visually undetectable.

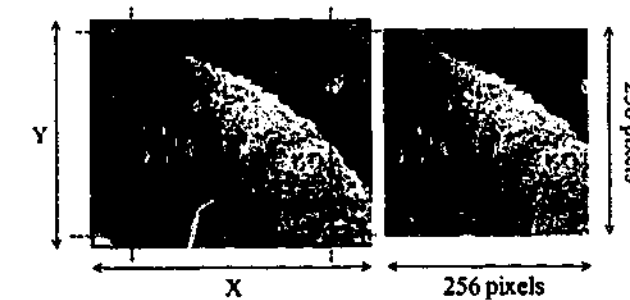


Figure 7.7: An example of image cropping in CCD.

7.1.3.3 Pilot Study

Prior to the experiment proper, we conducted a pilot study involving four computer literate students but ignorant as far as CBIR is concerned. The aims of the pilot study were to:

- test if the instructions were clear;
- test if the questionnaire was properly designed;
- determine the number of images to be used in the experiment because the results from the experiment are meaningful only if the task of looking for the target image manually using P1 is difficult; and
- uncover any unforeseen problems.

The results from the pilot study suggested that searching from 5,000 images is sufficient for the study. The average searching time using P1 is 11 minutes 53 seconds

compared to P4 at 6 minutes 15 seconds: using P1 is nearly twice as slow. The task using P1 was also rated at 4.5 (with 5 being the most unreasonable task) compared to P4 at 1.25. Also, two different participants gave up searching for the target image, one when using P1 and the other when using P2.

As a result of the pilot study, three changes were necessary. First, the instructions were reworded with jargon removed. Second, the questionnaire was modified slightly because in the pilot study, the participants were confused when asked if the task was realistic. We had to explain that "realistic" means if it is tedious, whether it is a reasonable task, something that they would like to do; therefore, realistic was changed to reasonable to better reflect the intended meaning. The final change was to allow participants to look at the target images whenever required because two participants had forgotten what the two targets looked like (87.5% recall rate). Research in human memory shows that human visual memory is unlimited but it is not 100% [94, 137]. This experiment requires a simulation of long term image memory, but not testing the memory. It is thus reasonable to allow them to view the target images again.

7.1.3.4 Participants

Twenty four volunteers participated in this study. All of them were undergraduate students or recent graduates from Monash University at the Gippsland Campus, with the following demography: 25% female, 75% male, 67% Computing and 33% non-Computing (Arts, Psychology, Business and Applied Sciences). The recruitment campaign was campus wide but the respondents were mainly computing students. Two of the participants have worked briefly on CBIR systems but were unaware of this project before they were recruited. All participants had normal or corrected to normal vision and normal colour vision (self reported). They were awarded with \$8.00 meal vouchers for participating in this study, and to avoid any bias, they were told we had developed all four programs.

7.2 Results and Discussion

The following sections present the results and analysis using the four criteria described in Section 7.1.1. The participants are labelled from 01 to 24, and any references made to their comments will be indicated using these labels.

7.2.1 Successful Search

As mentioned earlier, eyeMap could be used for solving the Page 0 problem and browsing. This section discusses the experimental results on the rate of successful search of all four programs: two programs based on traditional linear display (P1 and P2) and two programs based on eyeMap (P3 and P4). We will first discuss the results related to solving the Page 0 problem, then results related to browsing.

7.2.1.1 In Solving Page 0 Problem

Table 7.2 provides a summary of the numbers of both successful and unsuccessful searches for each program using each image - in total, the unsuccessful search rate was 24% ($\frac{\text{total no of fail}}{\text{total no of search}} = \frac{23}{96}$). The profile plots in Fig. 7.8 on the following page show the average successful search rate for each program - a higher successful search rate is more desirable. They are useful for visual comparison because we can clearly see which programs are better. It is obvious that having the visual query increases the chance of finding the target images, that is, P2 is better than P1, and P4 is better than P3. Similarly, having the visualisation layout also increases the chance of finding the target images, that is, P3 is better than P1, and P4 is better than P2. This means eyeMap is better than the traditional systems.

Numbers of Failed and Successful Searches

	P1		P2		P3		P4	
	fail	success	fail	success	fail	success	fail	success
fruits and flowers	1	5	2	4	1	5	0	6
fireman	2	4	0	6	5	1	2	4
monkey	1	5	0	6	1	5	0	6
girl	5	1	3	3	0	6	0	6
Total	9	15	5	19	7	17	2	22

Table 7.2: A search is unsuccessful if the participant failed to find the target image (in red) - the total failed or unsuccessful rate is 23.9%.

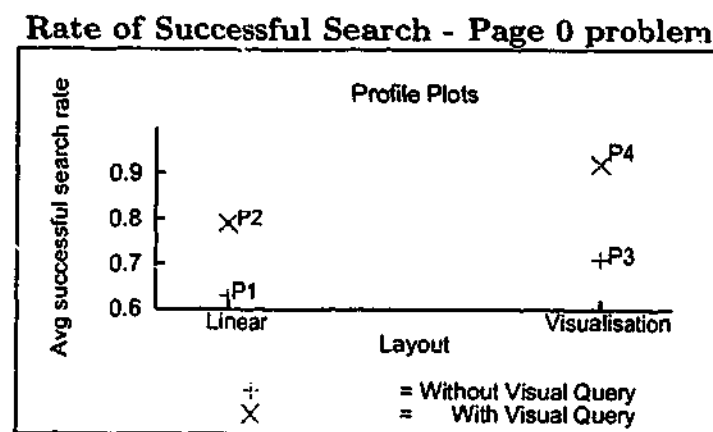


Figure 7.8: Profile plots for comparing the average successful search rate of P1, P2, P3 and P4 in searching for exact images, which is the task for testing the programs in solving the Page 0 problem.

The analysis is then followed by hypothesis testing using a two factor Analysis of Variance (ANOVA), as shown in Table 7.3, with the first factor being the type of image layout (linear or visualisation), and the second being the existence of query function (without or with). P1 and P2 are both displayed in linear layout but P2 has visual query. On the other hand, P3 and P4 are displayed in visualisation layout but P4 has visual query. The null hypothesis, H_0 , is that both factors do not significantly affect the successful search rate. The importance of using hypothesis testing is as follows.

Rate of Successful Search - Page 0 problem
Results of Hypothesis Testing

Factor	P
1 Type of Image Layout	*0.083
2 Existence of Visual Query	*0.057

* indicates significance at 0.1.

Table 7.3: Results of two factor ANOVA hypothesis testing.

In any study, two types of errors can occur: type I error for rejecting the H_0 when it is true and type II error for accepting the H_0 when it is false. The P -value indicates the probability of type I error occurring because "the P -value conveys much information about the strength of evidence against H_0 " [30, p309]. Rejecting an H_0 at P -value of 0.05, therefore, means that there is a 5% chance of type I error occurring. The accepted P -value is normally set at 0.05 to 0.1, depending on the seriousness of committing a type I error. The purpose of conducting a hypothesis test, in this case, is to evaluate if the

differences of the successful search rate are statistically significant. More specifically, it is to confirm that the differences do not happen by chance but are strongly backed by statistically significant evidence. The results of the test are given in Table 7.3: the P -values for the layout factor is 0.083 and for the visual query function is 0.057. This indicates that H_0 was rejected at 0.1 confidence level. This means P4, eyeMap with visual query, is significantly better than all the other programs.

7.2.1.2 In Browsing

The second type of application for eyeMap is in browsing, which will be analysed as follows. From Table 7.2, we observed that in P3, most participants gave up on fireman and only one each gave up on fruits and flowers and monkey. In the questionnaire, three out of five participants who gave up searching for fireman said that they could see at least one image similar to the target. The participants in this study normally enlarged the images to check if they had found the target; hence, we checked the function logs to see which images had been enlarged to find out which similar images they had found.

We found that all five participants who gave up on fireman found either visually or semantically similar images, some of which are given in Fig. 7.9. So, from that point of view, P3 did help them to find fireman. To be fair, we also checked the function logs in P1 and P4 - there is no need to check for P2, because all participants who had to find this image using P2 found the target image. For both P1 and P4, only one participant found a similar image.

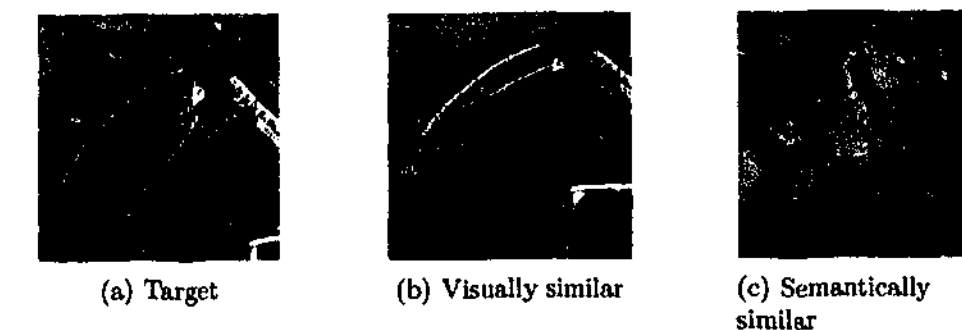


Figure 7.9: In P3, five participants gave up searching for image (a) but all five found other similar images like (b) or (c).

If the task was to be relaxed, that is users were only required to find a similar image instead of the exact one, then it becomes a browsing task, and as mentioned in the previous chapter, it is a more suitable task for P3. By redefining the task, the number of unsuccessful searches using P1 is now eight, P2 remains the same at five, P3 has two and P4 has one. The data was then reanalysed for this relaxed task, and the profile plots are given in Fig. 7.10 and the results of hypothesis test are given in Table 7.4. The P -value for the layout factor is now 0.002 and for the visual query factor, it is 0.328. Thus, the H_0 for the layout factor was rejected at 0.005 confidence level, but the H_0 for the visual query factor must be accepted.

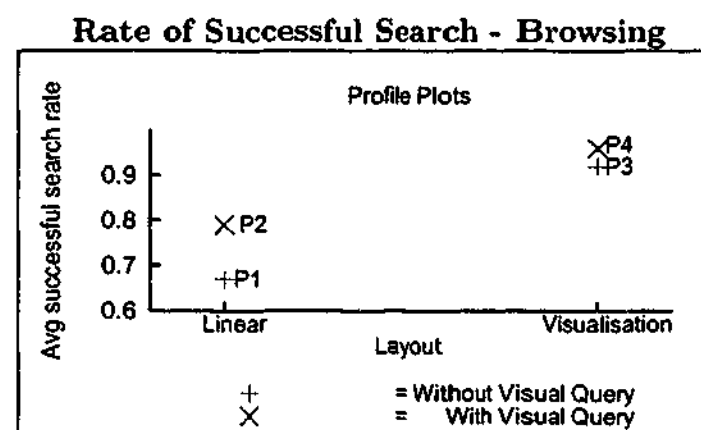


Figure 7.10: Profile plots for comparing the average successful search rate of P1, P2, P3 and P4 in searching for similar images, the task for testing the usefulness of programs for browsing.

Rate of Successful Search - Browsing
Results of Hypothesis Testing

	Factor	P
1	Type of Image Layout	***0.002
2	Existence of Visual Query	0.328

*** indicates significance at 0.01.

Table 7.4: Results of two factor ANOVA hypothesis testing.

It is now clear that for browsing, arranging images in the visualisation layout is more useful than using the visual query; however, when looking for the exact image the visual query function is essential. Searching for the exact image in P3, which has no visual query function, is difficult because when users found images similar to the targets in P3, they expected that the target images would be around the images they had found. When they failed to see the target images at the expected location or

when the target images were not at the expected location, they were disappointed, therefore, were more likely to give up. We mentioned in the previous chapter that it is not always possible to place all images at the correct location. For P4, they could issue visual queries using the sample images found, so they did not have to rely on the visual arrangement alone. Consequently, P4 is the best for either browsing or finding the exact image. From this analysis, we conclude that (1) eyeMap with visual query increases the chance of finding the exact image and (2) eyeMap with or without visual query increases the chance of finding similar images.

This completes the analysis on the effect of the two factors on the rate of successful search. The next section describes the analysis of the effect of the two factors on efficiency of search by analysing the time taken to find target images.

7.2.2 Search Efficiency

The raw data of the search time sorted by program and target image can be found in Table 7.5 on the following page. To analyse the search efficiency, the response or dependent variable is obviously the time taken to find the target image. What is less obvious is how to treat the data when the participants did not find the target image. We considered three options in treating such data but used only one. The first option is to naively analyse those data with the assumption that it would take at least as long as the time at which they gave up the search for the target image. The problem with this approach is that these two times have different meanings. If a participant gave up quickly, it indicates that the program failed to keep them interested in the task, so a short give up time is undesirable. On the other hand, if a participant finds the target image quickly, it indicates that the program helps them in completing the task, so a short search time is desirable. It is clear that this option is unsuitable for analysing search efficiency.

The second option for treating the missing data was obtained by consulting two cognitive scientists from the Centre of Bionics Studies at Monash University, Dr Barry Richardson and Dr. Dianne Wullemin. They suggested that the data should be con-

Search Time sorted by target image and program

	P1	P2	P3	P4
fruits and flowers	2m 1.54	1m 2.47	0m 57.99	0m 19.30
	2m 3.16	6m 26.45	1m 1.25	0m 27.57
	2m 8.89	6m 41.03	1m 22.19	0m 35.46
	3m 54.64	10m 39.04	3m 40.11	1m 29.30
	5m 22.09	12m 21.52	5m 58.79	2m 13.63
	14m 19.82	21m 39.77	10m 7.54	2m 26.33
fireman	5m 45.79	1m 22.92	1m 26.95	3m 21.03
	7m 51.83	3m 28.66	5m 1.38	3m 46.15
	8m 11.72	6m 12.83	5m 52.06	10m 24.43
	8m 42.22	6m 44.01	8m 19.43	10m 55.01
	17m 24.28	7m 35.79	10m 28.79	13m 11.29
	24m 5.78	8m 33.41	35m 19.40	19m 52.83
monkey	1m 35.77	0m 17.99	1m 50.98	1m 5.15
	2m 10.53	0m 22.86	3m 27.47	2m 15.85
	2m 47.81	0m 34.87	5m 10.03	3m 7.81
	4m 3.81	1m 16.22	6m 28.47	5m 1.21
	5m 55.36	1m 27.57	8m 53.74	7m 4.46
	7m 3.17	2m 38.51	19m 52.07	18m 13.56
girl	3m 50.83	5m 11.19	0m 36.20	0m 37.84
	5m 14.61	6m 36.99	1m 49.40	1m 25.26
	5m 42.15	9m 10.52	3m 56.74	2m 0.51
	11m 22.93	12m 25.10	5m 10.37	2m 51.19
	14m 8.57	15m 48.75	7m 2.95	8m 40.84
	18m 30.79	19m 2.13	8m 47.89	21m 30

Table 7.5: The time taken to find a target image or give up (in red).

sidered as missing and be replaced by the average searching time of all participants of that program because it is an accepted and commonly used approach by cognitive scientists and experimental psychologists. This function is also available as an option in the SPSS statistical package, so it is fair to assume it is also a widely used practice in the research community. Nevertheless, we still had reservations about the use of the *almost blind* average replacement.

We used the third option for treating the missing data, that is, to replace the missing data with the maximum time taken in that program to find the target image. This option is the most conservative approach compared to the other two. The justification for using this option is as follows. In experiments involving humans in which they have to complete a difficult task and the response variable is the time taken to complete the task, it is common to set a maximum time. When a participant fails to complete the task, the value of the response variable is the maximum time set. Because we did not set the maximum time, we could not use this approach, but we could predict the maximum tolerable time in completing the task. The maximum tolerable time was the maximum time taken to find an image using the program. Using this approach to analyse the search efficiency is extremely conservative considering that a good program is one that can keep users interested in the task. In spite of this, we decided to take this approach because it is more justifiable than any of the other options.

Using the rule defined above, when a participant failed to find a target image, the search time was replaced with the longest time taken in that program to find the target image. From Table 7.5, it can be seen that for P1 the longest time taken was 24 minutes, for P2 it was 21 minutes 40 seconds, and for P4 it was 21 minutes 30 seconds. These choices exceeded the longest time in unsuccessful search within the same program. For P3, the time used to replace the missing data was 19 minutes 52 seconds, the second longest time. The reason for not using the longest time is as follows. The longest search time in P3 was 35 minutes from an unsuccessful search by participant 24, and it is nearly 15 minutes longer than the second longest, which was used to replace the unsuccessful search. Participant 24 used a most unusual search technique: instead of

moving to the area where the image is most likely to be located, she moved to the area where the image is *least* likely to be and started hiding the images. Such a search technique is logical but highly inefficient. This is evident given that the second longest search time from another participant using the same program took only 19 minutes 52 seconds; although it is nearly fifteen minutes shorter, he found the target image.

For P4, the longest time was from participant 21 who refused to practise before performing the task, even though she was strongly encouraged to. After the experiment, she indicated that she could not remember some of the functions, so did not use all the available functions. This is why it took her much longer than other participants to find the target. In this way, she was very different from most participants, so using this search time to replace the missing data represents a very conservative approach.

The following sections first discuss the analysis of the results of the four programs in solving the Page 0 Problem, then in browsing. The nature of analysis on search efficiency is quantitative, which is useful to objectively establish which program is better. This type of analysis, however, reveals nothing about why some systems are better than others. This explanation can only be obtained using the qualitative data collected from the post experiment questionnaires and interviews. In this section, the qualitative analysis will follow the quantitative analysis.

7.2.2.1 In Solving Page 0 Problem

The average search time taken for each program is given in the profile plots in Fig. 7.11 on the following page. It is clear that P2 performs better than P1, P4 better than P2 and P3, and P3 better than P1. Next, we performed a hypothesis test using a two factor ANOVA. The H_0 is that both factors do not affect the search time. The results of the test can be found in Table 7.6 on the following page. The P -value for the type of image layout is 0.009; hence, we rejected the H_0 for the type of image layout. The P -value for the existence of visual query is 0.106. Because this value is very close to 0.01, so we also rejected the H_0 for the existence of visual query. This means P4 is significantly better than P1, P2 and P3, while both P2 and P3 are significantly better

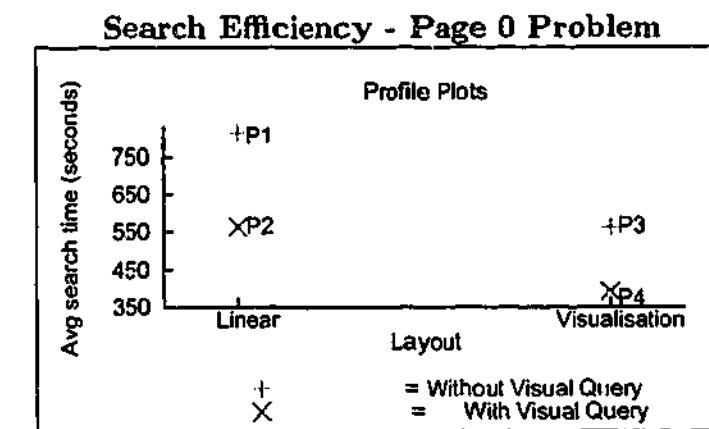


Figure 7.11: Profile plots for comparing the search efficiency of P1, P2, P3 and P4 in searching for exact images, the task used for testing the usefulness of programs for solving the Page Problem.

Search Efficiency - Page 0 Problem
Results of Hypothesis Testing

Factor	P
1 Type of Image Layout	***0.009
2 Existence of Visual Query	*0.106

*** indicates significance at 0.01 whilst * indicates significance close to 0.1.

Table 7.6: Results of two factor ANOVA hypothesis testing.

than P1. The results suggest that to search an image from a collection of 5,000 images, (1) the use of visual query function tends to be faster but the improvement is not as significant as using visualisation layout (eyeMap-based), and (2) eyeMap is significantly better than existing systems.

7.2.2.2 In Browsing

As in the analysis of the rate of successful search in the previous section, we were also interested in how each program performs if the task is relaxed, so that it becomes a browsing task. Recall that the programs logged not only the functions used but also the time at which the functions were used; so, we recovered the time at which a similar image was viewed - the search time is then from the time when such image was first viewed. The treatment for unsuccessful search is the same as before but because of the redefinition, the maximum time for P1 was now 18 minutes 30 seconds - it was 24 minutes. The maximum search times for other programs remain the same.

The data was then reanalysed using this relaxed rule. The profile plots are given in Fig. 7.12 and the results of hypothesis testing are given in Table 7.7. From the profile plots, it is clear that the relationships of the programs' performance remain the same, but the difference between the existence of the visual query factor is now smaller. The P -value for the type of layout is now 0.012, and for visual query is 0.669. The H_0 for the type of layout is rejected with 0.05 level of confidence, but the H_0 for the visual query must be accepted. Thus, for browsing a collection of 5,000 images, the use of visualisation layout is more efficient than visual query; that is, P3 is significantly better than P1, and P4 is significantly better than both P1 and P2. From this analysis, we conclude that eyeMap is significantly faster than existing systems for finding the exact images and for browsing.

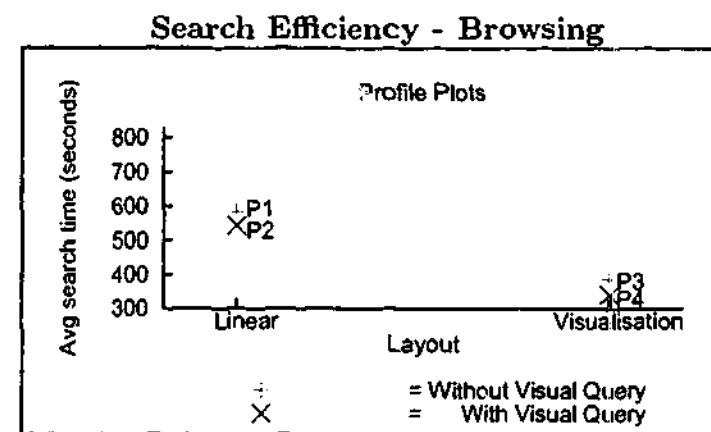


Figure 7.12: Profile plots for comparing the search efficiency of P1, P2, P3 and P4 in searching for similar images, the task used for testing the usefulness of programs for browsing.

Search Efficiency - Browsing		
Results of Hypothesis Testing		
Factor		P
1	Type of Image Layout	**0.012
2	Existence of Visual Query	0.669

** indicates significance at 0.05.

Table 7.7: Results of two factor ANOVA hypothesis testing.

Contrary to our expectation, the use of visual query does not significantly increase the search efficiency and only visualisation layout does. The hypothesis test above can only indicate the strength of evidence against H_0 , but cannot explain why. Such answers can only be revealed by analysing the qualitative data from post experiment

questionnaires and interviews. We analysed two components which may have influenced the search time. The first component was the usefulness of the visual query and visualisation layout, and the second one was the search strategy used. These two components are discussed in the next sections.

7.2.2.3 Usefulness of Visual Query and Visualisation Layout

Participants in the experiment had mixed feelings about the usefulness of visual query. Ten of them thought this function was useful and expressed that the task seemed easier because they only needed to find visually similar images instead of the exact one. On the other hand, when the visual query did not return the expected image, they became irritated. The comments from participant 03, 12 and 13 are given in the following:

03 *With query, (I) expect that it would be easier but when I can't find the target image, it becomes irritating.*

12 *...when doing a query, lots of similar images appear but not the one I am looking for. So, it can be hard to search as well.*

13 *I thought it would be easy with the query tool but it wasn't. I am not sure why.*

The truth of these statements can be seen in the search time for **fruits and flowers**: the average search time in P2, excluding the failed search was almost nine minutes compared to five minutes in P1 (nearly twice as long). We believe that the search time was longer because of the over-reliance on visual query to find the target image. As a consequence, when the visual query fails to return the expected image because of the differences between low level features and human's perception (semantic gap), the search time becomes even longer than for a manual search. On the other hand, with P4, the visual query was not the only tool, the image layout was available too. So, when the visual query does not return the expected results, users can fall back on the image layout. From this point of view, the visualisation layout is also useful for bridging the semantic gap. In fact, the visualisation layout is so attractive that nearly all participants, 83% or 20 out of 24, indicated that the image layout in the

visualisation view was useful:

- 01 ... moved the "ellipse" in the actual image set to a region I felt had a similar colour composition to the set image. ... allowed quick search where we could move straight to a region.
- 04 Scrolled around a similar colour then slowly looked through.
- 06 I used the focal point to get the images that looked the same then went into linear view and then if the target image was not there I used hide image then used the focal point again.
- 09 I just go to the same colour region and find it.
- 15 Firstly, according to the target picture to locate one area and then use the mouse wheel to clarify the picture.
- 23 When looking for fruits and flowers, can go directly to the red section.

However, grouping the images based on visual similarity can be a double-edged sword: while it makes it easy to narrow down a search, it sometimes makes it harder to find the target image among similar ones:

- 05 I was unsuccessful in this program. The colour grouping made it easy to pick where to begin looking. I found MANY similar pictures using this program but could not find the target image.
- 13 Same colour in many pictures.

This explains why searching for monkey took longer in P3 and P4 than in P1 or P2. In P3 and P4, the target was surrounded by many images with similar colours, so the participants found it hard to find. In P1, the images were not displayed by similarity of colours and monkey was surrounded by images with different colours, so it stood out; however, being able to narrow down the search area was more important because it, overall, significantly reduces the search time.

7.2.2.4 Differences in Search Strategies

Although all participants received the same treatment and had equal opportunity to practice, there was some variability in the search strategies, which would influence the search time. The strategy employed depends on several elements among which are personal preference, programs and the functions of the program that users remember. Because the functions provided in P1 and P2 are more similar compared to P3 and P4, it is fair to analyse P1 and P2 together, then P3 and P4.

For P1 and P2, the difference in the search technique appeared to rely on whether users could extract salient features of the images. Those who could tended to scroll quickly and their search techniques appeared to be more efficient. Nine participants indicated that in P1 or P2, they scrolled down fast and looked for the salient features which could be colours or shapes (eight of them indicated it was colours):

- 06 ... I used the scroll bar to scan the images. I looked for a similar overall colours in the picture; for example, if the target image was a picture of a golf ball, I would look for a white image.
- 07 Looked at the basic colour features of the target image and manually searched the database for the corresponding image.
- 14 Remember the main feature of the image; for example, the curve of the water then check one by one but not very carefully but just very quickly. When there is a picture with similar curve, then I looked more carefully; for example, the waterfall has the curve then I looked more carefully.
- 15 Look at whether it is bright or not and look more detailed later. ...

Those who did not utilise the significance of salient features reported that it was very hard to concentrate on an image in the presence of many. Also, they preferred to process a small number of images at a time:

- 02 *Too many images on the screen. Can only concentrate on about a third of the screen. (The) best way to scan is to take in a small chunk so the program should allow people to take in a small chunk at a time.*
- 10 *I scrolled through the images and tried to find images of similar colour and texture. As there were so many pictures this was difficult as there was no way to single out or group similar images.*
- 23 *Scroll down fast but can not remember anything. Can not pay attention.*

Their comments suggest that instead of looking for the salient features, they processed the details of the images. Also, participants who could only process a small number of images at a time suggested a page up or down function to bring a fresh set of images each time. These observations suggest that there are differences in how humans process image collections and the differences have an effect on the choice of tools, so designers of image browsing or search software program must take these differences into consideration. This completes the analysis on search strategies in P1 and P2. The rest of this section discusses the search strategies in P3 and P4.

P3 and P4 had the most functions and their user interface was less familiar; consequently, some participants only realised how the functions might be used after they had started the task:

- 08 *Setting the magnification factor to 4 is very important since there is no query. It is important to get as many pictures as possible in the linear view.*
- 11 *Only realise how it might be used at the end that I could use hide images to narrow things down. So it only get used towards the end.*
- 13 *When I realised I could use the linear view, ... I tried to get ... as many pictures as possible in the focal region and used the linear view to search for it*
- 17 *It then occurred to me that pictures which did not fit could be hidden away. ...*

- 21 *Did not realise I can change the distortion factor ... tried grouping lots of images at once.*

This shows that the learning curve for P3 and P4 was relatively higher than for P1 and P2, but it paid off, as users could find the images faster. The comments from participants 08, 13 and 21 suggest that when searching from a small number of images, searching from linear view is more efficient and a smaller distortion factor (four to eight, as indicated in the function log) is more useful than a high distortion factor - a smaller distortion factor means that they can view more images linearly.

This completes the analysis on users' performance using each program. The next two sections provide the analysis on how users perceive the programs.

7.2.3 Program Ratings

Table 7.8 on the following page shows the average rating of each program using a five point Likert scale, with one being "strongly agree" and five being "strongly disagree" - lower values are more desirable. In total, there are 10 statements: statements one to seven are relevant to all programs; statement eight is relevant only to programs having the visual query function, P2 and P4; and statements nine and ten are relevant only to programs with the visualisation display, P3 and P4.

The ratings for the first seven statements were subjected to a two factor ANOVA, while the ratings for the last three were subjected to a single factor ANOVA. The H_0 for the first seven statements, H_0^{1-7} , is that both factors have no effect on the ratings; the H_0^8 is that the layout factor has no effect on the ratings; and the H_0^{9-10} is that the query factor has no effect on the ratings. The column P^{query} shows the P -value of the query factor. If H_0^{query} is rejected, then the differences between P1 vs P2, and P3 vs P4 are significant. Likewise, the column P^{layout} shows the P -value of the layout factor. If H_0^{layout} is rejected, then the differences between P1 vs P3, and P2 vs P4 are significant.

Statements	P1	P2	P3	P4	<i>p_{query}</i>	<i>p_{layout}</i>
1 The task was reasonable	2.75	1.92	2.21	1.75	***0.006	0.141
2 The program was easy to use	2.75	2.08	2.67	1.79	***0.006	0.487
3 The program was enjoyable to use	3.67	2.42	2.50	1.83	***0.0	***0.003
4 The image layout helps me in deciding where to start searching	4.42	3.42	1.83	1.75	***0.004	***0.00
5 The program made it easy to find the target image	4.58	2.75	2.71	1.87	***0.00	***0.00
6 The magnification tool was useful	2.58	2.08	1.79	1.79	**0.031	***0.005
7 The magnification tool was easy to use	2.00	1.71	1.50	1.58	0.285	*0.053
8 Integration of visual query to browsing query was useful	-	1.71	-	1.50	0.233	-
9 The focal/context view was useful	-	-	2.04	1.79	-	0.110
10 Being able to hide images was useful	-	-	1.88	2.21	-	0.224

Table 7.8: A summary of the questionnaire using five point Likert scale, with one being strongly agree with the given statement and five being strongly disagree (** indicates significance at 0.05 and *** at 0.01)

Statement 1: The task was reasonable.

The first statement is to gauge if the task was reasonable. It is one of the three criteria for establishing if manually searching an image from 5,000 images is sufficient to conduct the experiments. The other two are concerned with the rate of successful search and search time, which were discussed and analysed in the previous sections. We mentioned earlier that the experiments are sensible only if manually searching the images are unreasonable but there is no guideline as to how many images would be sufficient. The pilot study and the analysis of participants' performance using each program in the previous sections indicate that 5,000 is sufficient for this study but we were also interested to know if other users also felt the same.

In the experiment proper, P4 had the best rating at 1.75, followed by P2 at 1.92, P3 at 2.21 and finally P1 at 2.75. Coincidentally, this ranking is consistent with the ranking of the rate of successful search when searching for the exact images; however,

in this ranking, it appears that only the query factor is statistically significant, which means that having the query function makes the task *perceived* to be easier. The results suggest that searching from 5,000 images is sufficient for the study because P1 is ranked the lowest. In addition, the participants also complained about using P1 to search for an image:

08 *I don't like Program 1. It makes me feel tired.*

17 *It is very tedious. Took four passes to find the target image.*

24 *The task was reasonable: being asked to find an image is a reasonable task but it is actually very hard to find an image from so many images.*

The rating for P1 has considerably improved from that in the pilot study. It was now rated at 2.75 instead of 4.5. An interview with the participants revealed that they interpreted the statement differently: six participants gave a rating of one which suggests they strongly agree with the statement but it may not reflect the truth because half of them gave up. One of them said that if she had enough time she would have found it, so from her point of view it was a reasonable task. Note that there is no maximum time limit to perform the task but she was unwilling to spend more than she already had. Others thought that it was the image they had to search for that made the task appear reasonable:

05 *it's a reasonable task to do given more time.*

13 *The picture was easy to find because of the colour I did not have to look at every picture I just needed to locate the colour.*

11 *...I only had to look for a green background.*

Statement 2: The program was easy to use.

The second statement evaluates how easy it is to use the programs. P3 and P4 have more functions than P1 or P2, and Distortion Oriented Displays (DOD) is still

an uncommon user interface. We expected that P3 and P4 would be rated harder to use than P1 or P2. Surprisingly, P4 had the best rating at 1.79, followed by P2, then P3 and, lastly, P1. It turned out that the participants also interpreted this statement differently. They rated the programs based on how useful the programs were in helping them to complete the task, or how much they liked the function or the type of display:

- 06 *There isn't any tools to make the task easier. Not getting any help from the computer. The user interface is easy to use but the ease of use is rated low because it is not a very useful program.*
- 08 *The program is not helpful so it's perceived to be more difficult to use. Looking at so many images is difficult.*
- 11 *Scrolling through heaps of thumbnail is difficult especially after playing with the ellipse.*

Accordingly, the ratings were very similar to that of the first statement, and only the query factor is statistically significant.

There was only one participant who expressed that P4 is difficult to use and rated P4 at four, but rated P1 and P2 at one:

- 03 *With query, expect that it would be easier but when I can't find the target image, it becomes irritating and there are a lot of functions and it is confusing.*

Although most participants interpreted the statement differently from what we intended, it is still fair to conclude that most of them did not have any problems with the user interface of P3 or P4, because if they did they would not have found the programs useful and would have given them low ratings.

Statement 3: The program was enjoyable to use.

The third statement finds out how much they liked or disliked the programs. Again, P4 was ranked highest, followed by P2, then P3 and, lastly, P1. Unlike the previous two

statements, the differences in the ratings are now significant for both factors. We also observed that the ratings for P2 and P3 were now very close, so it is fair to conclude that the visualisation layout is just as attractive to users as the visual query.

Statement 4: The image layout helps me in deciding where to start searching.

The fourth question establishes if users make use of image arrangement based on visual similarity. As expected, P4 was ranked the highest, followed by P3, then P2, and, lastly, P1. The effect of the layout factor was extremely strong given that P_{layout} is very small. This ranking is also reflected in users' comments: as mentioned earlier, 83% of the participants indicated that they used the arrangement of the colours to narrow down their search. Surprisingly, the query factor was also statistically significant: the value of P_{query} is 0.004. The participants' comments reveal that it was the layout of the query results that helped them in searching.

- 02 *The query results layout helps me to concentrate better.*
- 10 *It does and it doesn't. I can look for an image with similar colour content and then use the visual query.*
- 18 *The layout from the query results window is useful.*

Statement 5: The program made it easy to find the target image.

The fifth statement quantifies how useful the programs were, overall, in helping them to complete the task. Again, P4 was ranked the highest, then P3, followed closely by P2 and, finally, P1. Also, both factors are statistically significant. The participants felt that the visualisation layout was just as useful as the visual query function in helping them to look for the target image.

Statements 6: The magnification tool was useful.

The purpose of asking the sixth statement is to find out how useful the magnification tool is. This question is important because the images are displayed in thumbnails. It is even worse in P3 and P4, because in the context region, the images are only half the size of the thumbnails in P1 and P2. The participants thought the magnification tools were equally useful for P3 and P4, followed by P2 then P1. Again, both factors were statistically significant.

The reason that the tool is more useful in P2 than in P1 is because the search strategy in P2 is slightly different from in P1. In P1, participants simply relied on colours to search and only used the tool to check if they had found the target. In P2, they tended to view the images in full size before issuing queries.

- 08 *Use magnification tool to inspect the images more carefully because I need to select one which has similar colour properties to do the visual query.*
- 09 *Use content to search for P2 but for P1 rely on colours no need enlargement tool.*
- 16 *(P1) Didn't use it a lot, not useful.*

Statements 7: The magnification tool was easy to use.

The seventh statement finds out how easy it is to use the magnification tool. This question is important for the same reasons as in statement 6, that is, the images are initially displayed in thumbnails. Its ease of use in P3 and P4 was the highest, followed by P2 and P1. This time, only the layout factor was significant. We are unsure why the layout would make the tool easier or harder to use; it could be that, like the first and second statements, the participants rated the ease of use based on how useful the tool was.

Statement 8: Integration of visual query to browsing query was useful.

The eighth statement measures how useful the visual query function is in P2 and P4. P4 was ranked higher but the difference between the two is not significant. This means that visual query is perceived to be as useful whether it is in the linear layout or visualisation layout.

Statement 9: The focal/context view was useful

The ninth statement evaluates if the functions specific to DOD are used differently when the query function exists. The focal view was perceived to be more useful in P4 than in P3 because the search behaviour is slightly different when the query function exists:

- 6 *With P4, use the focal point to find a sample image. But for P3, mainly use the linear view.*

This difference, however, is only nearly significant at 0.1.

Statement 10: Being able to hide images was useful.

Similar to the previous statement, the purpose of this statement is to evaluate if the functions specific to DOD are used differently when the query function exists. Participants found that being able to hide images in P3 was more important than in P4, but this difference is statistically insignificant. Some of them thought it was useful but had forgotten about it until towards the end of the search. One participant did not use it at all until the next search, which was a P3:

- 11 *Only realise how it might be used at the end that I could use hide images to narrow things down. So it only get used towards the end.*
- 21 *Doesn't use it much.*

Summary of Program Ratings

From the program ratings, it is clear that most participants benefited from the visualisation layout and enjoyed using such a program. They also liked the visual query and expected that it would make their task easier but in reality this was not necessarily true. An analysis of the successful search rate and search efficiency indicates that the visual layout is, overall, the more important factor. The results show the importance of verifying what users said against their performance. Some usability studies only asked users what they felt without measuring their performance [152]; nonetheless, measuring their performance only is insufficient because the information gathered from post experiment questionnaires or interviews or both (as in this section) is much richer, for it provides insights into users' thoughts.

7.2.4 Preferred Program

The last question asked was if the participants have any preference for any method, and if they have, which one. This question was carefully phrased so that users would not think that they must have a preference. For this reason also, they were not asked to rank the programs; nevertheless, it was still possible to gauge whether P3 was suitable for browsing, because P3 is simply a P4 without the visual query: a preference for P4 would also mean a preference of P3 over P1.

It was found that twenty participants preferred P4, one each for P2 and P3, and two had no preference. There were two participants who could not find the target image using P4, and one of them had no preference for any method while the other preferred P4. We found it intriguing that he preferred a program in which he failed to find the target image, so we asked him why. His reply was:

- 15 *Even if I can not find the image using the software, there are still some functions that I can use...*

Sometimes, users found the target image faster using another program, yet they still preferred P4. Some of them suggested that they felt P4 helped them to find a sample

image to start a visual query; for instance, participant 08 found the target image faster in P2 than in P4, yet he still preferred P4. We asked him why and he replied:

- 08 *The target image in P2 is very bright so it is easy to find a sample image. The target image in P4 is not so obvious but the program helps me to find an image to start querying.*

The users' preference could also be explained in that they found the program entertaining thus did not feel that it took them longer. A recent study in psychology found the estimation of time taken to perform a task is affected by the type of task and scientifically proved that "time flies when you are having fun" [16].

The participant who preferred P2 did not like the image overlapping in P4. In fact, several participants expressed their dislike of the overlap but they found that existing tools helped them to tolerate it, and they still gained from the colour chart provided in eyeMap.

- 17 *Dislike the overlapping but having the query counter the effect of overlapping.*
- 22 *Much more searching required than in Program 4. A series of linear searches were performed and the images hidden to make navigation using the ellipse easier. Searching mainly colour based with the ellipse. The target was eventually located inside a linear search.*

Participant 21 is the only one who preferred P3. She used P4 before P3 and when using P4, she did not practice it on her own; as a result, she did not remember all the functions in the program and only realised this when using P3. This explained why she preferred P3 over P4.

7.3 Conclusions

In this chapter, we showed that eyeMap is the best system for browsing and solving the Page 0 problem by conducting a usability study on colour-based eyeMap and existing systems. The study confirms that users can successfully transfer their daily browsing

experience into image browsing using eyeMap because eyeMap has the (1) highest rate of successful searches, (2) shortest successful search time, (3) highest program rating and (4) highest preference. The first two criteria indicate participants' performance in using eyeMap whilst the last two indicate their attitude towards eyeMap. From the analysis of these criteria, it is appropriate to conclude that eyeMap is the best system for browsing and solving the Page 0 problem. Also, eyeMap is useful for bridging the semantic gap between human and computer because when the visual query fails to return the expected image, users can explore the images using eyeMap to find visually similar ones.

In addition, the evaluation study also provides insights into how humans process images. We discovered that there are at least two different ways humans process images. Some look for the salient features in the images and can process many images at a time. Others look at the details of the images and can only process a small number of images at a time; they also prefer to display a new set of images at one time instead of scrolling to the next row of images. Further, we found that when searching from a *small number* of images, linear search is most efficient but when searching from a *large number* of images, users value grouping by visual similarity, as it helps them to narrow down their search. These findings are important for the design of any image browsing and search method.

eyeMap is a browsing concept suitable for use with other types of images as long as the correct feature descriptor is selected; for example, it can be used for browsing texture databases by selecting appropriate texture descriptors. The next chapter demonstrates how to use eyeMap for texture images by focussing on how to use existing texture descriptors for browsing, thus creating texture-based eyeMap.

Texture-Based eyeMap: Browsing Texture Image Databases

The last two chapters concentrated on browsing general colour image databases, and research showed that eyeMap is the best system for browsing and solving the Page 0 problem. The prime reason behind eyeMap's success is that it enables users to transfer their daily browsing behaviour into browsing image databases; therefore, image browsing in eyeMap is intuitive. For this reason, the eyeMap framework can be used for browsing not only general colour image databases but also other image database types in the real world, such as those for texture i.e. textiles, carpets and wall papers.

The aim of this chapter is to show how eyeMap can be used for browsing texture databases by evaluating two types of texture descriptors defined in the MPEG-7 standard. These two descriptors are Texture Retrieval Descriptor (TRD) for retrieval and Texture Browsing Descriptor (TBD) for browsing. The purpose of the evaluation is to establish which texture descriptor is more suitable for browsing. The research described in this chapter is more than just an evaluation study; it also shows how to use TBD for browsing, and then questions the validity of TBD as a feature for browsing. To date, there has been no research which investigates how to use TBD for browsing.

8.1 How to Browse a Texture Image Database

From Chapter 6, it is clear that the quality of the layout generated for display depends on the input into the MDS algorithm. So, to browse texture images, the input into the algorithm is texture features (descriptors). The MPEG-7 standard defines two types of texture descriptors: TRD for retrieval and TBD for browsing. Using TRD for retrieval is straightforward but using TBD for browsing is less straightforward because the browsing process is undefined. This section demonstrates how to extract TRD and TBD from texture images and to generate layouts for browsing using these features. We then studied the layouts to establish which one is more suitable for browsing using the two evaluation criteria described in Section 6.1.1: the spatial PR graphs and visual inspection. For convenience, from this point onwards in this chapter, unless specified otherwise, database means texture image database.

8.1.1 Texture Retrieval Descriptor (TRD)

8.1.1.1 Feature Extraction

TRD is generated from the following process which consists of two steps. The first step in this process is to convolve an image $I(x, y)$ with a set of Gabor filters:

$$I'_{m,n}(x, y) = \sum_{s=0}^{K-1} \sum_{t=0}^{K-1} I(x-s, y-t) g_{m,n}^*(s, t) \quad (8.1)$$

where $I'_{m,n}$ is the filtered image at scale m , orientation n , K is the filter mask size, and $g_{m,n}^*$ is the complex conjugate of:

$$g(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}(x^2/\sigma_x^2 + y^2/\sigma_y^2)} e^{j2\pi Wx}$$

where W is the modulation frequency. The filter for scale m and orientation n is obtained from:

$$\begin{aligned} g_{m,n}(x, y) &= a^{-m} g(\tilde{x}, \tilde{y}) \\ \tilde{x} &= a^{-m}(x \cos \theta + y \sin \theta) \\ \tilde{y} &= a^{-m}(-x \sin \theta + y \cos \theta) \end{aligned}$$

where $a > 1$ and $\theta = \frac{n}{N}\pi$. The frequency response of the texture at different scale and orientation is captured by filters at different m and θ . The variables above are defined as:

$$\begin{aligned} a &= \left(\frac{U_h}{U_l}\right)^{\frac{1}{M-1}} \\ W_{m,n} &= a^m U_l \\ \sigma_{x,m,n} &= \frac{(a+1)\sqrt{2\ln 2}}{2\pi a^m(a-1)U_l} \\ \sigma_{y,m,n} &= \frac{1}{2\pi \tan(\frac{\pi}{2N}) \sqrt{\frac{U_h^2}{2\ln 2} - \left(\frac{1}{2\pi\sigma_{x,m,n}}\right)^2}} \end{aligned}$$

where M and N are the maximum number of scale and orientation. The values of the constants are: $U_l = 0.05$, $U_h = 0.4$, $K = 60$, $M = 4$ and $N = 6$.

The second step for generating TRD involves calculating the energy of the filtered image at scale m and orientation n . The energy is simply the sum of magnitude of the filtered image:

$$E_{m,n} = \sum_{x=0}^{P-1} \sum_{y=0}^{Q-1} |I'_{m,n}(x, y)| \quad (8.2)$$

where P and Q are the image width and height. The response of the texture at each scale and orientation can be summarised as the ratio of the energy at each scale and orientation to the sum of energy at all scales and orientations (8.3), and the feature

vector is (8.3) at different scales and orientations (8.4).

$$\%E_{m,n} = \frac{E_{m,n}}{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} E_{m,n}} \quad (8.3)$$

$$f_{TRD} = \%E_{0,0}, \dots, \%E_{3,5} \quad (8.4)$$

The distance between two TRD feature vectors is calculated using a dissimilarity metric such as $L1$ or Earth Mover's Distance (EMD) proposed by Rubner [117]. The next section describes how to derive a layout using TRD.

8.1.1.2 Deriving a Layout Using MDS

With any MDS algorithm, it is possible to derive a layout if the distance between any two objects are known. As described in the previous section, it is possible to calculate the distance between two textures using TRD, so it is possible to derive a layout using TRD. As in Chapter 6, the MDS algorithm used is the one defined in [88]. In TRD, there are three options used for generating a layout using MDS and we studied all three options in this experiment. The first option for generating a layout is by using the EMD dissimilarity metric to calculate the distance between TRD feature vectors \mathcal{Q} and \mathcal{I} . Rubner suggested that EMD can be used to calculate two texture feature vectors by redefining the ground distance $d_{i,j}$ as [117]:

$$d((m^{\mathcal{Q}}, n^{\mathcal{Q}}), (m^{\mathcal{I}}, n^{\mathcal{I}})) = \Delta m + \Delta n$$

$$\Delta m = |m^{\mathcal{Q}} - m^{\mathcal{I}}|$$

$$\Delta n = \min(|n^{\mathcal{Q}} - n^{\mathcal{I}}|, N - |n^{\mathcal{Q}} - n^{\mathcal{I}}|)$$

where $m^{\mathcal{Q}}$ and $m^{\mathcal{I}}$ are the m scales from feature vectors \mathcal{Q} and \mathcal{I} respectively, while $n^{\mathcal{Q}}$ and $n^{\mathcal{I}}$ are the n orientation from feature vectors \mathcal{Q} and \mathcal{I} respectively.

The second option for generating a layout is by using the $L1$ dissimilarity metric instead of EMD. The problem with EMD is that it is computationally expensive. Calculating the distance between two feature vectors using EMD on an Intel P4, 1.4 GHz PC running Linux 2.4 takes 0.28 ms. Using $L1$, it takes a negligible $1e^{-7}$ ms. If we are

calculating only two feature vectors, the speed gained by using $L1$ is unimportant but the MDS algorithms need to calculate the distances many times ($\frac{N(N-1)}{2}$ times), and this needs to be done iteratively until a suitable layout is found. In this case, using $L1$ will speed up the process of finding a suitable layout.

The third option for generating a layout is by modifying $\%E_{m,n}$ (8.3). The reason for investigating this option is because in Chapter 6, the colour histogram does not capture the differences across different bins, and consequently, the generated layout appears random. For this reason, TRD potentially has the same problem if it fails to capture the differences of energy across different scale and orientation. This problem can be illustrated with the following simple example. Assume that the feature vectors are generated from filters of two scales and two orientations, so the feature vector is $(\%E_{0,0}, \%E_{0,1}, \%E_{1,0}, \%E_{1,1})$. Also assume that all three texture images (\mathcal{A} , \mathcal{B} and \mathcal{C}) respond to only one filter; thus, $f_{TRD}^{\mathcal{A}} = (1,0,0,0)$; $f_{TRD}^{\mathcal{B}} = (0,1,0,0)$; and $f_{TRD}^{\mathcal{C}} = (0,0,0,1)$. The distances between \mathcal{A} and the other two images are $L1(\mathcal{A}, \mathcal{B})$ is 2 and $L1(\mathcal{A}, \mathcal{C})$ is also 2; however, differences of energy from the same scale and orientation (between \mathcal{A} and \mathcal{B}) should be less than differences from different scale or orientation (between \mathcal{A} and \mathcal{C}). EMD, unlike $L1$, can correctly differentiate dissimilarities of energy from the same scale and orientation to those from different scale or orientation: $EMD(\mathcal{A}, \mathcal{B})$ is 1 and $EMD(\mathcal{A}, \mathcal{C})$ is 2. Unfortunately, it is computationally expensive; thus an alternative solution should be considered.

One alternative is by accumulating $\%E$ in the same way we have accumulated the colour histogram (see Section 2.1.1.1 on cumulative histogram). This texture feature vector is known as f_{cTRD} where c stands for cumulative. By using $cTRD$, the feature vectors now become $f_{cTRD}^{\mathcal{A}} = (1,1,1,1)$; $f_{cTRD}^{\mathcal{B}} = (0,1,1,1)$; and $f_{cTRD}^{\mathcal{C}} = (0,0,0,1)$. The $L1$ dissimilarity metric can now capture the differences between the feature vectors more accurately: $L1(\mathcal{A}, \mathcal{B})$ is 1 and $L1(\mathcal{A}, \mathcal{C})$ is 3.

8.1.2 Texture Browsing Descriptor (TBD)

The previous section discussed three options for using TRD to generate layouts for browsing. This section describes the relationship between TRD and TBD, and explains how to use this feature for browsing.

TBD is also extracted from the convolution of the image with Gabor filters as described in Section 8.1.1.1. The relationships between both features are given in Fig. 8.1 on the following page; for a detailed description of TBD, please see [72][p214-223]. The TBD feature extractor *transforms* a set of filtered images into feature vectors understood by human, by extracting three texture properties: regularity or structuredness, coarseness, and directionality. TBD is formally defined as:

$$f_{TBD} = [v_1, v_2, v_3, v_4, v_5] \quad (8.5)$$

$v_1 \in \{1, \dots, 4\}$: four classes of texture which describe the regularity or structuredness of the texture with higher value being more regular or structured.

$v_2, v_3 \in \{1, \dots, 6\}$: two dominant directions of the texture; it is vague what v_3 means if there is only one dominant direction. TBD was derived from Gabor filters with 6 orientations, so the maximum value of v_2 and v_3 is 6.

$v_4, v_5 \in \{1, \dots, 4\}$: quantised dominant scales of the texture along the two main dominant directions with higher value being coarser.

Although the browsing feature is defined in the MPEG-7 standard, the browsing process is undefined in the standard, so it is unclear how to use TBD to browse a database because the monitor is a two dimensional medium and TBD is a five dimensional feature vector. To display the images on a two dimensional medium, the vectors must be reduced to two dimensions as well. This can be done either by using MDS or by selecting only the more meaningful features. These two techniques are described in the next sections.

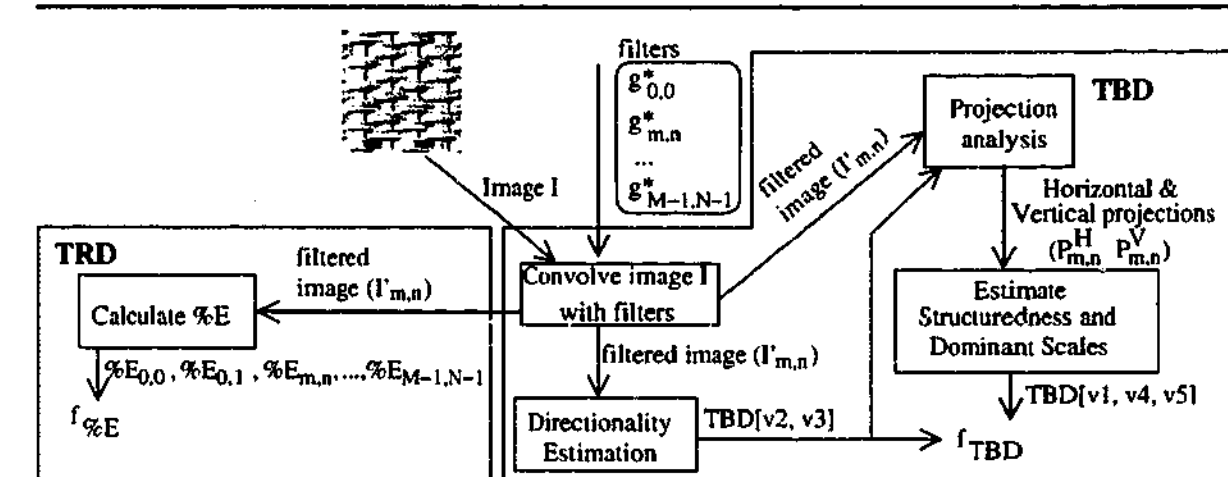


Figure 8.1: Relationship of TRD and TBD.

8.1.2.1 Deriving a Layout Using MDS

The input to the MDS algorithm is TBD. The distance between any two TBD feature vectors A and B is $L1(A, B)$ with some modification for v_2 and v_3 to take into account the circular nature of orientation:

$$L1(v_2^A, v_2^B) = \min(7 - |v_2^A - v_2^B|, |v_2^A - v_2^B|) \quad (8.6)$$

$$L1(v_3^A, v_3^B) = \min(7 - |v_3^A - v_3^B|, |v_3^A - v_3^B|) \quad (8.7)$$

8.1.2.2 Deriving a Layout by Feature Selection

Another possible method of reducing the number of dimensions is to use only the more meaningful features. The dominant directions (v_2 and v_3) can be discarded for three reasons. First, unstructured textures do not have dominant directions. Second, perceptually similar textures which have been rotated would have different dominant directions, and they would have been considered different which is incorrect. Third, the structuredness measure, v_1 , to a certain extent already encodes information about directionality. The scale information, v_4 and v_5 , can be reduced to a single value by adding them up. Eventually, the feature vectors are in two dimensions: v_1 and $scale = v_4 + v_5$.

The values of v_1 , v_4 , and v_5 given in (8.5) are obtained from a very coarse quan-

tisation process, so if these values are used, then the maximum possible number of combinations is only 28 i.e. 4 levels of regularity \times 7 scales (2, ..., 8). Using the quantised values is clearly insufficient for browsing, so we used the values before quantisation with *scale* being the x axis and v_1 being the y axis, where a higher value of x means the texture is coarser, and a higher value of y means the texture is more structured.

We implemented TRD and used the program in [72] to extract TBD, and generated five layouts to determine which feature is more suitable for browsing. The next section describes the experiment set up and evaluation criteria.

8.2 Experimental Setup and Evaluation Criteria

A summary of the layouts studied in this chapter is given in Fig. 8.2. In total, five layouts were generated: three using TRD and two using TBD. All three TRD layouts (MDS-EMD, MDS-L1 and MDS-Cum) and one TBD layout (MDS-TBD) are generated using MDS. Note that there is no need to generate the layout for TBD when we select only the meaningful features because the feature vectors are already in two dimensions - this layout is called 2D-TBD. For TRD, the feature vector is rotationally normalised using the circular shift algorithm described by Zhang et al. [160]. Because the TBD feature itself was derived from Gabor filters, to ensure that all parameters are the same, we modified the filter mask size (K) to 60 as opposed to 80 as defined by the MPEG-7 standard. A detailed study on the choice of filter mask size (K), number of scale (M) and orientation (N) is given in [21].

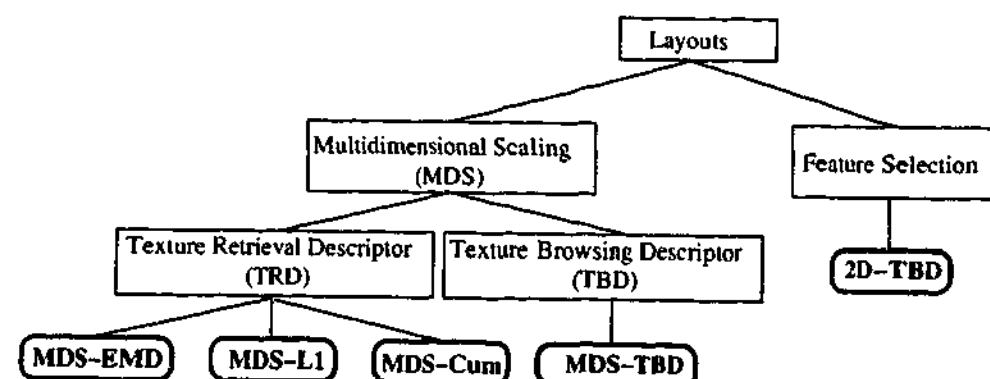


Figure 8.2: The five layouts studied in this chapter. Three layouts were generated using TRD and two using TBD. One layout from TBD was generated using MDS and the other one by selecting only the more meaningful features.

The experiments were conducted on the Brodatz texture database, which contains 1852 images, commonly used by the texture retrieval community [117, 71, 160]. The images in the database were derived as follows. Initially, there were 112 images of 512 \times 512 pixels; then, each image was partitioned into 16 sub-images of equal size. In addition, 60 images were created by rotating existing images. The sub-images from the same original texture form the relevant images, and their names have the format of *dxxx-xx*. The first letter is common to all images, and the first three digits indicate the texture type numbered from 1 to 112 - relevant images have the same number. The last two digits indicate the sub-images ranging from 00 to 16, and sub-images numbered 16 are rotated textures.

The two evaluation criteria described in Section 6.1.1 are also applicable here. To recap, they are spatial PR graphs and visual inspection. For visual inspection, instead of looking for arrangement of colours, we would be looking for the arrangement of textural properties such as scale, directions and structuredness.

8.3 Results and Discussion

This section compares each layout using the two evaluation criteria i.e. spatial PR graphs and visual inspection.

8.3.1 Spatial PR Graphs

Figure 8.3 on the following page shows the spatial PR graphs of the five layouts. To show that the five layouts generated were not arranged by chance, we compared their spatial PR graphs with the PR graph of a randomly generated layout. It is clear that the five generated layouts were not arranged by chance as they all have higher spatial precision than the randomly generated layout. It is also clear that both MDS-EMD and MDS-L1 have much higher spatial precision, indicating that in both layouts relevant images are located closer to each other. The graphs also suggest that although the spatial precisions of MDS-TBD and 2D-TBD are very low, they are not random

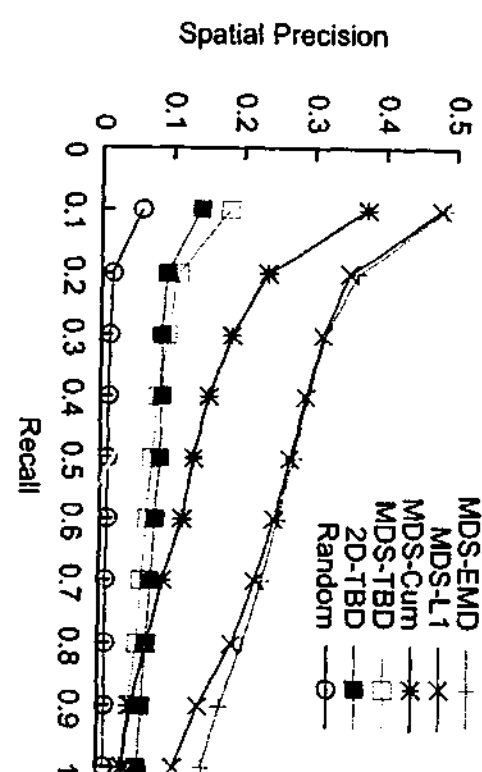


Figure 8.3: Spatial PR graphs of the layouts studied and a randomly generated layout. because they are more effective than the layout generated randomly.

8.3.2 Visual Inspection

The previous section contains the comparison of each layout using spatial PR graphs. In this section, we compare the performance of the layouts by visually inspecting each layout. Figures 8.4 to 8.8 on the following pages show the layouts of the Brodatz texture images in MDS-EMD, MDS-L1, MDS-Cum, MDS-TBD, and 2D-TBD. Each layout will be discussed in the following sections.

8.3.2.1 MDS-EMD Layout

For MDS-EMD (see Fig. 8.4), the layout appears meaningful. MDS does not identify which are the axes nor what they mean, so to understand the layout better, we added some "virtual axes" by inspecting the layout. We use dashed lines to plot the virtual axes in order to differentiate them from the *real* axes used in 2D-TBD. The layout appears meaningful because textures made up of fine textures are located correctly as identified by the scale axis, where the larger value of scales means coarser texture. The other axis seems to indicate the strength of dominant direction(s), and the value

increases in the direction of the arrow. When the value is small, the textures have only one very clear dominant direction, and as the value increases, the second direction, though not strong, starts appearing i.e. bricks texture. When the value is large, the textures have two very strong dominant directions. In between the two extremes, the textures do not have a clear single dominant direction: they are unstructured. This observation is true except for some incorrectly placed textures (outliers), which are enclosed within the dotted polygon. The outliers are so few that the layout is, overall, contextually meaningful. After visually inspecting all layouts, in Section 8.3.2.6 we explain why this group of textures are placed incorrectly.

8.3.2.2 MDS-L1 Layout

For MDS-L1, in Fig. 8.5, the layout also appears less random. It could even be argued that it is better than MDS-EMD as it is more spread out. Also note that the group of problematic textures

in EMD are also located incorrectly in this layout, and this will also be analysed in Section 8.3.2.6 after visually inspecting all layouts.

The use of $L1$ for TRD appears to have no adverse effect on the layout and this could be explained as follows. The example given in Section 8.1.1.2 which illustrates the problem of using $L1$ for TRD, is an extreme case as it is assumed that each texture responds to only one filter. In reality, a homogeneous texture convolved with filters of four scales and six orientations responds to more than one filter, so it has a different physical meaning to that of colour histogram. Colour histogram measures *how many* pixels each bin has, and %E measures *how strong* the response of the texture is at each scale and orientation.

8.3.2.3 MDS-Cum Layout

For cumulative %E in Fig. 8.6, it is only possible to assign meaning to one axis i.e. the scale, and it is unclear what meaning the other axis conveys. Overall, this layout is less contextually meaningful compared to MDS-EMD and MDS- $L1$. It is less contextually mean-

ingful than MDS- $L1$ because, as explained earlier, %E has different physical meaning to that of colour histogram. Consequently, accumulating the energy to the next scale and orientation destroys the orientation information.

8.3.2.4 MDS-TBD Layout

For MDS-TBD (see Fig. 8.7), it is difficult to extract any meaning from the display as the images seem to be placed randomly. We hypothesise that this happened because the dominant directions, as mentioned earlier, are sometimes meaningless: both v_2 and v_3 are meaningless if the texture is unstructured, and v_3 is meaningless if the texture has only one dominant direction. If this hypothesis is true, it means that in 2D-TBD, the layout of textures which have two dominant directions will be meaningful. The truth of this hypothesis is examined by visually inspecting the 2D-TBD layout.

8.3.2.5 2D-TBD Layout

For 2D-TBD (see Fig. 8.8), the x axis is the scale and the y axis is the structuredness of the texture (the values of both x and y axes increase in the direction of the arrows). The bottom left quadrant of the layout is meaningful in that it consists of small scale and structured textures. As the texture becomes unstructured, the layout loses its meaningfulness; for example, on the top left, several images look rather coarse yet they are located in close proximity to textures which are very fine. In fact, the top half of the layout does not conform with visual perception at all: it is contextually mean-

ingless and similar textures appear scattered. However, compared to MDS-TBD in Fig. 8.7, 2D-TBD is more contextually meaningful. It appears that removing the dominant directions has slightly improved the layout. This confirms the hypothesis that the dominant directions which are sometimes meaningless cause the display to appear random. The scale estimation algorithm relies on the correct detection of the two principal directions to estimate the value of *scale*. Because unstructured texture does not have any principal direction, as mentioned in Section 8.1.2, the estimated *scale* is unreliable.

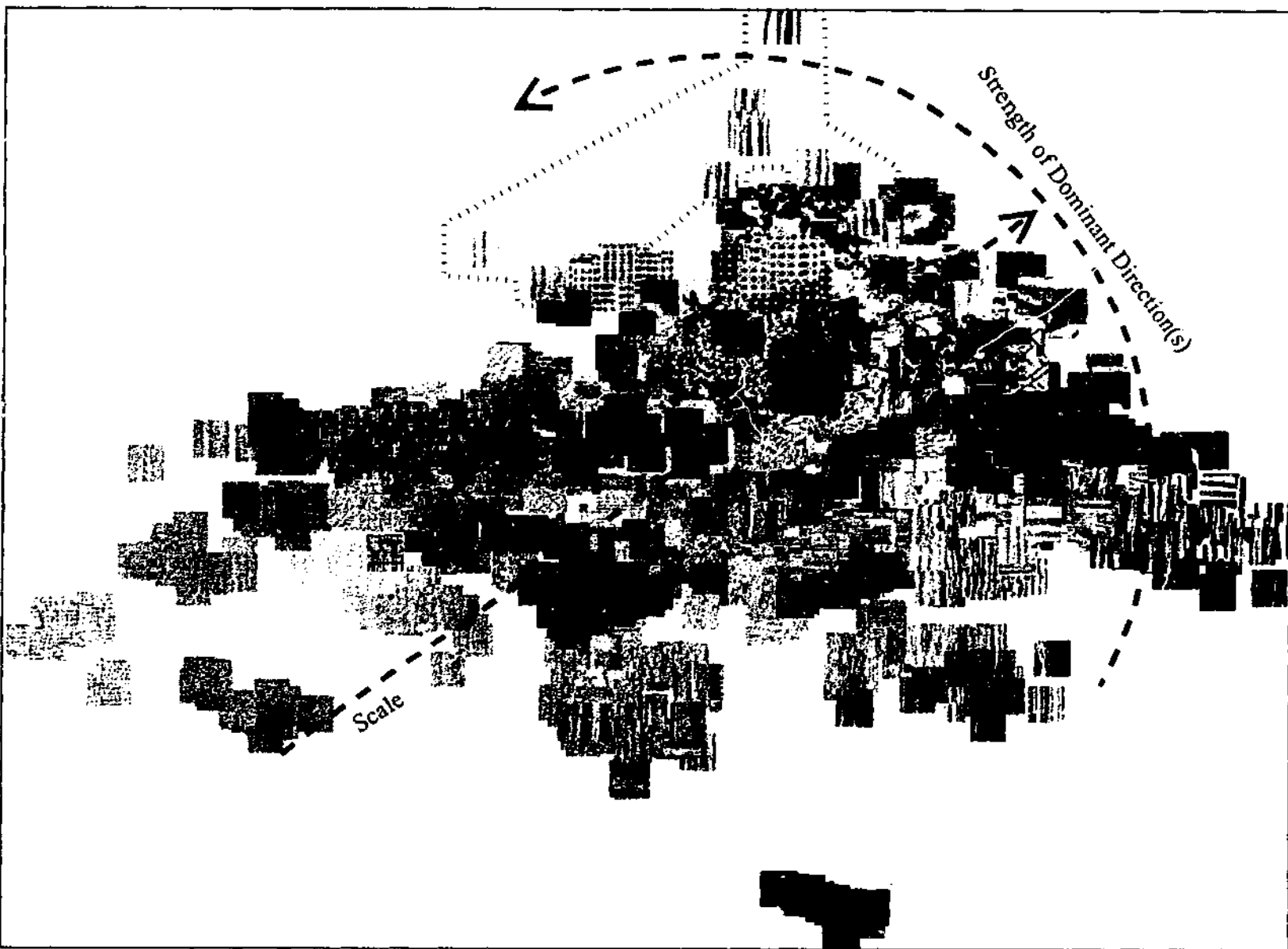


Figure 8.4: MDS-EMD and this layout appears contextually meaningful. The two axes are *virtual* because they are not identified by the MDS algorithm; we identified the axes after a visual inspection.

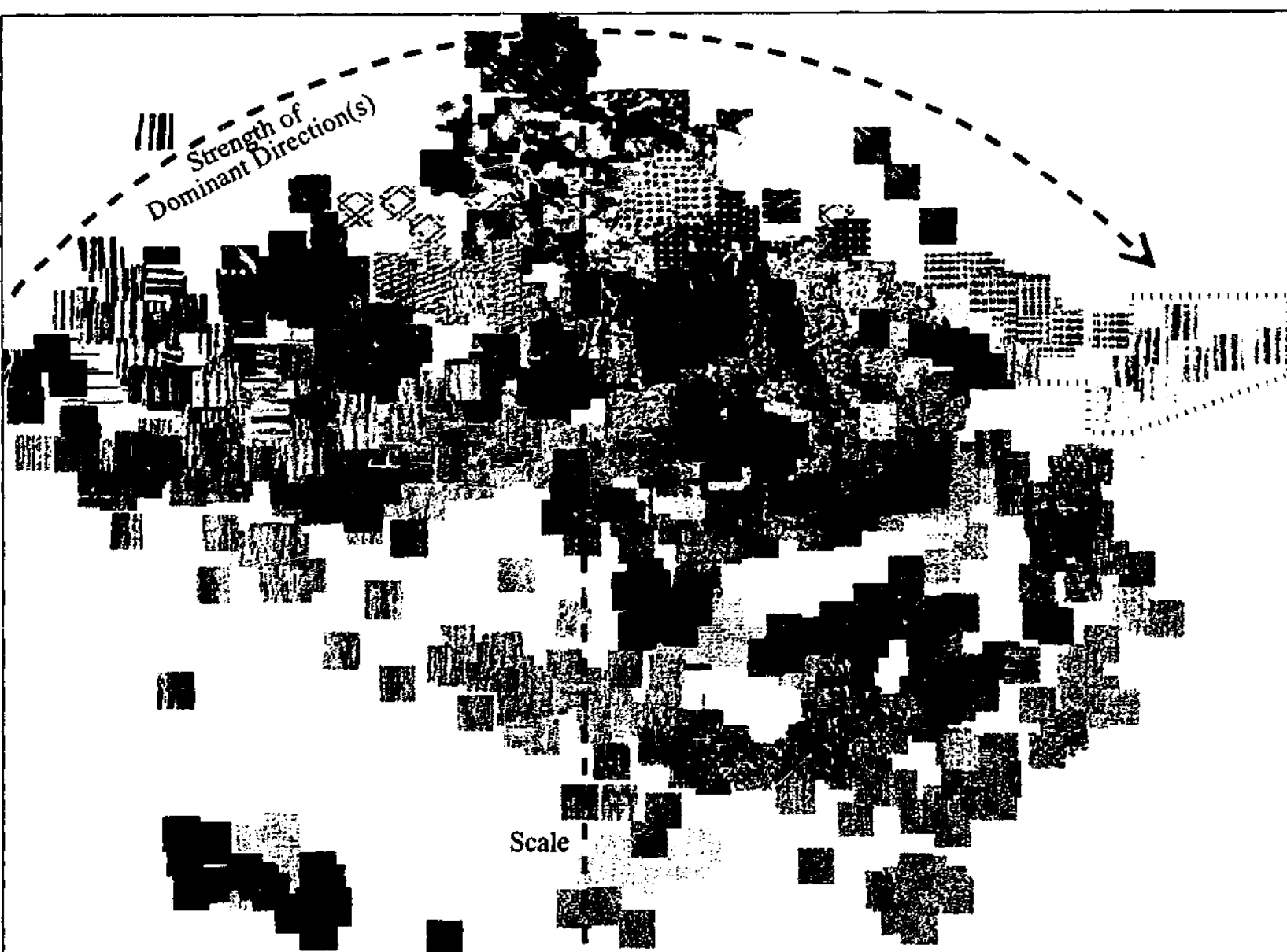


Figure 8.5: MDS-L1. Like MDS-EMD, the two axes are also *virtual*. The texture also appears contextually meaningful

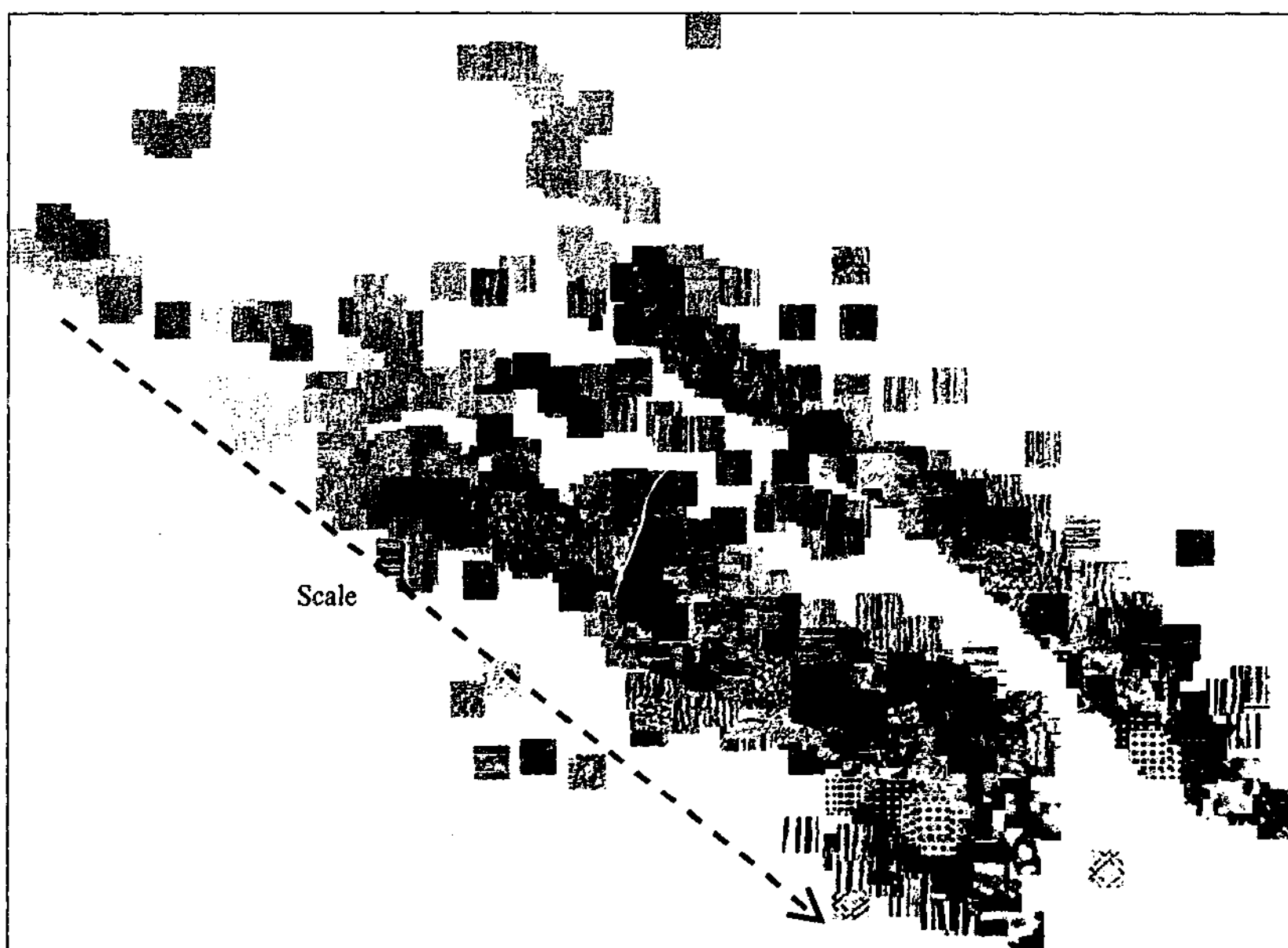


Figure 8.6: MDS-Cum. It is only possible to assign meaning for one axis, scale.

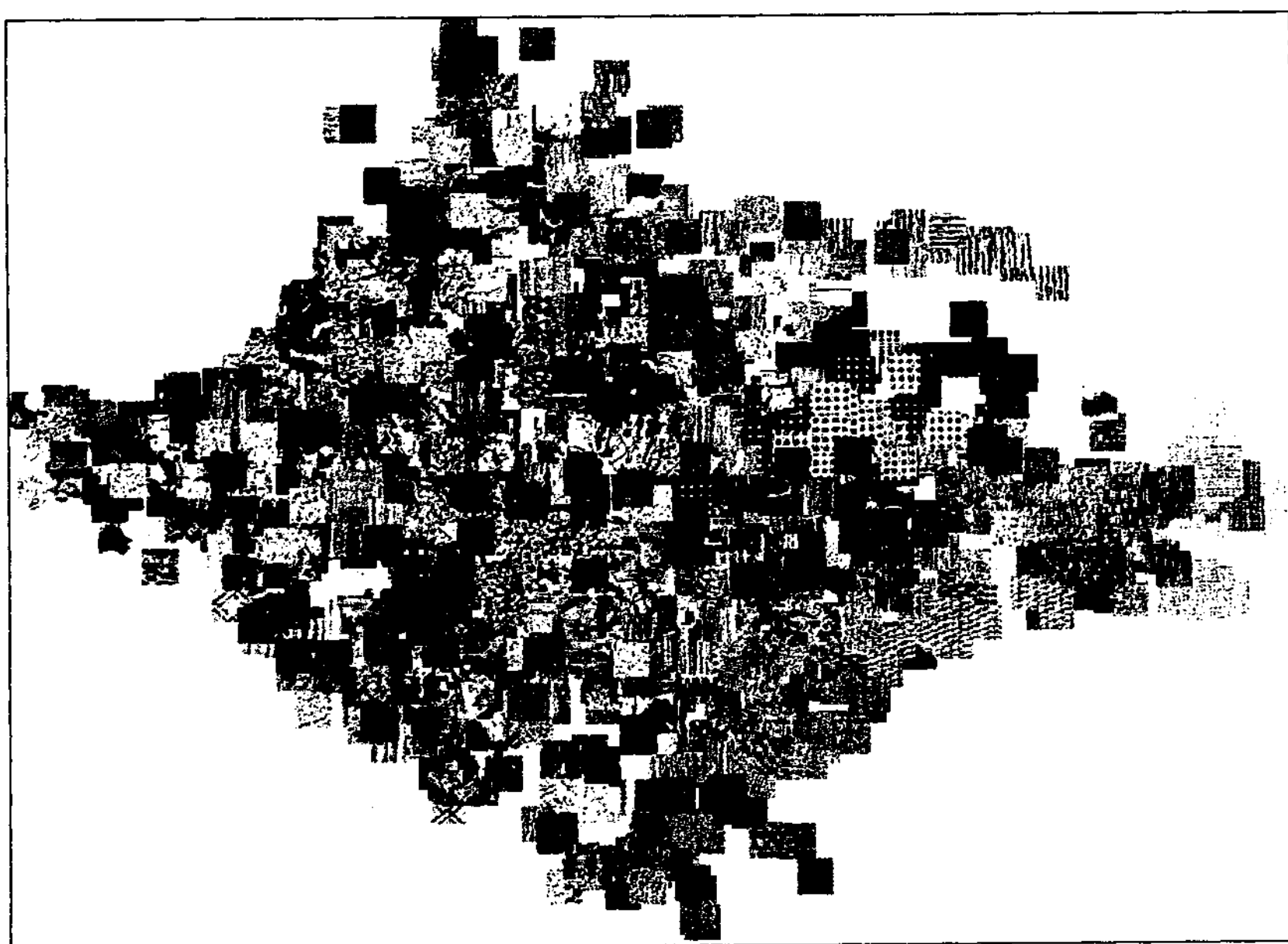


Figure 8.7: MDS-TBD, the lack of contextual meaningfulness is evident as the display appears random.

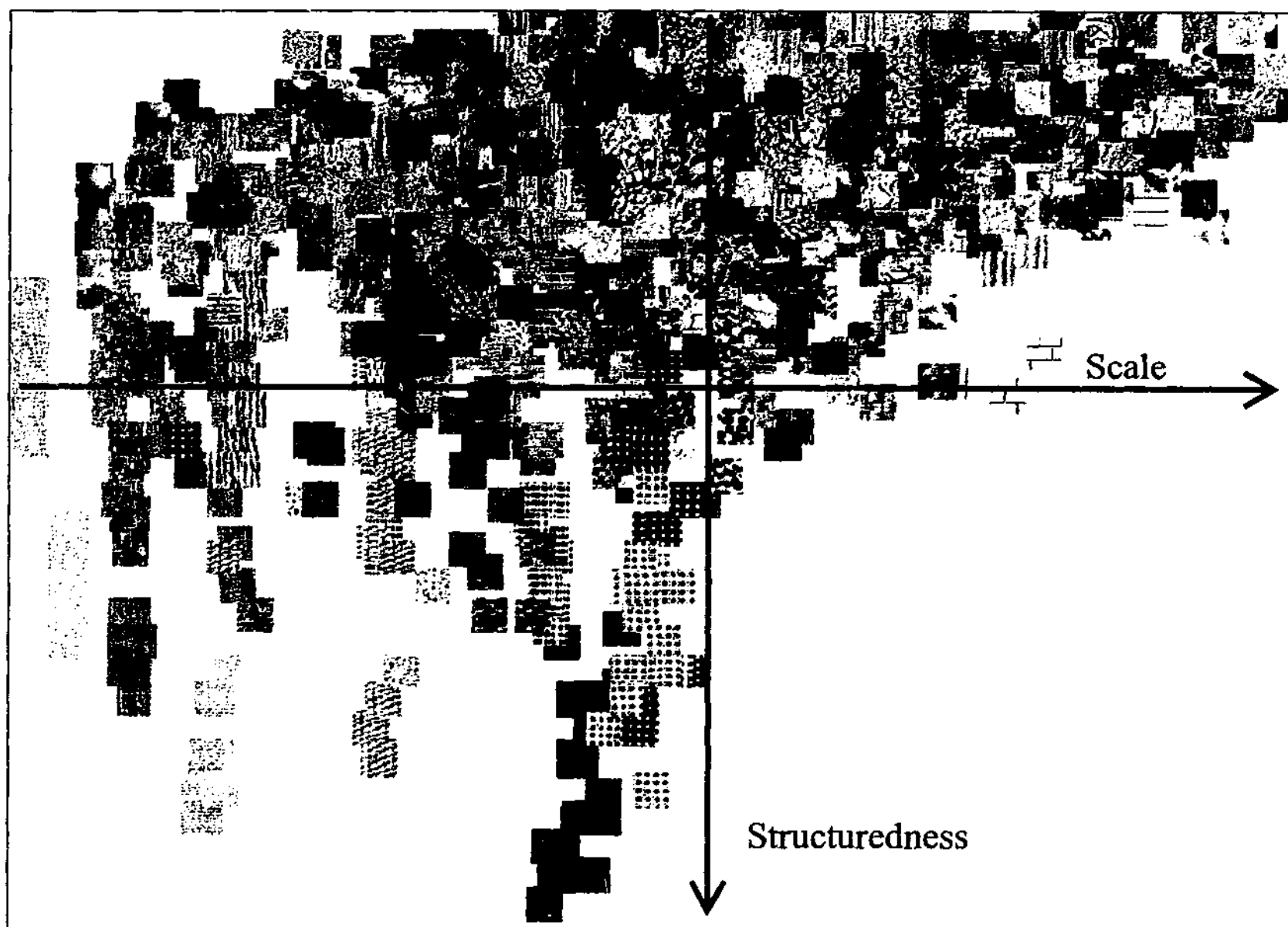


Figure 8.8: 2D-TBD, the two *real* axes were obtained from selecting the more meaningful features. The display is only meaningful for structured textures which are in the lower quadrant of the display.

8.3.2.6 Outliers in MDS-EMD and MDS-L1

The previous sections discuss the comparisons of the five layouts using visual inspection. We found that both MDS-EMD and MDS-L1 are, overall, more contextually meaningful; however, a small number of textures in MDS-EMD and MDS-L1 are placed incorrectly. This section explains why these textures are placed incorrectly.

The group of textures placed incorrectly, outliers, in MDS-EMD and MDS-L1 within the polygons (see Fig. 8.4 and 8.5), belong to texture type d50. This happened because Gabor filters sometimes fail to accurately capture texture properties, hence *visually dissimilar* textures sometimes have similar feature vectors. This means that occasionally, visually dissimilar textures are considered more similar than visually similar textures regardless of which dissimilarity metrics are used.

To demonstrate this problem, Table 8.1 on the following page shows the EMD distances of all 17 textures belonging to texture type d50. Although they belong to the same texture type, their distances are non-uniform; for example $EMD(01,02)$ is only 0.08 but $EMD(01,00)$ is 1.18. To highlight the distances ≥ 0.9 , they are printed in red. The EMD distances between the outliers in the MDS-EMD layout (that is, 01, 02, 03, 06, 07, 11, 13, 14, 15)

and texture 00 are all ≥ 0.9 . Yet, the EMD distances between these outliers and their nearby textures in the layout, which are visually different, are < 0.7 (see Table 8.2). As $EMD(d50-01, d20-11) < EMD(d50-01, d50-00)$, based on the feature vectors, d50-01 is considered more similar to d20-11 than it is to d50-00. Thus, MDS placed it close to d20-11. The same observation can be made for the other outliers as well.

The outliers in MDS-L1 (see Fig. 8.5) were placed incorrectly for the same reason as above. The $L1$ distances between the outliers (that is, 01, 02, 03, 06, 07, 11, 13, 14, 15) and texture 00 are all > 0.6 , and they are highlighted in red in Table 8.3. Like their EMD distances, although they belong to the same texture type, the $L1$ distances are also non-uniform: $L1(01,02)$ is only 0.07 but $L1(01,00)$ is 0.98. However, the distances between 01 and nearby textures in the MDS-L1 layout, which are visually dissimilar, are all < 0.6 as given in Table 8.4. As in EMD, $L1(d50-01, d20-11) < L1(d50-01, d50-00)$ so judging from the feature vectors, d50-01 is more similar to d20-11 than it is to d50-00. Therefore, MDS placed d50-01 close to d20-11. The same can be said for the rest of the outliers.

	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16
00	0.00																
01	1.15	0.00															
02	1.11	0.08	0.00														
03	1.09	0.10	0.05	0.00													
04	0.20	1.30	1.33	1.30	0.00												
05	0.18	1.18	1.12	1.08	0.22	0.00											
06	0.39	0.21	0.16	0.15	1.18	0.07	0.00										
07	1.01	0.32	0.26	0.23	1.07	0.87	0.18	0.00									
08	0.20	1.19	1.13	1.10	0.20	0.06	0.08	0.88	0.00								
09	0.19	1.30	1.24	1.20	0.10	0.12	1.09	0.08	0.11	0.00							
10	0.07	1.28	1.22	1.19	0.17	0.16	1.07	1.00	0.16	0.16	0.00						
11	1.10	0.20	0.14	0.13	1.20	0.08	0.14	0.14	0.09	1.10	1.00	0.00					
12	0.25	1.08	1.02	0.98	0.32	0.13	0.87	0.79	0.11	0.22	0.22	0.88	0.00				
13	1.09	0.51	0.44	0.41	0.01	0.02	0.29	0.21	0.03	0.03	1.07	0.31	0.87	0.00			
14	1.05	0.34	0.28	0.25	1.05	0.88	0.16	0.11	0.89	0.07	1.03	0.14	0.83	0.17	0.00		
15	0.90	0.45	0.38	0.35	0.05	0.74	0.23	0.18	0.76	0.85	0.88	0.25	0.67	0.20	0.17	0.00	
16	0.01	1.22	1.16	1.12	0.21	0.20	1.02	1.02	0.20	0.20	0.08	1.11	0.25	1.12	1.07	0.92	0.00

Table 8.1: The EMD distance of the seventeen textures belonging to d50. Distances ≥ 0.9 are highlighted in red.

	d71-08	d20-00	d20-07	d20-16	d51-01	d50-01	d50-02	d50-03	d50-06	d50-07	d50-11	d50-13	d50-14	d50-15
d71-08	0.00													
d20-00	0.50	0.00												
d20-07	0.51	0.03	0.00											
d20-16	0.50	0.00	0.03	0.00										
d51-01	0.81	0.80	0.81	0.80	0.00									
d50-01	0.53	0.71	0.71	0.70	0.29	0.00								
d50-02	0.47	0.63	0.63	0.63	0.36	0.08	0.00							
d50-03	0.45	0.63	0.63	0.63	0.39	0.10	0.05	0.00						
d50-06	0.33	0.54	0.55	0.53	0.50	0.21	0.16	0.15	0.00					
d50-07	0.28	0.57	0.60	0.57	0.62	0.32	0.26	0.23	0.18	0.00				
d50-11	0.34	0.67	0.69	0.66	0.49	0.20	0.14	0.13	0.14	0.14	0.00			
d50-13	0.24	0.67	0.69	0.67	0.80	0.51	0.44	0.41	0.29	0.21	0.31	0.00		
d50-14	0.24	0.63	0.65	0.62	0.64	0.34	0.28	0.25	0.16	0.11	0.14	0.17	0.00	
d50-15	0.11	0.47	0.49	0.47	0.74	0.45	0.38	0.35	0.23	0.18	0.25	0.20	0.17	0.00

Table 8.2: EMD distances of the outliers and nearby textures, which are visually dissimilar. All distances are < 0.9 .

00	0.00																
01	0.08	0.00															
02	0.02	0.07	0.00														
03	0.02	0.08	0.06	0.00													
04	0.25	0.00	0.03	0.02	0.00												
05	0.20	0.08	0.03	0.02	0.14	0.00											
06	0.03	0.20	0.15	0.18	0.03	0.00											
07	0.00	0.22	0.19	0.17	0.00	0.15	0.00										
08	0.22	0.03	0.05	0.05	0.15	0.09	0.00	0.00									
09	0.23	0.03	0.08	0.08	0.07	0.10	0.00	0.12	0.00								
10	0.13	0.07	0.09	0.09	0.15	0.15	0.00	0.20	0.15	0.00							
11	0.08	0.17	0.15	0.14	0.08	0.11	0.16	0.00	0.00	0.00							
12	0.26	0.01	0.05	0.05	0.21	0.15	0.00	0.11	0.17	0.24	0.00						
13	0.00	0.33	0.30	0.28	0.00	0.00	0.21	0.00	0.00	0.19	0.00	0.00					
14	0.03	0.23	0.19	0.18	0.00	0.00	0.12	0.13	0.00	0.00	0.12	0.00	0.00				
15	0.00	0.33	0.28	0.29	0.00	0.00	0.18	0.20	0.00	0.00	0.23	0.53	0.13	0.17	0.00		
16	0.03	0.00	0.00	0.00	0.23	0.18	0.00	0.00	0.19	0.21	0.10	0.00	0.00	0.00	0.00	0.00	0.00

Table 8.3: The L1 distance of the 17 textures belonging to d50. Distances ≥ 0.6 are highlighted in red.

d20-11	0.00												
d20-15	0.06	0.00											
d20-08	0.05	0.06	0.00										
d71-14	0.53	0.51	0.50	0.00									
d50-01	0.57	0.58	0.55	0.43	0.00								
d50-02	0.52	0.53	0.50	0.39	0.07	0.00							
d50-03	0.54	0.54	0.51	0.38	0.08	0.00							
d50-06	0.48	0.49	0.45	0.31	0.20	0.15	0.18	0.00					
d50-07	0.49	0.49	0.46	0.30	0.22	0.19	0.17	0.15	0.00				
d50-11	0.53	0.53	0.50	0.37	0.17	0.15	0.14	0.11	0.16	0.00			
d50-13	0.60	0.57	0.56	0.25	0.33	0.30	0.28	0.21	0.21	0.19	0.00		
d50-14	0.47	0.48	0.45	0.32	0.23	0.19	0.18	0.12	0.13	0.12	0.15	0.00	
d50-15	0.51	0.48	0.48	0.19	0.33	0.28	0.29	0.18	0.20	0.23	0.13	0.17	0.00

Table 8.4: L1 distances of the outliers nearby textures, which are visually dissimilar. All distances are < 0.6 .

8.3.3 Summary and Implications

Based on both evaluation criteria, TRD using EMD and $L1$ (MDS-EMD and MDS- $L1$) are most suitable for browsing because they have the highest spatial precisions and their layouts are most contextually meaningful. This finding is interesting because it shows that TBD, which is designed for browsing, is *less* suitable for browsing than TRD.

Another way of looking at the TBD feature extraction method is that it is a type of dimension reduction technique, much like MDS algorithms, but it is a very specialised one. It reduces the original data in the high dimensions into low dimensions which have defined meanings. Other types of dimension reduction techniques, such as MDS algorithms, are generic in that they reduce the original data into a set of low dimensions without knowing the meanings of the low dimensions (generic). It turns out that this generic approach is more useful than the TBD feature extraction method because it successfully discovers prominent texture features.

8.4 Conclusions

This chapter extends the use of eyeMap to browsing texture image databases by evaluating the layouts generated using different texture features. It described two methods for generating layouts for texture images using two texture descriptors defined in the MPEG-7 standard: Texture Retrieval Descriptor (TRD) and Texture Browsing Descriptor (TBD). The layouts were then subject to quantitative and qualitative evaluations. This study is more than just an evaluation because it also demonstrated how to use TBD for browsing and questions the validity of using TBD for browsing.

We conclude that TRD is more suitable than TBD for browsing. Because the physical meaning of energy is different from that of histogram, TRD can be used without accumulating the energy of each scale and orientation. In fact, doing so destroys the orientation information. Both EMD and $L1$ are valid distance metrics for MDS algorithms as the qualities of the generated layouts are comparable. It could even be

argued that the layout generated using $L1$ is better than that of EMD because it is more spread out. However, if speed is a concern, $L1$ should be used because EMD is more computationally expensive.

The study also found that the dominant directions features in TBD are inappropriate for browsing, and only the scale and structuredness features are useful for browsing. By removing the dominant directions, TBD is good for browsing only if the textures are structured. The implication of these findings is that MDS, a generic dimension reduction technique, is more accurate than the TBD feature extraction method, a specialised dimension reduction technique, for discovering texture features because it succeeds where the TBD feature method fails.

Conclusions

9.1 Summary of Main Findings

The research reported in this thesis aimed to bring CBIR systems one step closer to real world applications by improving the effectiveness and efficiency of colour-based feature extraction methods, as well as by improving retrieval methods by formulating a framework to facilitate users for browsing and finding a sample image to initiate a visual query. This section summarises the research that contributes to this goal.

The research for improving the effectiveness and efficiency of colour-based feature extraction methods started in Chapter 3. In this chapter, we answered the most fundamental question, that is which colour space is most suitable for colour-based CBIR by evaluating six colour spaces: RGB, LUV and LAB in Cartesian coordinates, as well as HSV, LUV and LAB in polar coordinates (pLUV and pLAB). A colour space is considered suitable for CBIR if the feature vectors generated from its quantised space are both effective and efficient. We conclude that HSV colour space is, overall, most suitable for colour-based CBIR because it is at least as effective as but more efficient than any of the other colour spaces. The recommended quantisation option for HSV is quantising the hue axis into 18 intervals, the saturation and value axes into three intervals, as the best compromise between effectiveness and efficiency. The finding that HSV is the most suitable colour space for colour-based CBIR is significant because it has always been assumed that perceptually uniform colour spaces proposed by colour scientists (LUV, LAB, pLUV and pLAB) are more suitable for colour-based CBIR. Our findings suggest that such an assumption is untrue because those colour spaces

have different purposes from CBIR. In colour science, even small colour differences are important whereas in CBIR, the small colour differences are less important, which is why in CBIR colour quantisation is necessary.

We then progressed to improving the effectiveness of colour-based feature extraction methods in Chapter 4 by making use of spatial relationships of colours. The findings in the colour space studies enable us to make informed decision about the choice of colour space and quantisation parameters. In this chapter, we proposed I-autocorrelogram (I-auto) which describes the distribution of colours and their spatial relationships. An evaluation of I-auto in comparison with other contemporary techniques establishes that it is most preferred.

CBIR systems are mainly used for image retrieval and the most intuitive method for this task is by using query-by-example. The main problem with this process is that users do not always have a sample image to initiate a query (the Page 0 problem). In Chapter 5, we formulated the specification of eyeMap, an image browsing framework, by using the paradigm of daily browsing behaviour in shops or libraries so that users can transfer that behaviour into image browsing. This paradigm demands that all images in a database are arranged systematically to enable users to visualise the content of the database. It also requires that users can navigate the database. To support visualisation, eyeMap displays the content of the entire database systematically so that users have a quick overview of the database. Systematic arrangement in eyeMap is achieved by carefully selecting a suitable feature in evaluation studies; for example, in Chapter 6, to implement a colour-based eyeMap, we evaluated several colour features. To support navigation, eyeMap provides a user interface for interactive navigation. Because eyeMap is capable of displaying large numbers of images, it can also be used to find a sample image to initiate a visual query, thus solving the Page 0 problem.

In chapter 6, as mentioned before, we determined which colour feature, among several, is more suitable for browsing. The colour features evaluated were colour histogram, colour moments, EMD-based colour signature and cumulative histogram. It was found that cumulative histogram is most suitable for browsing colour images be-

cause the layouts generated using this feature have high spatial precision and contextual meaningfulness. We then implemented colour-based eyeMap, a fully functional image browser for large scale general colour image databases. eyeMap was integrated with a CBIR system developed using I-auto, so it can be used to solve the Page 0 problem.

In Chapter 7, we showed that eyeMap is useful for browsing and solving the Page 0 problem by conducting a usability study. In this study, the participants had a very specific task, that is, to find a target image using systems developed based on traditional linear methods and the eyeMap framework. An analysis of their performance in and perception of each system determined that eyeMap is the most effective, efficient and preferred method for browsing and solving the Page 0 problem; thus, the research establishes that eyeMap is the best approach for performing these two tasks. These positive results are indications that eyeMap has successfully enabled users to transfer their daily browsing behaviour into image browsing.

As eyeMap is an image browsing framework, it can be used for browsing other databases such as textures (textiles, carpets or wallpapers) by using a suitable feature: this version of eyeMap is known as texture-based eyeMap. To show how to use eyeMap for texture images, we investigated two texture descriptors in Chapter 8. These two texture descriptors are defined in the MPEG-7 standard: Texture Retrieval Descriptor (TRD) and Texture Browsing Descriptor (TBD). We found that layouts generated using TRD are more suitable for browsing, and $L1$ and EMD are equally valid choices for calculating the distance between any two TRD feature vectors. Because $L1$ is more efficient than EMD, $L1$ should be used instead of EMD if the speed of generating a layout is a concern.

By improving the efficiency and effectiveness of colour-based CBIR, formulating a powerful image browsing framework (eyeMap), then developing fully functional programs based on eyeMap, and then evaluating eyeMap by comparing the developed programs against traditional methods, this thesis, therefore, has brought CBIR systems one step closer to real world applications.

9.2 Potential Future Research Directions

Potential future research directions as a result of the studies conducted in this thesis fall into two main categories: on eyeMap itself and on the evaluation of eyeMap in a different environment. The specification of eyeMap is complete and any implementation from this specification will result in a fully functional system; nonetheless, additional features for the retrieval engine such as relevance feedback, efficient search techniques and data structures [23, 24, 27, 45, 68, 134, 135] will make eyeMap even more useful for expert users. The relevance feedback is useful not only for the retrieval engine but also for the browsing engine during the generation of layouts. The implementation of these additional features must be done with care to ensure the programs remain usable for the novice.

Colour-based and texture-based eyeMap are fully working systems and useful for browsing and retrieving general colour and texture images; however, the image searching task described in this thesis is simulated in that users were asked to find target images in a laboratory setting. It will be interesting to evaluate eyeMap in a field study which involves real users in their working environment. To date, field studies of image retrieval are only found in concept-based retrieval not content-based [6, 40]. Unlike content-based retrieval, concept-based retrieval uses only text annotation.

Bibliography

- [1] Gimp (gnu image manipulation program). <http://www.gimp.org/>, 2004.
- [2] Imagemagick. www.imagemagick.org, Aug 2004.
- [3] S Abbasi, F Mokhtarian, and J Kittler. Enhancing CSS-based shape retrieval for objects with shallow concavities. In *Image and Vision Computing*, volume 18, pages 199-211, Feb 2000.
- [4] K Arbter. Affine-invariant fourier descriptors. In J C Simon, editor, *From Pixels to Features*, 1989.
- [5] K Arbter, W E Snyder, H Burkhardt, and G Hirzinger. Application of affine-invariant fourier descriptors to recognition of 3-d objects. *IEEE Trans. PAMI*, 12(7):640-647, 1990.
- [6] L H Armitage and P G Enser. Analysis of user needs in image archives. *Journal of Information Science*, 23(4):287-299, 1997.
- [7] H Asada and M Brady. The curvature primal sketch. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):2 - 14, Jan 1986.
- [8] Sandrine Balbo and John Murphy. *Usability evaluation: an introduction to what, why and how to*. The University of Melbourne and Design4Use, Australia, 2004.
- [9] Wojciech Basalaj. *Proximity Visualisation of Abstract Data*. PhD thesis, Cambridge Uni Comp Lab, 2001.
- [10] S O Belkasim, M Shridhar, and M Ahmadi. Pattern recognition with moment invariants: a comparative study and new results. *Pattern Recognition*, 24(12):1117 - 1138, Dec 1991.

- [11] Stefano Berretti, Alberto Del Bimbo, and Pietro Pala. Retrieval by shape similarity with perceptual distance and effective indexing. *IEEE Transactions on Multimedia*, 2(4):225-239, Dec 2000.
- [12] E Binaghi, I Gagliardi, and R Schettini. Image retrieval using fuzzy evaluation of color similarity. *Intl Journal of Pattern Recognition & AI*, 1994.
- [13] Chad Carson, Serge Belongie, Hayit Greenspan, and Jitendra Malik. Region-based image querying. In *CVPR '97 Workshop on Content-Based Access of Image and Video Libraries*, 1997.
- [14] Chad Carson, Megan Thomas, Serge Belongie, Joseph M. Hellerstein, and Jitendra Malik. Blobworld: A system for region-based image indexing and retrieval. In *Third Int. Conf. on Visual Information Systems*. Springer-Verlag, Jun 1999.
- [15] Youssef Chahir and Liming Chen. Searching images on the basis of color homogeneous objects and their spatial relationship. *Journal of Visual Communication and Image Representation*, 11(1):302-326, 2000.
- [16] Anthony Chaston and Alan Kingstone. Time estimation: The effect of cortically mediated attention. *Brain and Cognition*, 55(2):286-289, Jul 2004.
- [17] C Chen and M Czerwinski. Empirical evaluation of information visualizations: An introduction. *International Journal of Human-Computer Studies*, 53:631-635, 2000.
- [18] Cheomei Chen, George Gagnaudakis, and Paul Rosin. Similarity-based image browsing. In *Proceedings of the 16th IFIP World Computer Congress. International Conference on Intelligent Information Processing*, pages 206-213. Beijing, China, Aug 2000.
- [19] Jau-Yuen Chen, C A Bouman, and J C Dalton. Hierarchical browsing and search of large image databases. *Image Processing, IEEE Transactions on*, 9(3):442-455, Mar 2000.

- [20] Kan-Min Chen and Shu-Yuan Chen. Color texture segmentation using feature distributions. *Pattern Recognition Letters*, 23(7):755-894, May 2002.
- [21] L Chen, G Lu, and D S Zhang. Effects of different gabor filter parameters on image retrieval by texture. In *10th MMM*, pages 273-278, Australia, Jan 2004.
- [22] Yixin Chen and J Z Wang. A region-based fuzzy feature matching approach to content-based image retrieval. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(9):1252-1267, Sep 2002.
- [23] Tat-Seng Chua, Wai-Chee Low, and Chun-Xin Chu. Relevance feedback techniques for color-based image retrieval. In *Proceedings of Multi-Media Modelling 1998*, pages 24-31, Lausanne, Switzerland, Oct 1998. IEEE Press.
- [24] G Ciocca and R Schettini. A relevance feedback mechanism for content-based image retrieval. *Information Processing & Management*, 35(5):589-721, Sep 1999.
- [25] J D Cohen. Drawing graphs to convey proximity: An incremental arrangement method. *ACM Transactions on CHI*, 4:197-229, Sep 1997.
- [26] T A Combs and B Bederson. Does zooming improve image browsing? In *Proc. of the 4th ACM Intl. Conf. on Digital Libraries*, 1999.
- [27] I J Cox, M L Miller, S M Omohundro, and P N Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *Proc. Int. Conf. on Pattern Recognition*, pages 361-369, Vienna, Austria, Aug 1996.
- [28] Trevor F Cox and Michael A A Cox. *Multidimensional Scaling*. Chapman and Hall / CRC, 2001.
- [29] Y Deng, B S Manjunath, C Kenney, M S Moore, and H Shin. An efficient color representation for image retrieval. *IEEE Trans. on Image Processing*, 2001.
- [30] Jay L Devore. *Probability and Statistics for Engineering and the Sciences*. Brooks/Cole, 1987.

- [31] A S Dudani, K J Breeding, and R B McGhee. Aircraft identification by moment invariants. *IEEE Trans. on Computers*, C-26(1):39-45, Jan 1977.
- [32] Mark D Fairchild. Refinement of the lab colour space. *Color Research and Application*, 21(5):338-346, Oct 1996.
- [33] Mark D Fairchild. *Color Appearance Models*. Addison-Wesley, 1997.
- [34] Christos Faloutsos, Ron Barber, Myron Flickner, Jim Hafner, Wayne Niblack, Dragutin Petkovic, and William Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231-262, 1994.
- [35] Christos Faloutsos and King-Ip Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In Michael J. Carey and Donovan A. Schneider, editors, *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163-174, San Jose, California, 1995.
- [36] Julien Fauqueur and Nozha Boujemaa. Logical query composition from local visual feature thesaurus. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 187-194, Rennes, France, Sep 2003.
- [37] G D Finlayson and G Y Tian. *Texture Analysis in Machine Vision*, volume 40, pages 101-111. World Scientific, 2000.
- [38] James D Foley, Andries van Dam, Steven K Feiner, John F Hughes, and Richard L Phillips. *Introduction to Computer Graphics*. Addison Wesley, 1994.
- [39] J M Francos, A Z Meiri, and B Porat. A unified texture model based on a 2-d world-like decomposition. *Signal Processing, IEEE Transactions on*, 41(8):2665-2678, Aug 1993.
- [40] C O Frost, B Taylor, A Noakes, S Markel, D Torres, and K M Drabenstott. Browse & search patterns in a digital image database. *Info. Retrieval*, 1(4):287-313, 2000.

- [41] G W Furnas. Generalized fisheye views. In *Proceedings of the CHI'86 Conference on Human Factors in Computing Systems and Graphic Interfaces*, pages 16-23, Boston, May 1986. Addison-Wesley.
- [42] George W Furnas, Thomas K Landauer, Louis M Gomez, and Susan T Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964-971, 1987.
- [43] GIFT. GIFT : The gnu image-finding tool. <http://www.gnu.org/software/gift>, .
- [44] Neil J Gunther and Giordano Beretta. Full image: A benchmark for image retrieval using distributed systems over the internet: Birds-i. Technical report, HP, 2000.
- [45] Guo-Dong Guo, Anil K Jain, Wei-Ying Ma, and Hong-Jiang Zhang. Learning similarity measure for natural image with relevance feedback. In *CVPR*, volume I, pages 731-736, 2001.
- [46] Atsushi Hiroike, Yoshinori Matsushita, Akihiro Sugimoto, and Yasuhide Mori. Visualization of information spaces to retrieve and browse image data. In *Third International Conference On visual Information Systems (VISUAL'99)*, number 1614 in *Lecture Notes in Computer Science*, The Amsterdam, Netherlands, Jun 1999.
- [47] J Huang, S Kumar, M Mitra, W Zhu, and R Zabih. Image indexing using color correlograms. In *IEEE CVPR*, San Juan, Puerto Rico, Jun 1997.
- [48] J Huang, S R Kumar, M Mitra, and W J Zhu. Spatial color indexing and applications. In *Proc. of 8th Intl. Conference on Computer Vision*, 1998.
- [49] Jing Huang. *Spatial Color Indexing and Applications*. PhD thesis, Cornell University, 1998.
- [50] F Idris and S Panchanathan. Algorithms for indexing of compressed images. In *Proceedings of International Conference on Visual Information Systems*, 1996.

- [51] J S Jin, R Kurniawati, G Xu, and X Bai. Using browsing to improve CBIR. *Jnl of Vis Comp & Img Rep*, 12(2):123-135, Jun 2001.
- [52] Mohan S Kankanhalli, Babu M Mehtre, and Kang Wu Jian. Cluster-based color matching for image retrieval. *Pattern Recognition*, 29(4), 1996.
- [53] I Kompatsiaris, E Triantafillou, and M G Strintzis. Region-based color image indexing and retrieval. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 658-661, 2001.
- [54] J B Kruskal and M Wish. *Multidimensional Scaling*. Sage Publications, Beverly Hills, Calif., 1978.
- [55] Wei Lai and Peter Eades. Removing edge-node intersections in drawings of graphs. *Information Processing Letters*, 81(2):105-110, Jan 2002.
- [56] John Lamping, Ramana Rao, Peter Pirolli Ramana Rao, and Stuart K. Card. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI'95)*, pages 401-408, Denver, Colorado, United States, 1994. ACM Press.
- [57] Anton Leouski and James Allan. Evaluating a visual navigation system for a digital library. In *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries (ECDL'98)*, pages 535-554, 1998.
- [58] W K Leow and R Li. Adaptive binning and dissimilarity measure for image retrieval and classification. In *Proc. CVPR*, 2001.
- [59] Y K Leung. Human-computer interface techniques for map based diagrams. In G. Salvendy and M.J. Smith, editors, *Designing and Using Human-Computer Interfaces and Knowledge Based Systems*, pages 361-368, 1989.
- [60] Y K Leung and M D Apperley. A review and taxonomy of distorted-oriented presentation techniques. *ACM Transaction on Computer-Human Interaction*, 1(2):126-160, Jun 1994.

- [61] S Lim, L Chen, G Lu, and R Smith. Browsing texture image databases. In *11th International Conference on Multimedia Modelling*, 2005.
- [62] S Lim and G Lu. Effectiveness and efficiency of six colour spaces for content based image retrieval. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 215-222, Rennes, France, Sep 2003.
- [63] S Lim and G Lu. Improving colour autocorrelogram for CBIR. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 63-70, Rennes, France, Sep 2003.
- [64] S Lim and G Lu. Spatial statistics for content based image retrieval. In *Intl. Conf. on Information Technology: Coding and Computing (ITCC 2003)*, pages 155-159, Nevada, USA, Apr 2003.
- [65] S Lim, R Smith, and G Lu. i-map: An interactive visualisation and navigation system of an image database for finding a sample image to initiate a visual query. In *OZCHI'04*, 2004.
- [66] Fang Liu and Rosalind W Picard. Periodicity, directionality, and randomness: World features for image modeling and retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):722-733, Jul 1996.
- [67] G Lu and J Phillips. Using perceptually weighted histograms for colour-based image retrieval. In *Fourth International Conference on Signal Processing*, Beijing, Oct 1998.
- [68] Guojun Lu. Techniques and data structures for efficient multimedia retrieval based on similarity. *IEEE Transactions on Multimedia*, 4(3):372-384, 2002.
- [69] Guojun Lu and Atul Sajjanhar. Region-based shape representation and similarity measure suitable for content-based image retrieval. *ACM Multimedia System Journal*, 7(2):165-174, 1999.
- [70] Jock D Mackinlay, George G Robertson, and Stuart K Card. The perspective wall: Detail and context smoothly integrated. In *Proceedings of the SIGCHI*

- conference on Human factors in computing systems (CHI'91), pages 173-176, New Orleans, Louisiana, United States, Apr 1991.
- [71] B S Manjunath, P Wu, S Newsam, and H D Shin. A texture descriptor for browsing and similarity retrieval. *Journal of Signal Processing: Image Communication*, 16(1-2):33-43, Sep 2000.
- [72] B.S. Manjunath, P. Salembier, and T. Sikora, editors. *Introduction to MPEG-7. Multimedia Content Description Interface*. John Wiley and Sons Ltd, 2002.
- [73] Ranchandra Manthalkar, P K Biswas, and B N Chatterji. Rotation and scale invariant texture classification using gabor wavelets. In *The 2nd International workshop on texture analysis and synthesis*, Jun 2002.
- [74] M Markkula and E Sormunen. End-user searching challenges indexing practices in the digital newspaper photo archive. *Info. Retrieval*, 1:259-285, 2000.
- [75] Kim Marriott, Peter Stuckey, Vincent Tam, and Weiqing He. Removing node overlapping in graph layout using constrained optimization. *Constraints*, 8(2):143-171, Apr 2003.
- [76] E Mathias and A Conci. Comparing the influence of color spaces & metrics in CBIR. In *SIBGRAPI'98 Image Processing & Vision IMPA*, 1998.
- [77] T Meiers, T Sikora, and I Keller. 3d browsing environment for mpeg-7 image databases. In Minerva M Yeung, Chung-Sheng Li, and Rainer W Lienhart, editors, *Storage and Retrieval for Media Databases 2002, Proceedings of SPIE*, volume 4676, pages 324-335, Jan 2002.
- [78] K Messer and J Kittler. A region-based image database system using colour and texture. *Pattern Recognition Letter*, 20(11-13):1323-1330, Nov 1999.
- [79] Majid Mirmehdi and Radhakrishnan Perissamy. CBIR with perceptual region features. In T Cootes and C Taylor, editors, *Proc. of the 12th British Machine Vision Conf.*, pages 511-520, Sep 2001.

- [80] Majid Mirmehdi and Radhakrishnan Perissamy. Perceptual image indexing and retrieval. *Journal of Visual Communication and Image Representation*, 13(4):460-475, Dec 2002.
- [81] Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages and Computing*, 6(2):183-210, Jun 1995.
- [82] Baback Moghaddam, Qi Tian, and Thomas S Huang. Spatial visualization for content-based image retrieval. In *Proc. of International Conference on Multimedia and Expo (ICME'01)*, 2001.
- [83] Baback Moghaddam, Qi Tian, Neal Lesh, Chia Shen, and Thomas Huang. Visualization and layout for personal photo libraries. Technical report, Mitsubishi Electric Research Laboratories, 2001.
- [84] Baback Moghaddam, Qi Tian, Neil Lesh, Chia Shen, and Thomas S Huang. PDH: A human-centric interface for image libraries. In *ICME*, Lausanne, Switzerland, Aug 2002.
- [85] D Mohamed, G Sulong, and S S Ipson. Trademark matching using invariant moments. In *Proc. 2nd Asian Conference on Computer Vision*, volume I, pages 439-444, Singapore, Dec 1995.
- [86] F Mokhtarian, S Abbasi, and J Kittler. Efficient and robust retrieval by shape content through curvature scale space. In *Int. Workshop on Image Databases and Multimedia Search*, pages 35-42, Amsterdam, The Netherlands, 1996.
- [87] F Mokhtarian, S Abbasi, and J Kittler. Robust and efficient shape indexing through curvature scale space. In *Proc. British Machine Vision Conference*, pages 53-62, Edinburgh, UK, 1996.
- [88] A Morrison, G Ross, and M Chalmers. Fast multidimensional scaling through sampling, springs & interpolation. *Info Vis*, 2(1):68-77, Mar 2003.

- [89] Henning Müller, Wolfgang Müller, Stéphane Marchand-Maillet, David Squire, and Thierry Pun. Automated benchmarking in content-based image retrieval. In *Proceedings of the 3rd Intl Workshop on Multimedia Information Retrieval (in conjunction with ACM Multimedia 2001)*, Ottawa, Canada, 2001.
- [90] Henning Müller, Wolfgang Müller, David Squire, Stéphane Marchand-Maillet, and Thierry Pun. MRML: A communication protocol for content-based image retrieval. In *4th Intl Conf On Visual Information Systems (VISual 2000)*, Lyon, France, Nov 2000.
- [91] Henning Müller, Wolfgang Müller, David McG. Squire, Stéphane Marchand-Maillet, and Thierry Pun. Performance evaluation in content-based image retrieval: Overview and proposals. *Pattern Recognition Letters*, 22(5):593-601, 2001.
- [92] Henning Müller, Wolfgang Müller, David McG. Squire, Zoran Pečenović, Stéphane Marchand-Maillet, and Thierry Pun. An open framework for distributed multimedia retrieval. In *Recherche d'Informations Assistée par Ordinateur (RIAO'2000) Computer-Assisted Information Retrieval*, pages 701-712, Paris, France, Apr 2000.
- [93] Wolfgang Müller, Zoran Pecenovici, Henning Müller, Stephane Marchand-Maillet, Thierry Pun, David Squire, Arjen P De Vries, and Christoph Giess. MRML: An extensible communication protocol for interoperability and benchmarking of multimedia information retrieval systems. In *SPIE Photonics East - Voice, Video, and Data Communications*, Boston, MA, USA, Nov 2000.
- [94] R S Nickerson. Short term memory for complex meaningful configurations: A demonstration of capacity. *Canadian Journal of Psychology*, 19(2):155-160, 1965.
- [95] Jakob Nielsen. *Usability Engineering*. Morgan Kaufmann, 1993.
- [96] Chris North and Ben Shneiderman. Snap-together visualization: can users construct and operate coordinated visualizations? *International Journal of Human-Computer Studies*, 53(5):715-739, Nov 2000.

- [97] Jens-Rainer Ohm, Leszek Cieplinski, Heon J Kim, Santhana Krishnamachari, B S Manjunath, Dean S Messing, and Akio Yamada. *Introduction to MPEG-7. Multimedia Content Description Interface*, chapter 13, pages 187-228. Wiley, 13, 2002.
- [98] J R Parker. *Algorithms for Image Processing and Computer Vision*. John Wiley and Sons, Inc., 1997.
- [99] Greg Pass and Ramin Zabih. Histogram refinement for content-based image retrieval. In *IEEE Workshop on Applications of Computer Vision*, Sarasota, Florida, Dec 1996.
- [100] Greg Pass and Ramin Zabih. Comparing images using joint histograms. *Multimedia Systems*, 7, 1999.
- [101] Greg Pass, Ramin Zabih, and J Miller. Comparing images using color coherent vectors. In *4th ACM Multimedia Conference*, Nov 1996.
- [102] Zoran Pečenović, Minh N. Do, Martin Vetterli, and Pearl Pu. Integrated browsing and searching on large image collections. In Robert Laurini, editor, *Fourth International Conference on Visual Information Systems (VISual 2000)*, volume 1929 of *LNCS*, pages 279-289, Nov 2000.
- [103] Richard J Prokop and Anthony P Reeves. A survey of moment-based techniques for unoccluded object representation and recognition. *CVGIP: Graphical Models and Image Processing*, 54(5):438 - 460, Sep 1992.
- [104] J Puzicha, T Hofmann, and J Buhmann. Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. In *Proc. of the IEEE Intl Conf on Comp Vis an Pattern Recognition*, pages 267-272, San Juan, 1997.
- [105] J Puzicha, Y Rubner, C Tomasi, and J Buhmann. Empirical evaluation of dissimilarity measures for color and texture. In *Proceedings the IEEE International Conference on Computer Vision (ICCV-1999)*, pages 1165-1173, 1999.

- [106] G Qiu. Image indexing using a coloured pattern appearance model. In *Storage and Retrieval for Media Databases*, San Jose, CA, USA, Jan 2001.
- [107] G Qiu and K-M Lam. Spectrally layered color indexing. In *CIVR2002, The Challenge of Image and Video Retrieval, Lecture Notes in Computer Science*. Springer Verlag, 2002.
- [108] A Rao, R Srihari, and Z Zhang. Spatial color histograms for content-based image retrieval. In *11th IEEE Intl Conf on Tools With AI*, 1999.
- [109] Ramana Rao and Stuart K Card. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI'94)*, pages 318-322. Boston, Massachusetts, United States, 1994. ACM Press.
- [110] K C Ravishankar, B G Prasad, S K Gupta, and K K Biswas. Dominant color region based indexing for CBIR. In *Intl conf. on Image Analysis & Processing*, 1998.
- [111] Kirsten Ridsen, Mary P Czerwinski, Tamara Munzner, and Daniel B Cook. An initial examination of ease of use for 2d and 3d information visualizations of web content. *International Journal of Human-Computer Studies*, 53(5):695-714, Nov 2000.
- [112] George G Robertson and Jock D Mackinlay. The document lens. In *Proceedings of the 6th annual ACM symposium on User interface software and technology (UIST'93)*, pages 101-108, Atlanta, Georgia, United States, 1993.
- [113] Kerry Rodden. *Evaluating Similarity-Based Visualisations as Interfaces for Image Browsing*. PhD thesis, Newnham College, University of Cambridge, Cambridge, UK, 2001.
- [114] Kerry Rodden, Wojciech Basalaj, David Sinclair, and Kenneth Wood. A comparison of measures for visualising image similarity. In *The Challenge of Image*

- Retrieval (CIR 2000)*, Brighton, May 2000.
- [115] Kerry Rodden, Wojciech Basalaj, David Sinclair, and Kenneth Wood. Does organisation by similarity assist image browsing? In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 190-197, Seattle, Washington, United States, 2001. ACM Press.
- [116] Bernice E Rogowitz, Thomas Frese, John R Smith, Charles A Bouman, and Edward Kalin. Perceptual image similarity experiments. In *Proc. of SPIE/IS&T Conf. on Human Vision and Electronic Imaging III*, volume 3299, pages 576-590, San Jose, CA, Jan 1998.
- [117] Yossi Rubner. *Perceptual Metrics for Image Database Navigation*. PhD thesis, CS Stanford Uni., 1999.
- [118] Yossi Rubner and Carlo Tomasi. *Perceptual metrics for image database navigation*. Kluwer Academic, 2001.
- [119] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *Proc. IEEE ICPR*, 1998.
- [120] S.J. Sangwine and R.E.N Horne, editors. *The Colour Image Processing Handbook*. Chapman and Hall, 1998.
- [121] Simone Santini. *Exploratory Image Databases*. Academic Press, 2001.
- [122] Simone Santini and Ramesh Jain. Integrated browsing and querying for image databases. *Multimedia, IEEE*, 7(3):20-39, Jul 2000.
- [123] Manojit Sarkar and Marc H Brown. Graphical fisheye views of graphs. In Penny Bauersfeld, John Bennett, and Gene Lynch, editors, *Human Factors in Computing Systems, CHI'92 Conference Proceedings: Striking A Balance*, pages 83-91. ACM Press, May 1992.
- [124] R Schettini, G Ciocca, and S Zuffi. A survey of methods for colour image indexing and retrieval in image databases. *Color Imaging Science: Exploiting Digital Media*, 2001.

- [125] Michael W Schwarz. An experimental comparison of RGB, YIQ, LAB, HSV, and opponent color models. *ACM Transactions on Graphics*. 6(2):123-158, Apr 1987.
- [126] Ben Shneiderman. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison Wesley Longman, 1998.
- [127] David Sinclair. Image parsing for image retrieval from large image data bases: from coloured image to coloured regions. Technical report, AT&T Laboratories Cambridge, 24a Trumpington Street, Cambridge CB2 1QA, England, 1997.
- [128] Alvy Ray Smith. Color gamut transform pairs. In *SIGGRAPH'78- Proc. of the 5th annual conference on Computer Graphics and Interactive Techniques*, Aug 1978.
- [129] John R Smith and Shih-Fu Chang. Tools and techniques for color image retrieval. In *Symposium on Electronic Imaging: Science and Technology - Still Image Compression*, volume 2669, San Jose, CA, Jan 1996. IS&T/SPIE.
- [130] John R Smith and Apostov (Paul) Natsev. Spatial and feature normalization for content-based retrieval. In *ICME 2002*, 2002.
- [131] Ray Smith. *Distortion Oriented Displays for Demanding Applications*. PhD thesis, Gippsland School of Computing and IT, Monash University, 1997.
- [132] R Spence and M Apperley. Database navigation: an office environment for the professional. *Behaviour and Information Technology*, 1(1):43-54, 1982.
- [133] Bill Spitzak et al. Fast lightning tool kit (FLTK). <http://www.fltk.org/>, 2004.
- [134] David Squire, Wolfgang Müller, and Henning Müller. Relevance feedback and term weighting schemes for content-based image retrieval. In *3rd Intl. Conf. On Visual Information System (VISUAL'99)*, pages 545-556. Springer-Verlag, Jun 1999.
- [135] David Squire, Wolfgang Müller, Henning Müller, and Jilali Raki. Content-based query of image databases, inspirations from text retrieval: inverted files,

- frequency-based weights and relevance feedback.. In *The 11th Scandinavian Conference on Image Analysis (SCIA '99)*, pages 143-149, Kangerlussuaq, Greenland, Jun 1999.
- [136] L Standing. Learning 10,000 pictures. *Quarterly Journal of Experimental Psychology*, 25:207-222, 1973.
- [137] L Standing, J Conezio, and R N Haber. Perception and memory for pictures: Single trial learning of 2500 visual stimuli. *Psychonomic Science*, 19(2):73-74, 1970.
- [138] M Stricker and A Dinnai. Spectral covariance and fuzzy regions for image indexing. *Machine Vision and Applications*, 10:66-73, 1997.
- [139] Markus A Stricker and Markus Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases III(SPIE)*, pages 381-392, Bellingham, Wash., Feb 1995.
- [140] A G Sutcliffe, M Ennis, and J Hu. Evaluating the effectiveness of visual user interfaces for information retrieval. *International Journal of Human-Computer Studies*, 53(5):741-763, Nov 2000.
- [141] Michael J Swain and Dana H Ballard. Color indexing. *Intl Journal of Computer Vision*, 7(1):11-32, 1991.
- [142] Michael J Swain and Dana H Ballard. Indexing via color histograms. In *Proc. of the IEEE Conf. on Comp. Vision*, pages 390-393, 90.
- [143] R Tan, M Indrawan, and G Lu. Comparative studies on performance of three types of colour spaces. In *3rd Intl Conf on Info, Comm & Signal Processing*, 2001.
- [144] Yi Tao and William I Grosky. Delaunay triangulation for image object indexing: a novel method for shape representation. In *Proceedings of IS&T/SPIE's Symposium on Storage and Retrieval for Image and Video Databases VII*, pages 631-642, San Jose, California, Jan 1999.

- [145] Yi Tao and William I Grosky. Spatial color indexing using rotation, translation and scale invariant anglograms. *Multimedia Tools and Applications*, 15, 2001.
- [146] David S Taubman and Michael W Marcellin. *JPEG2000. Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, 2002.
- [147] M R Teague. Image analysis via the general theory of moments. *Journal Optical Society of America*, 70(8):920-930, 1980.
- [148] S Teng. *Image indexing and retrieval based on vector quantization*. PhD thesis, Gippsland School of Computing and IT, 2003.
- [149] S Teng and G Lu. Performance study of image retrieval based on vector quantization. In *Intl Conf on Intelligent Multimedia and Distance Education (ICIMADE)*, pages 76-80, North Dakota, USA, 2001.
- [150] Ricardo S Torres, Celmar G Silva, Claudia B Medeiros, and Heloisa V Rocha. Visual structures for image browsing. In *Proceedings of the 12th Intl Conf. on Information and Knowledge Management*, pages 49 - 55, New Orleans, LA, USA, 2003.
- [151] M Unser. Texture classification and segmentation using wavelet frames. *IEEE Trans. Image Processsing*, 4(11):1549-1560, Nov 1995.
- [152] Jana Urban, Joemon M Jose, and C C van Rijsbergen. An adaptive approach towards content-based image retrieval. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 119-126, Rennes, France, Sep 2003.
- [153] Egon van den Broek, L Vuurpijl, P Kisters, and J von Schmid. Content-based image retrieval: Color-selection exploited. In *Proceedings of the 3rd Dutch-Belgian Information Retrieval Workshop*, pages 37-46, Belgium - Leuven, Dec 2002.
- [154] Xia Wan and C C Jay Kuo. A new approach to image retrieval with hierarchical color clustering. *IEEE Transactions on Circuits and Systems for Video Technology*, 8(5):628-643, Sep 1998.

- [155] James Z Wang. *Integrated Region-based Image Retrieval*. Kluwer Academic Publishers, 2001.
- [156] Günther Wyszecki and W S Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. John Wiley and Sons, 2000.
- [157] Forest W Young. *Multidimensional Scaling-History, Theory & Applications*. Lawrence Erlbaum Assoc., 1987.
- [158] Forrest W Young. *Introduction to Multidimensional Scaling-History-Theory, Methods, and Applications*. Academic Press, 1981.
- [159] D S Zhang and G Lu. Shape based image retrieval using generic fourier descriptors. *Signal Processing: Image Communication*, 17(10):825-848, 2002.
- [160] D S Zhang, A Wong, M Indrawan, and G Lu. CBIR using Gabor texture features. In *Proc. 1st IEEE Pacific-Rim Conf. on MM (PCM'00)*, 2000.
- [161] Deng Sheng Zhang. *Image Retrieval Based on Shape*. PhD thesis, Gippsland School of Computing and Information Technology, Mar 2002.
- [162] X M Zhou, C H Ang, and T W Ling. Image retrieval based on object's orientation spatial relationship. *Pattern Recognition Letters*, 22(5):469-477, 2001.
- [163] Lei Zhu. *Keyblock: An Approach for Content Based Image Retrieval*. PhD thesis, University of New York, Buffalo, 2001.

Key Publications

- [1] S Lim and G Lu. Effectiveness and efficiency of six colour spaces for content based image retrieval. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 215-222, Rennes, France, Sep 2003.
- [2] S Lim and G Lu. Improving colour autocorrelogram for CBIR. In *3rd Intl Workshop on Content Based Multimedia Indexing*, pages 63-70, Rennes, France, Sep 2003.
- [3] S Lim and G Lu. Spatial statistics for content based image retrieval. In *Intl. Conf. on Information Technology: Coding and Computing (ITCC 2003)*, pages 155-159, Nevada, USA, Apr 2003.
- [4] S Lim, R Smith, and G Lu. i-map: An interactive visualisation and navigation system of an image database for finding a sample image to initiate a visual query. In *OZCHI'04*, 2004.
- [5] S Lim, L Chen, G Lu, and R Smith. Browsing texture image databases. In *11th International Conference on Multimedia Modelling*, 2005.

Supplementary Results for Colour Space Evaluations Presented in Chapter 3

B.1 PR Graphs in PCD for Cartesian Coordinate Colour Spaces

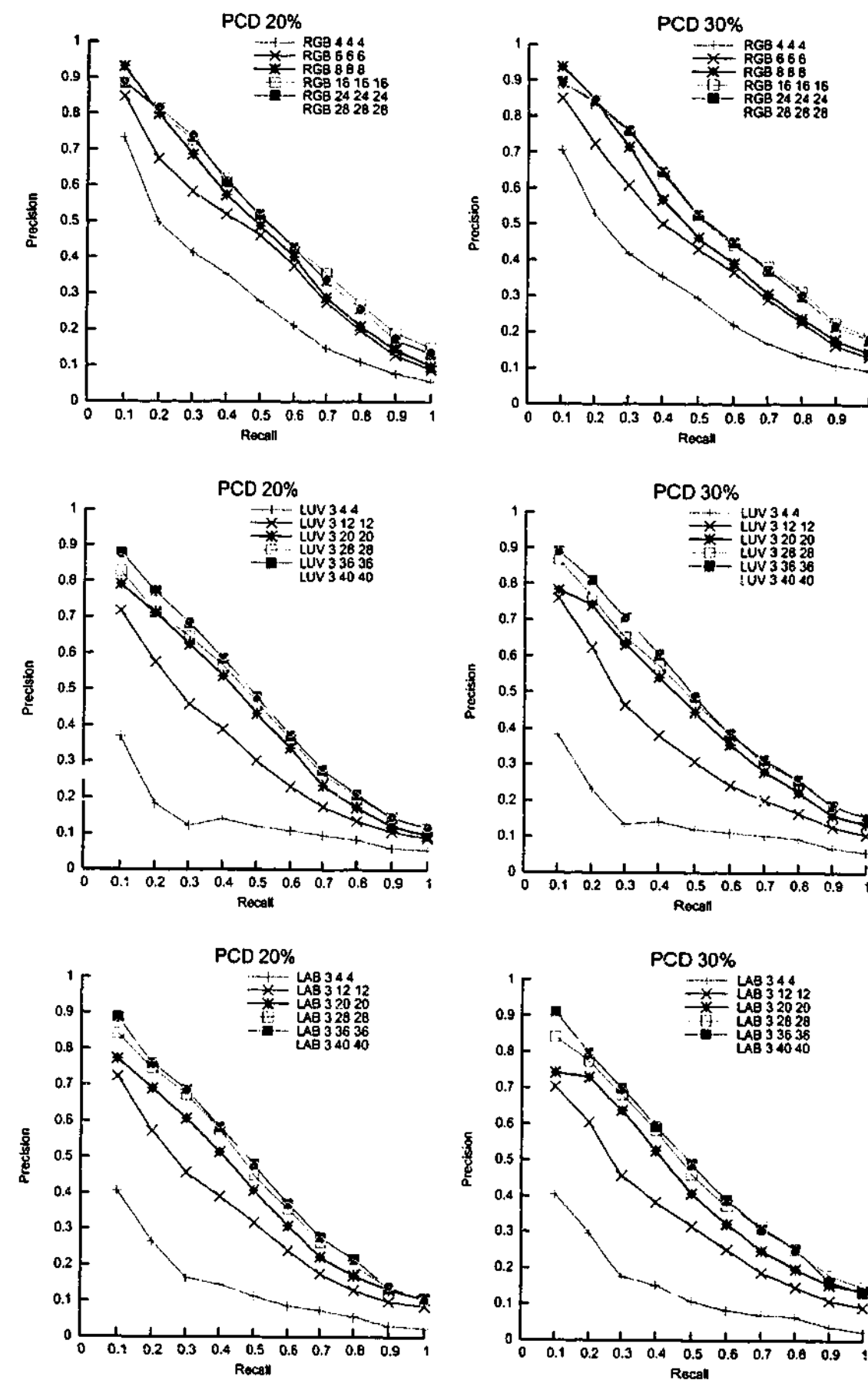


Figure B.1: PR graphs in PCD at 20%, 30%, 50% and 70% levels of agreement using RGB, LUV and LAB colour spaces at different numbers of quantisation intervals. ... continued on next page.

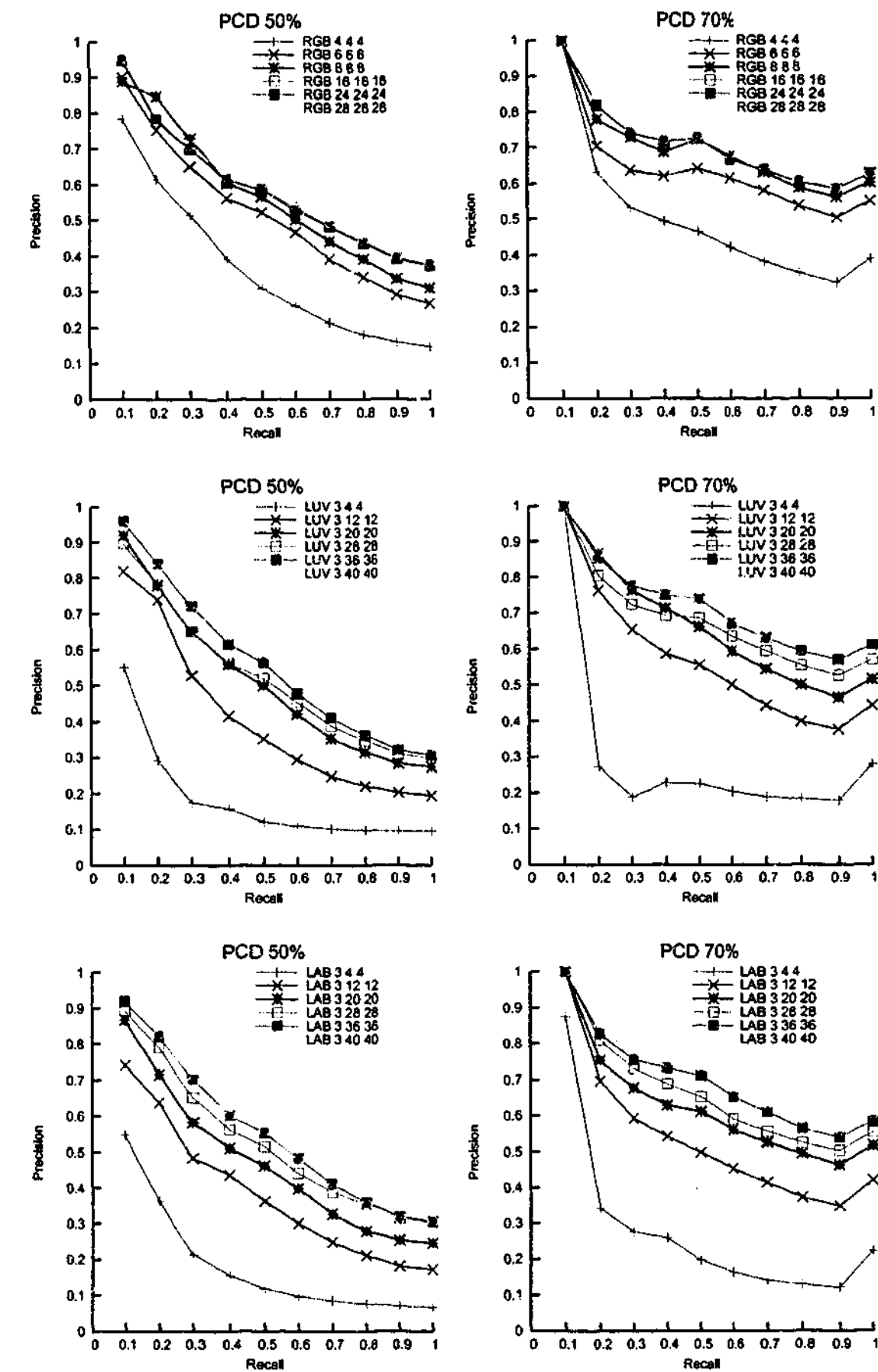
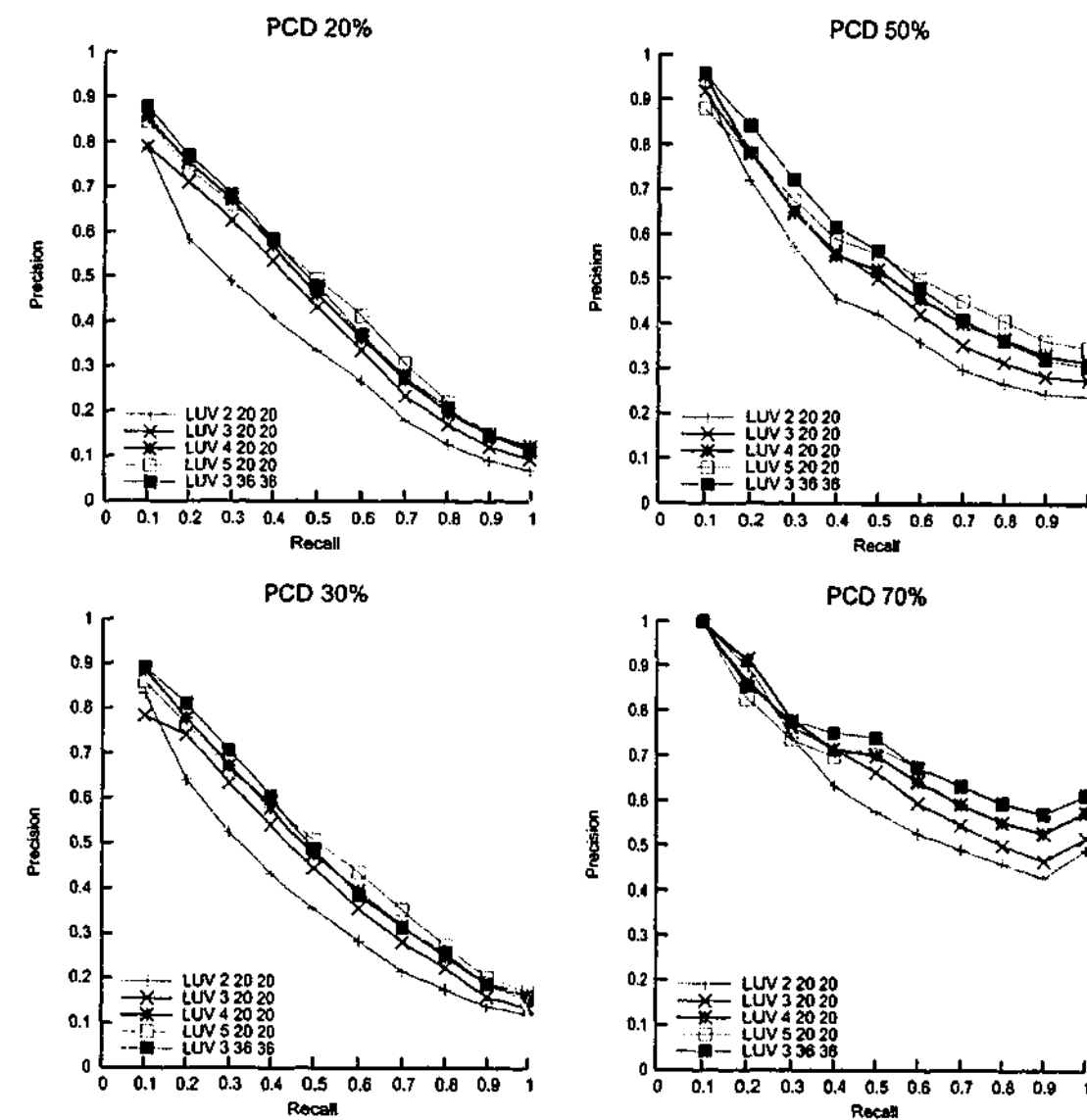
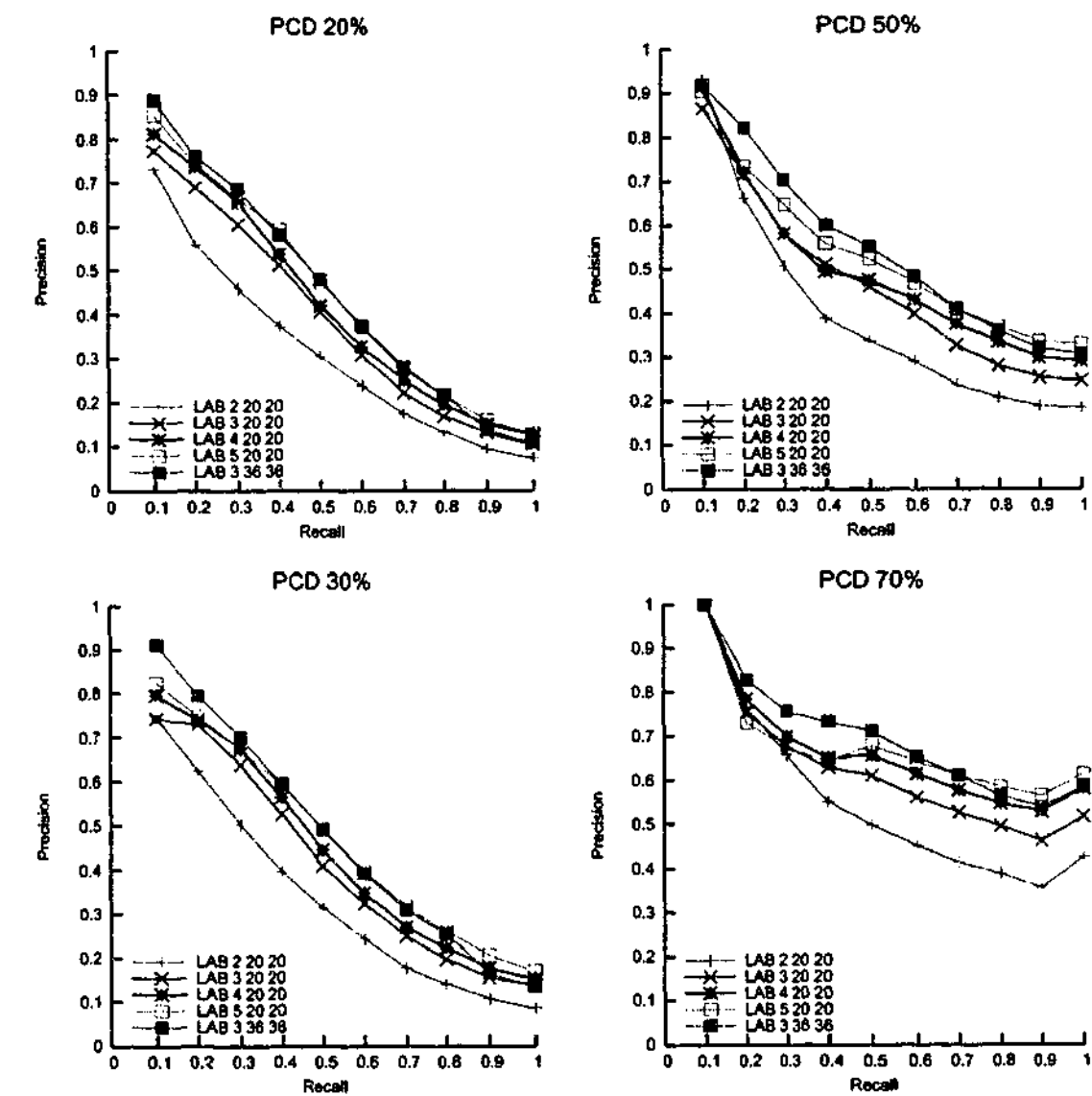


Figure B.1: ... continued from previous page.

Figure B.2: PR graphs of LUV colour space in PCD when the value of L is varied.Figure B.3: PR graphs of LAB colour space in PCD when the value of L is varied.

B.2 PR Graphs in PCD for Polar Coordinates Colour Spaces

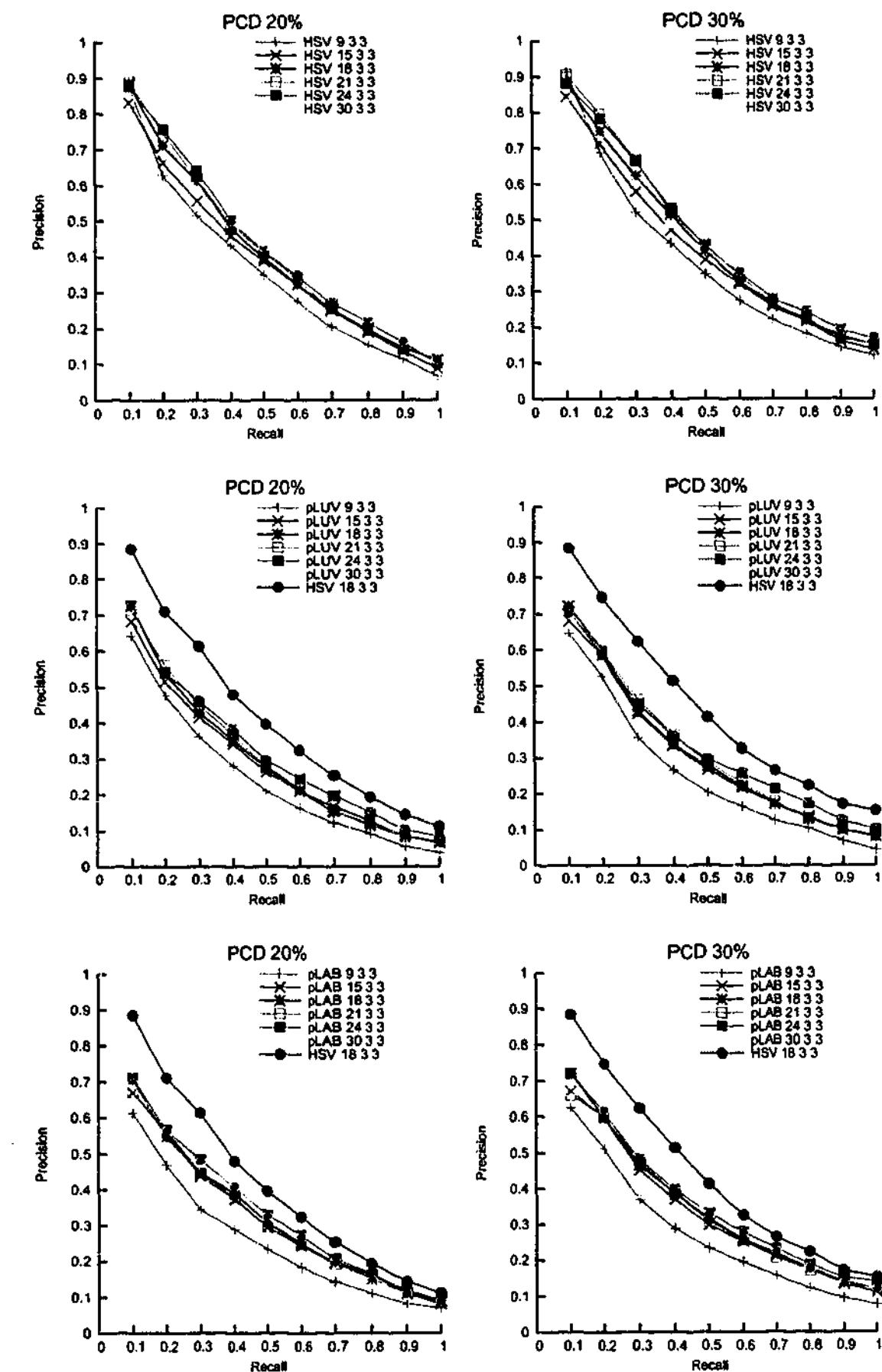


Figure B.4: PR graphs for PCD at 20%, 30%, 50% and 70% levels of agreement in HSV, LUV polar and LAB polar colour spaces at different numbers of quantisation intervals ... continued on next page.

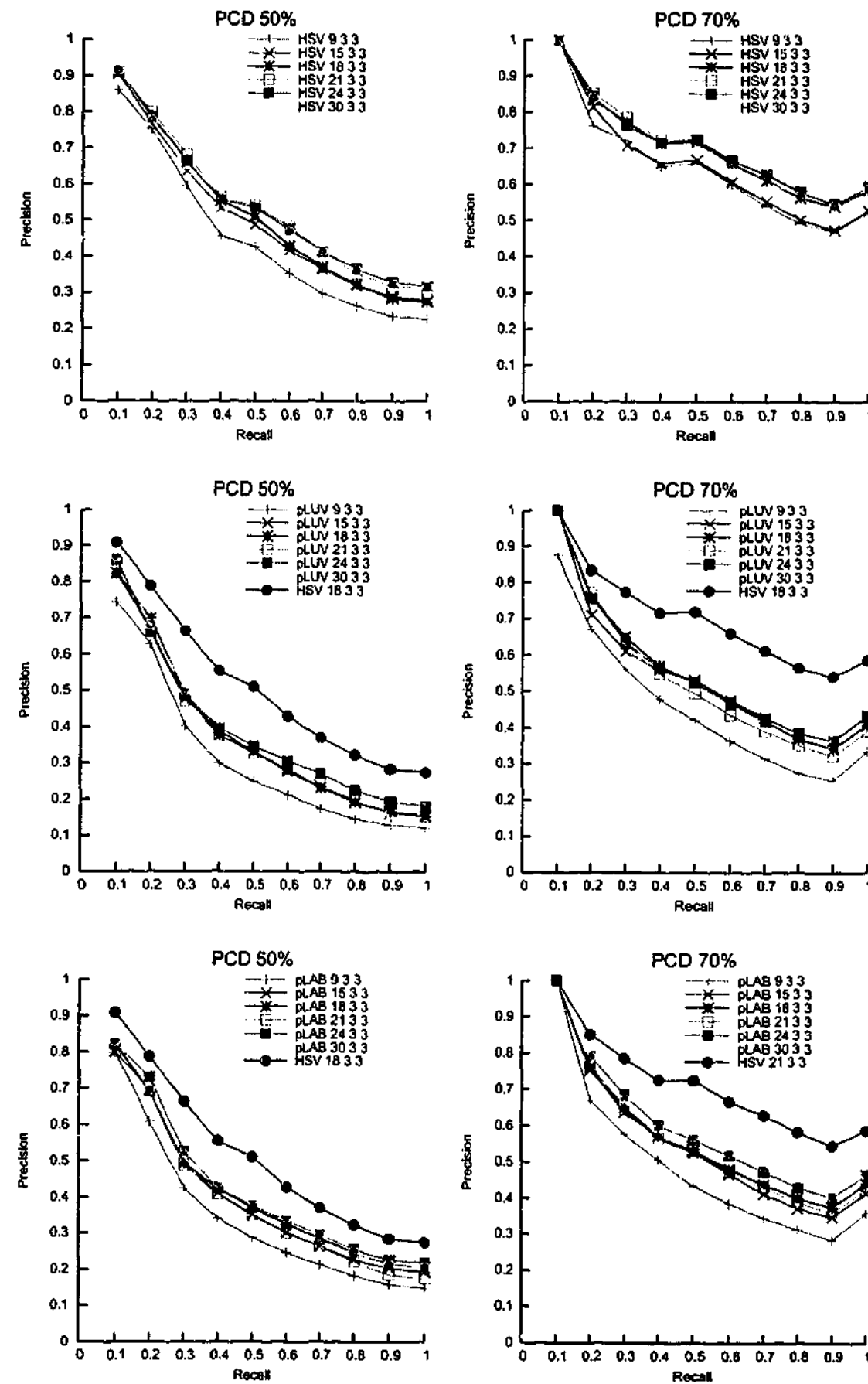


Figure B.4: ...continued from previous page.

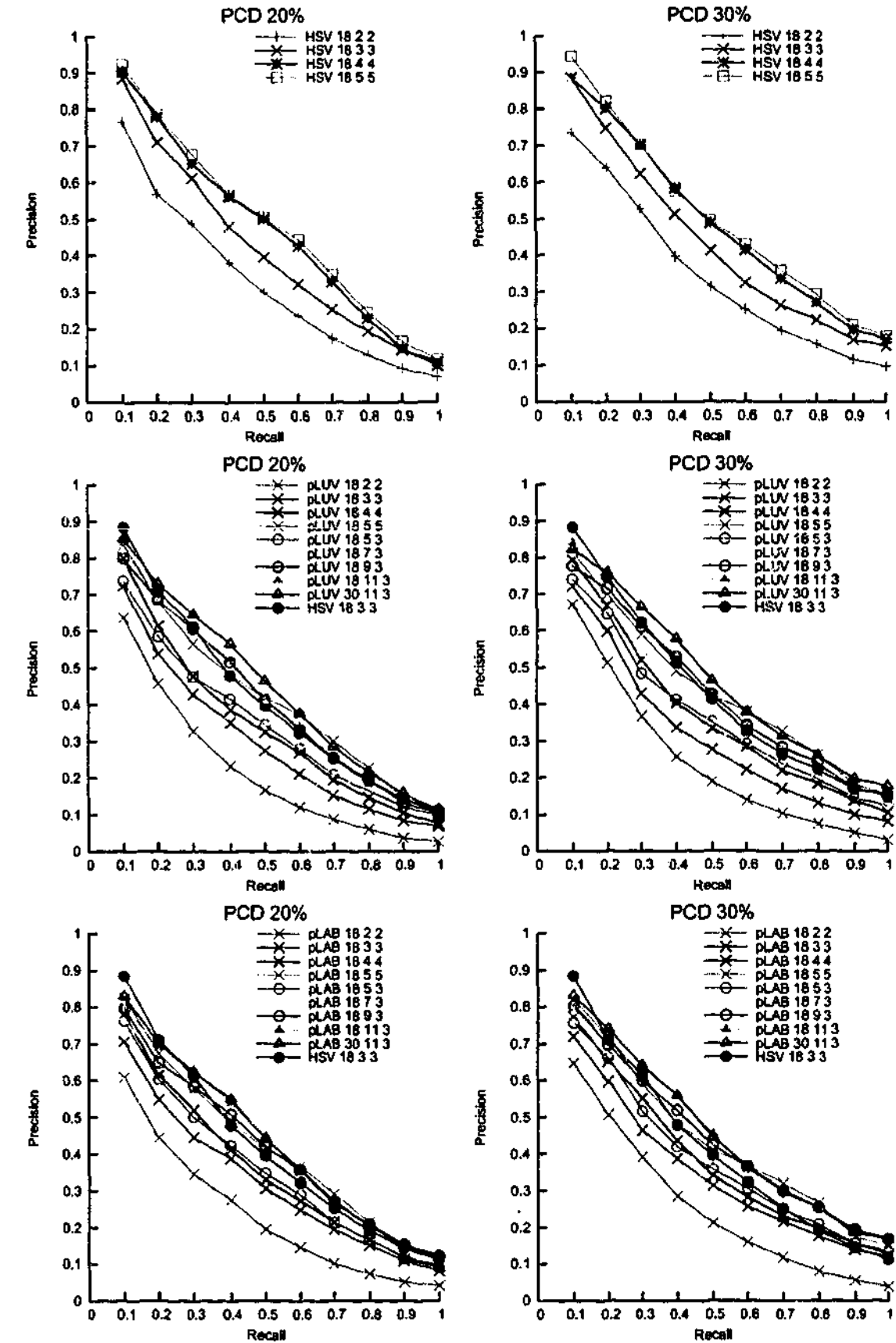


Figure B.5: PR graphs for PCD at 20%, 30%, 50% and 70% levels of agreement in HSV, LUV polar and LUV polar colour spaces at different numbers of quantisation intervals. ...continued on next page.

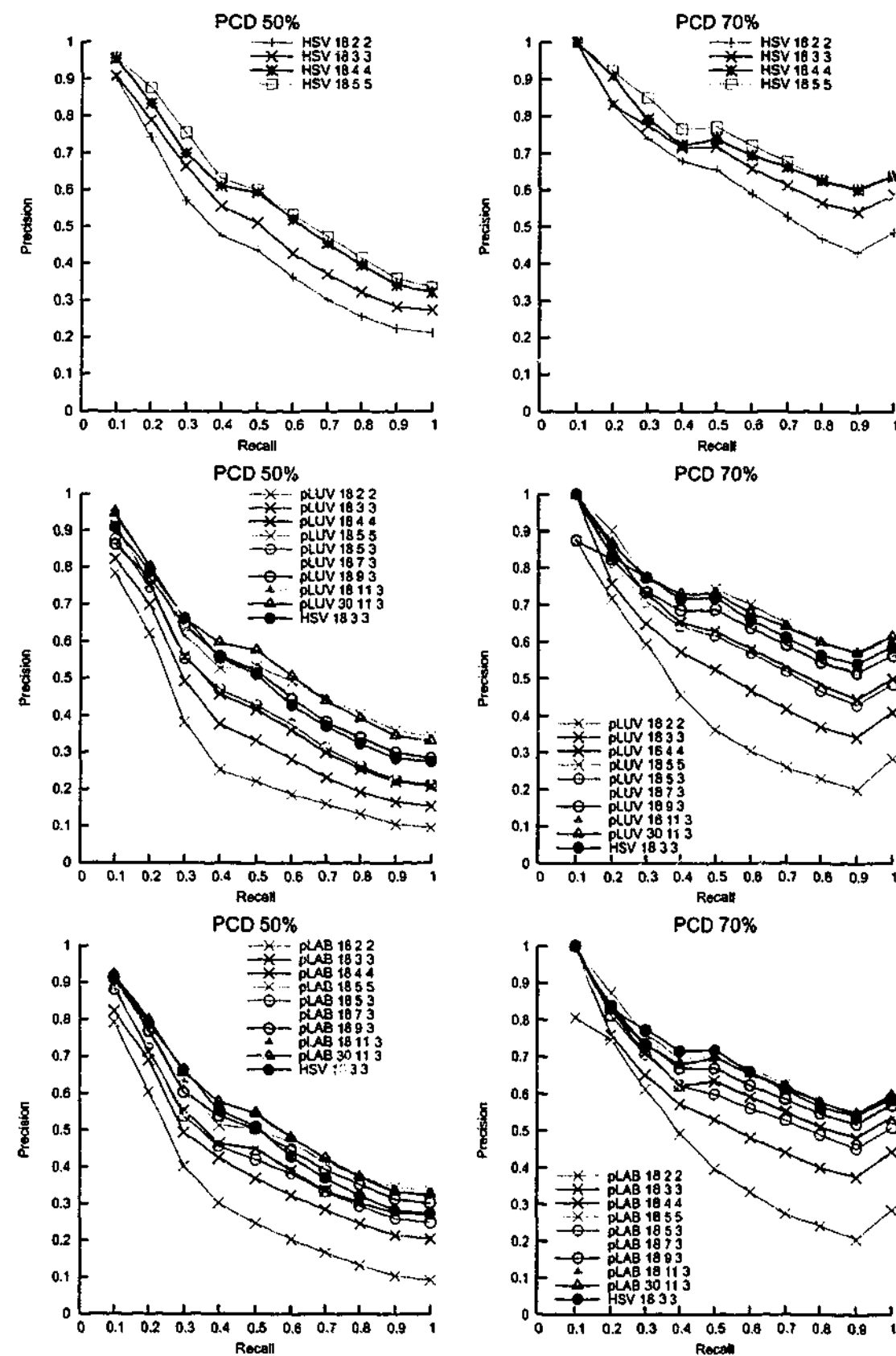


Figure B.5: ...continued from previous page.

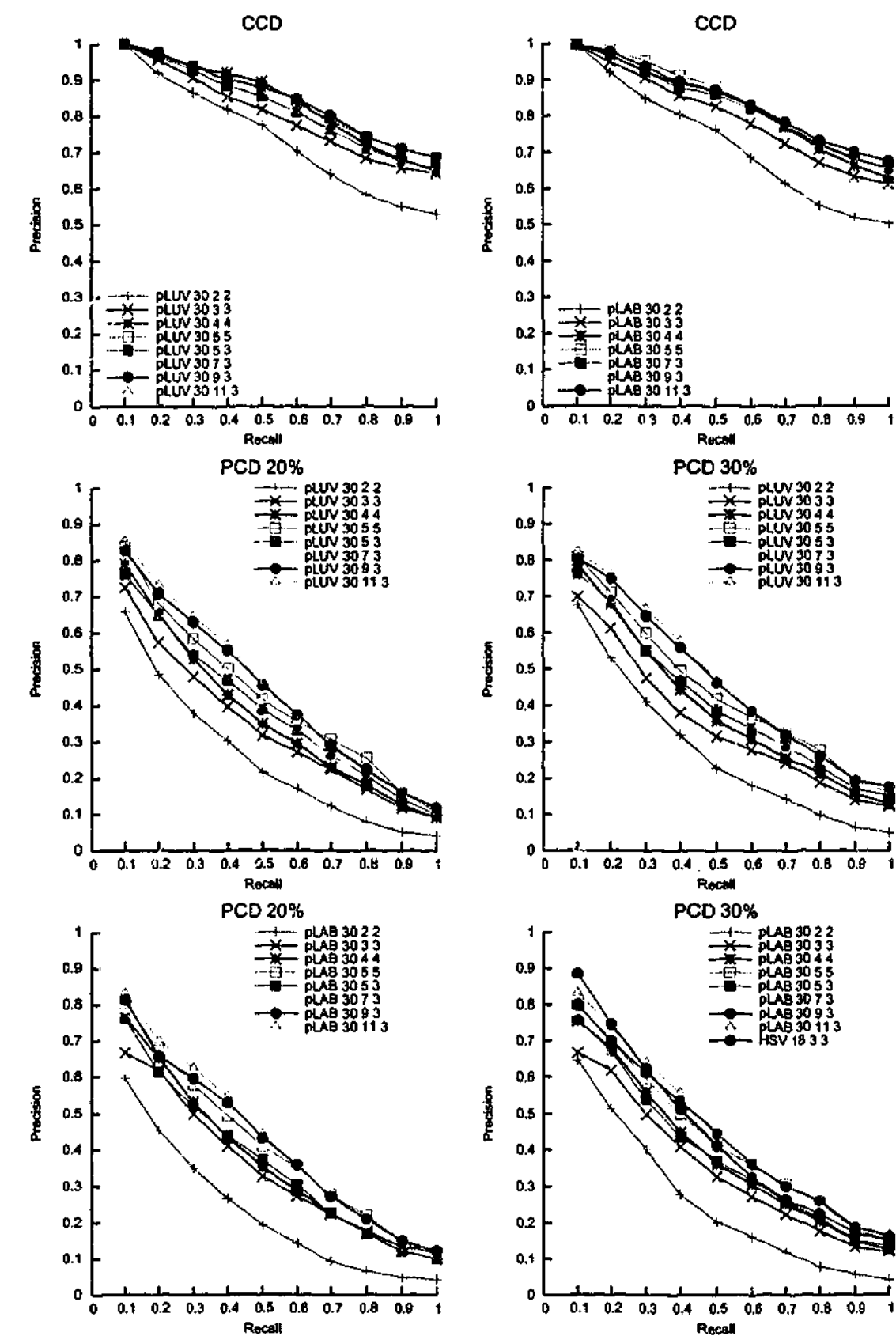


Figure B.6: PR graphs for CCD and PCD at 20%, 30%, 50% and 70% levels of agreement in LUV and LAB polar colour spaces. Similar to Fig. B.5 but the hue axis is quantised into 30 intervals. ...continued on next page.

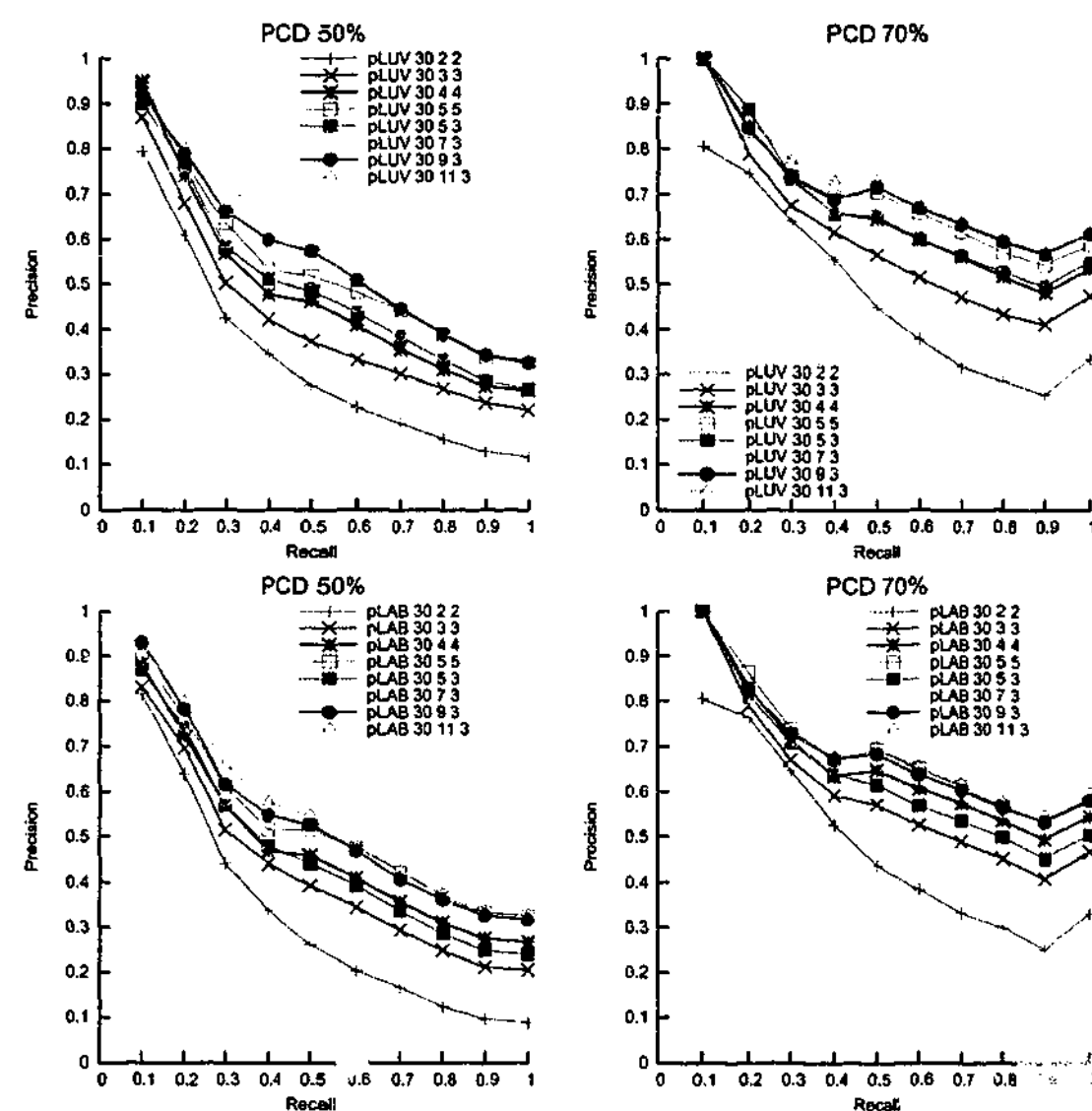


Figure B.6: ... continued from previous page.

B.3 PR Graphs in PCD for Most Effective Quantisation Options

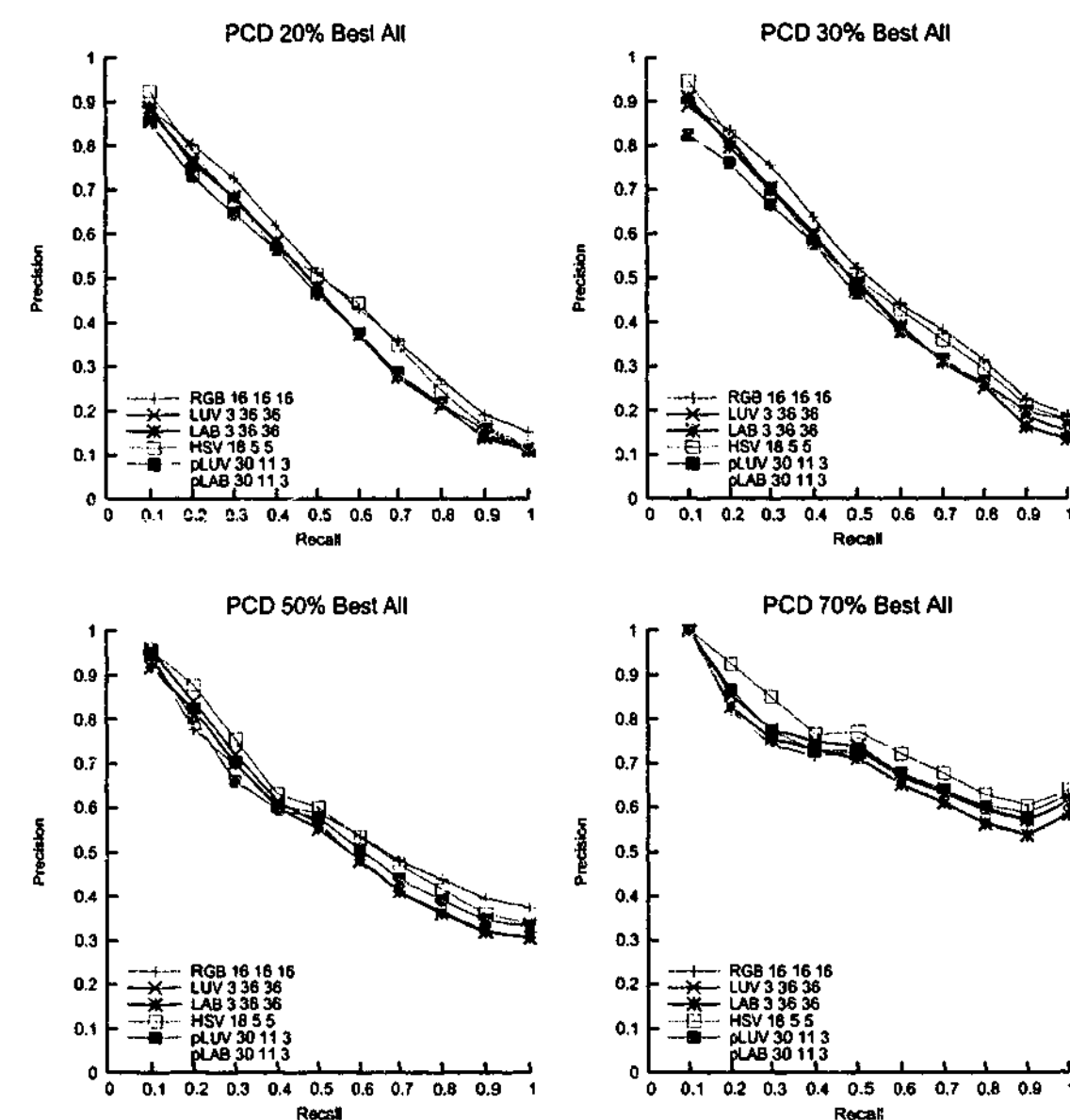


Figure B.7: PR graphs of RGB, LUV and LAB, HSV, LUV polar and LAB polar colour spaces at the highest effectiveness in PCD.

B.4 PR Graphs in PCD for Quantisation Options at Similar Sizes

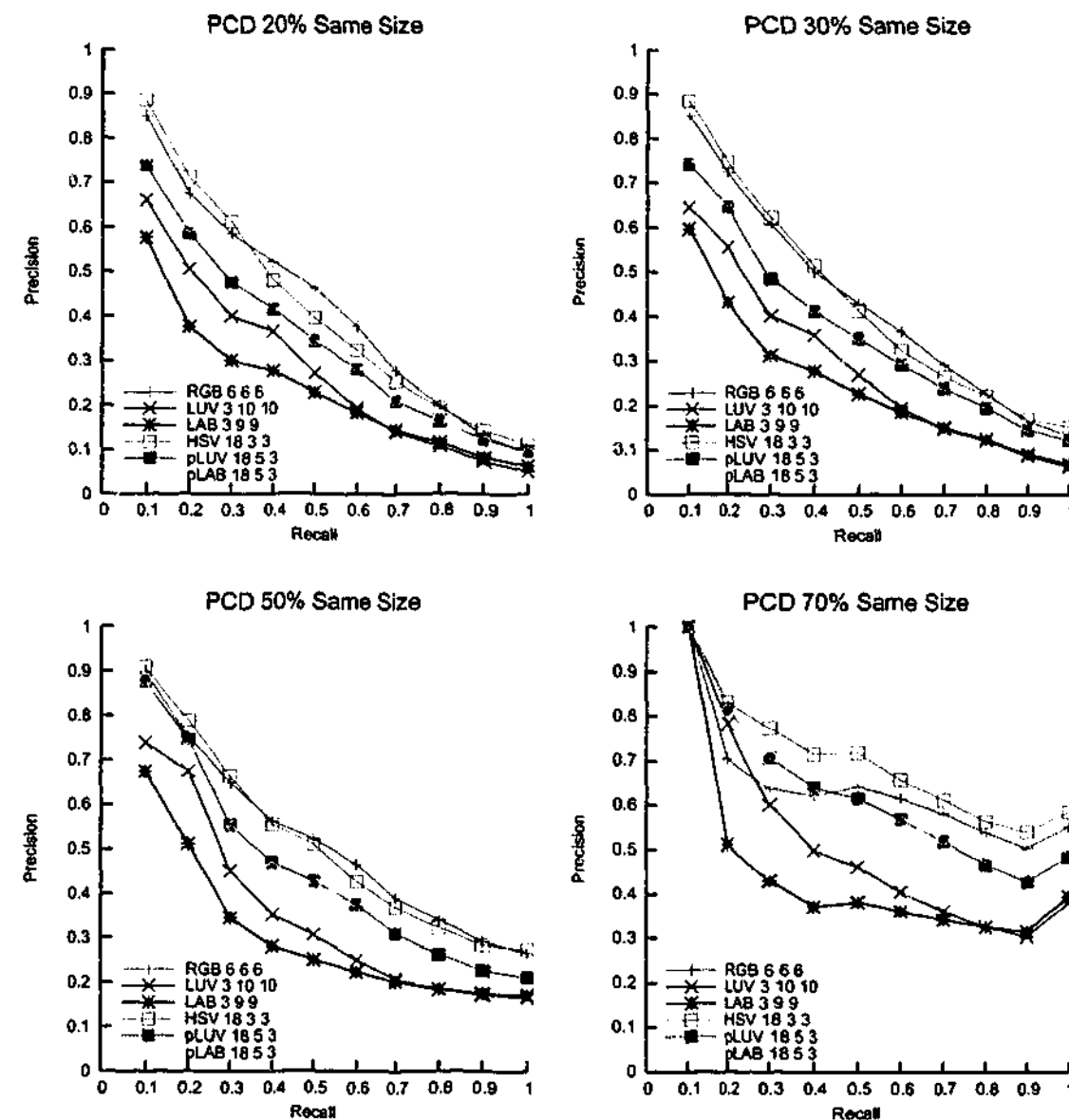


Figure B.8: PR graphs of RGB, LUV and LAB, HSV, LUV polar and LAB polar colour spaces with similar number of bins.

Supplementary Results for Colour Spatial Features Presented in Chapter 4

C.1 Autocorrelogram - Effect of Dissimilarity Metrics in CCD and PCD

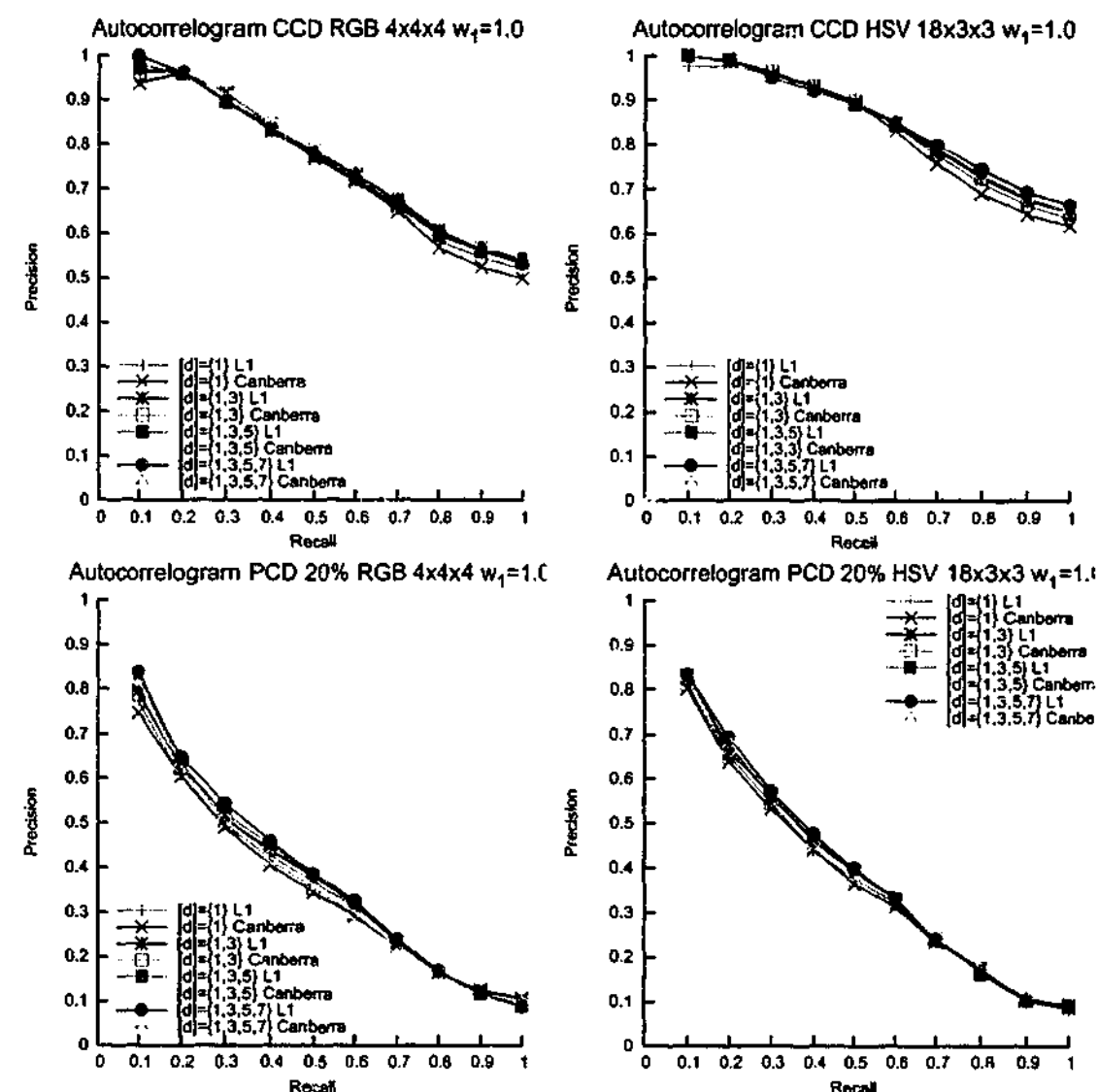


Figure C.1: PR graphs of autocorrelogram in CCD and PCD at different d with L1 and Canberra dissimilarity metrics using RGB and HSV colour spaces. ... continued on next page.

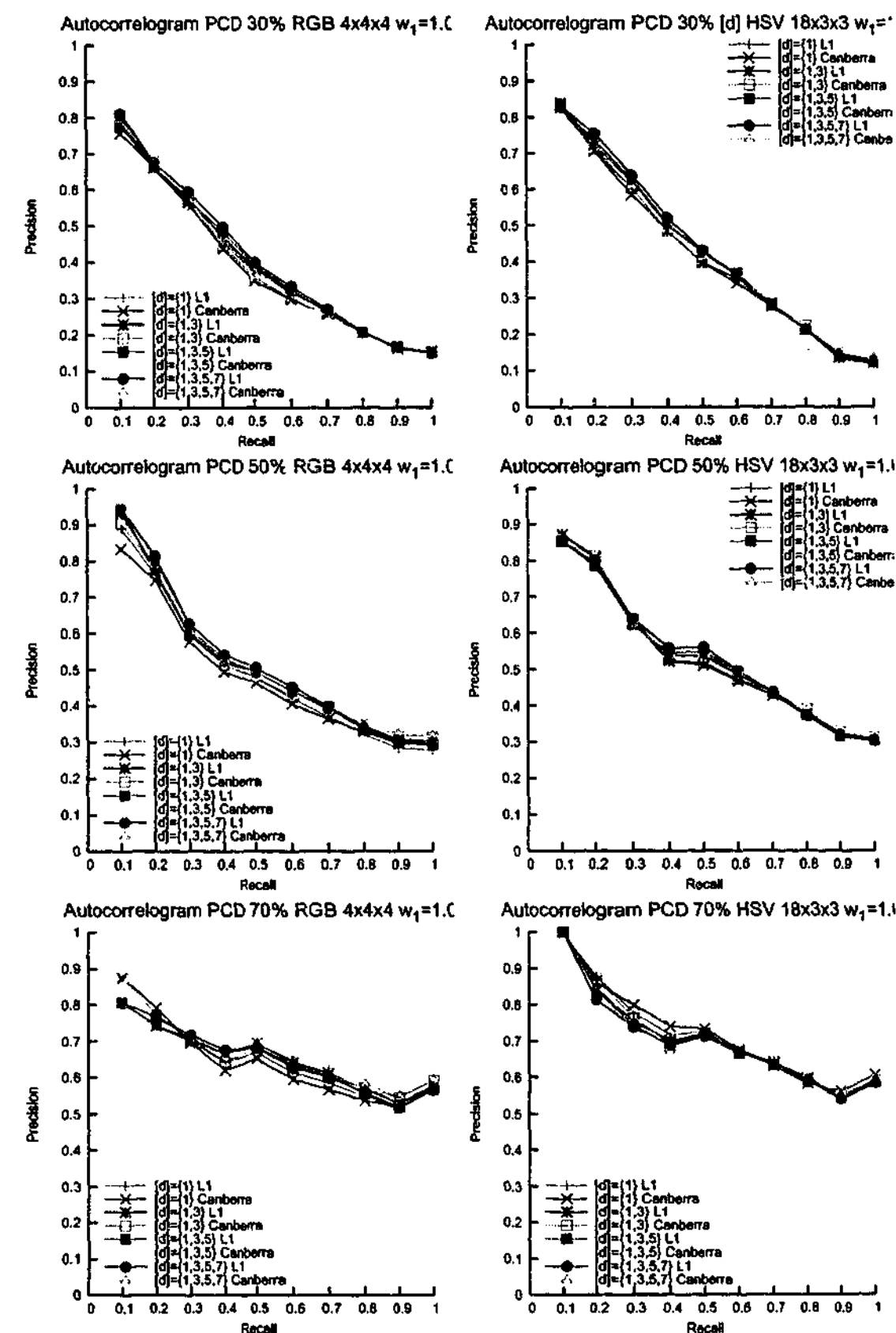


Figure C.1: ... continued from previous page.

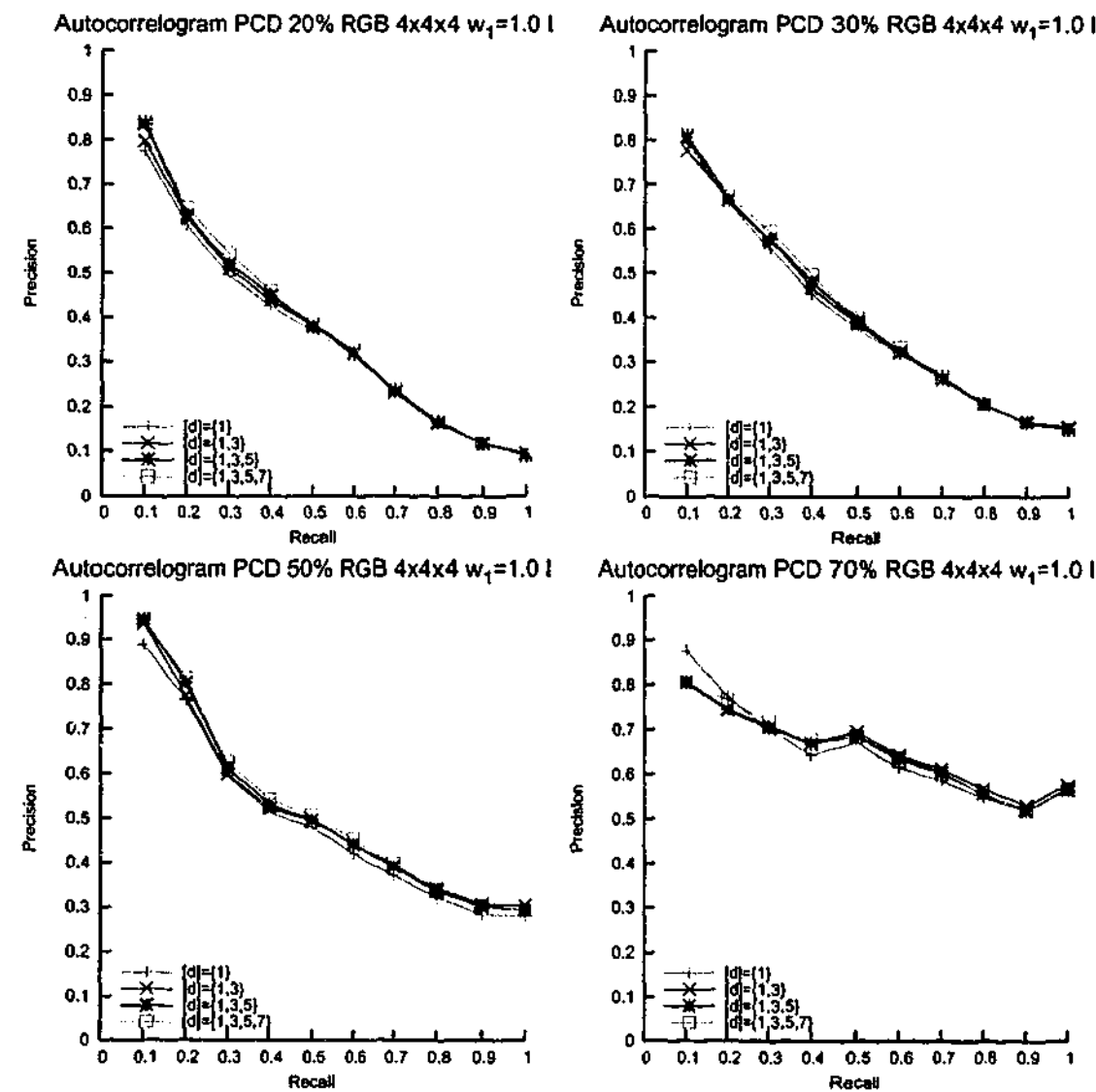
C.2 Autocorrelogram - Effect of d in PCD

Figure C.2: The PR graphs of the autocorrelogram at different d using RGB colour space in PCD.

C.3 Autocorrelogram - Effect of Colour Space in PCD

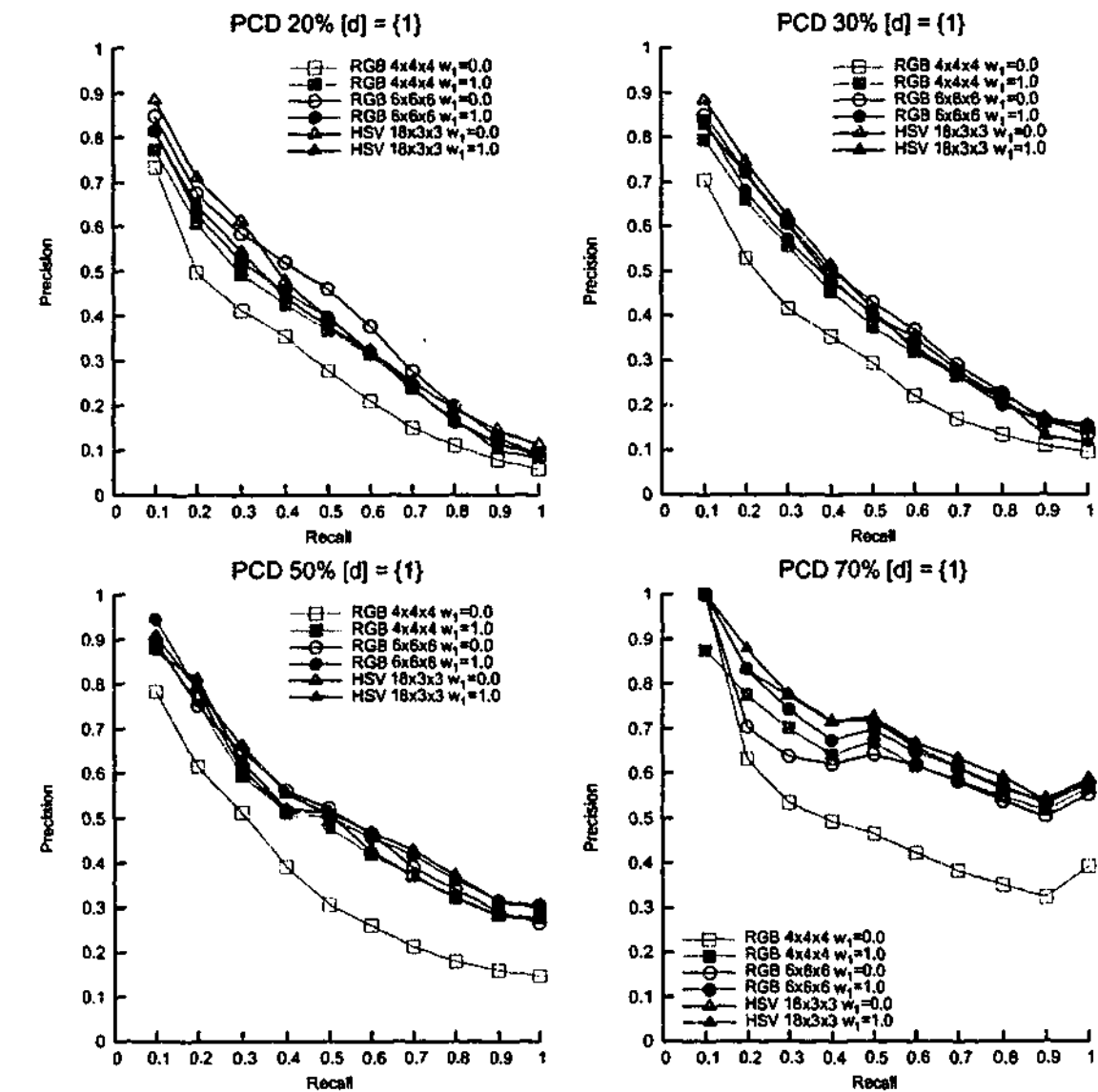
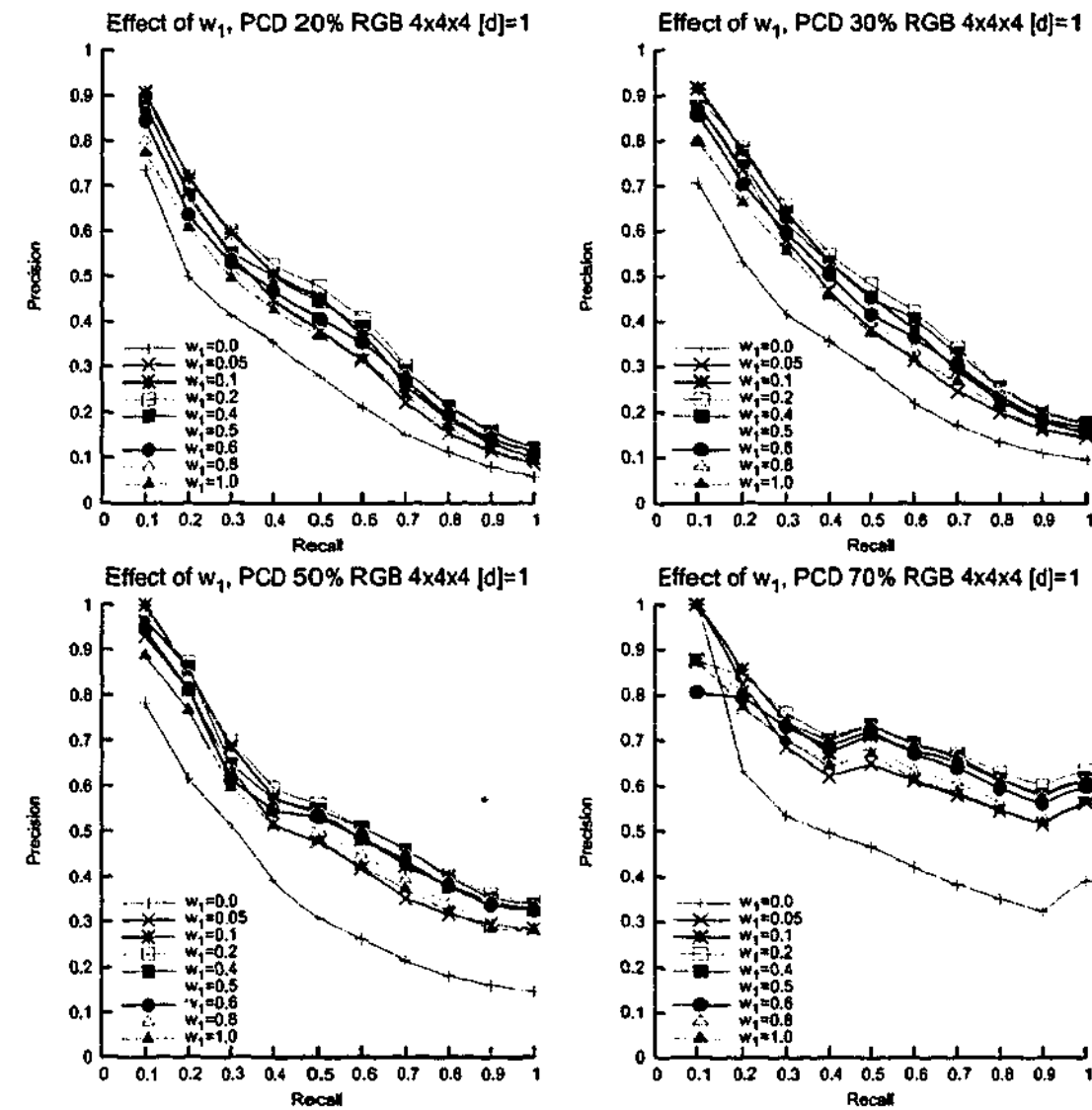
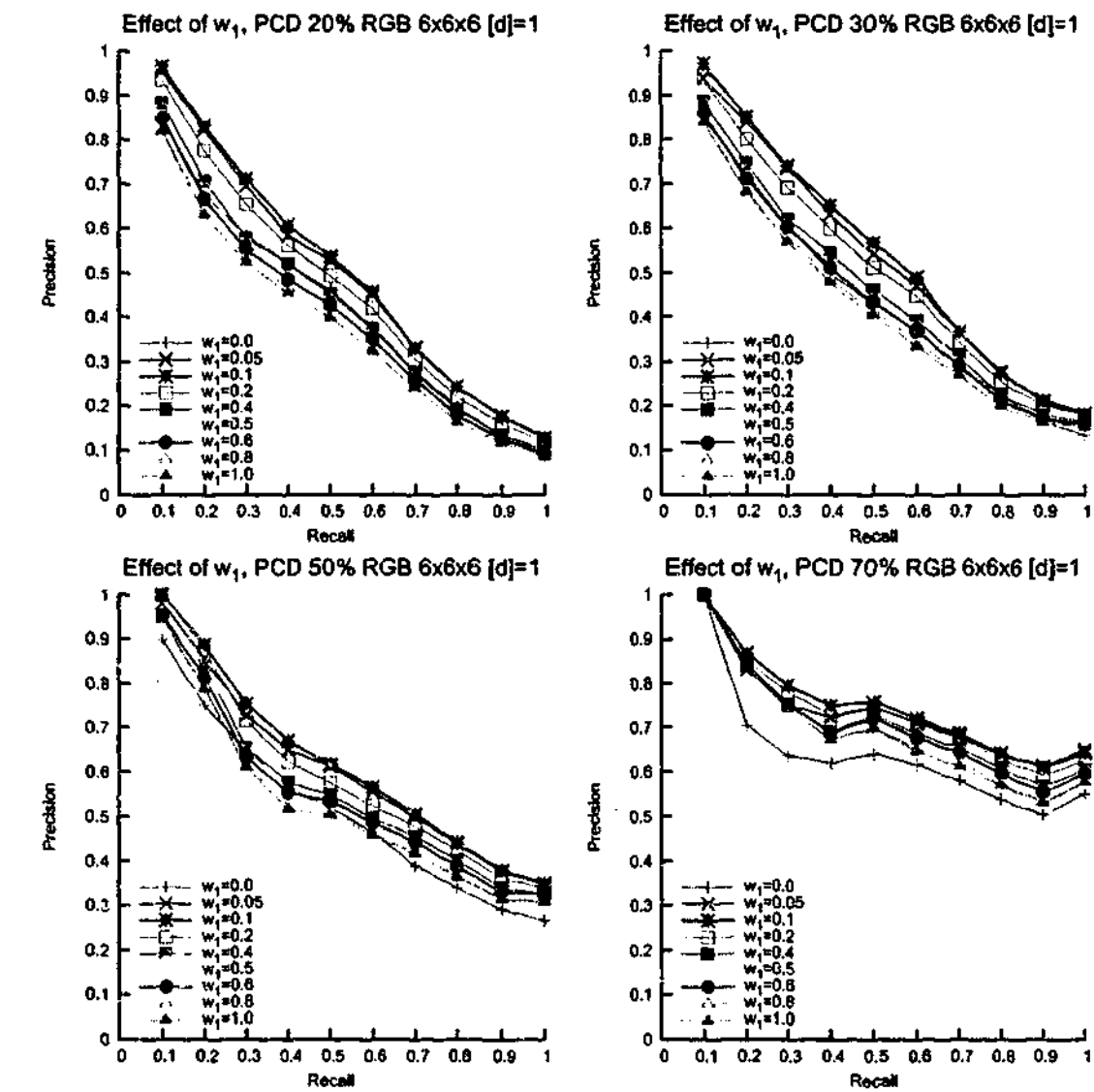
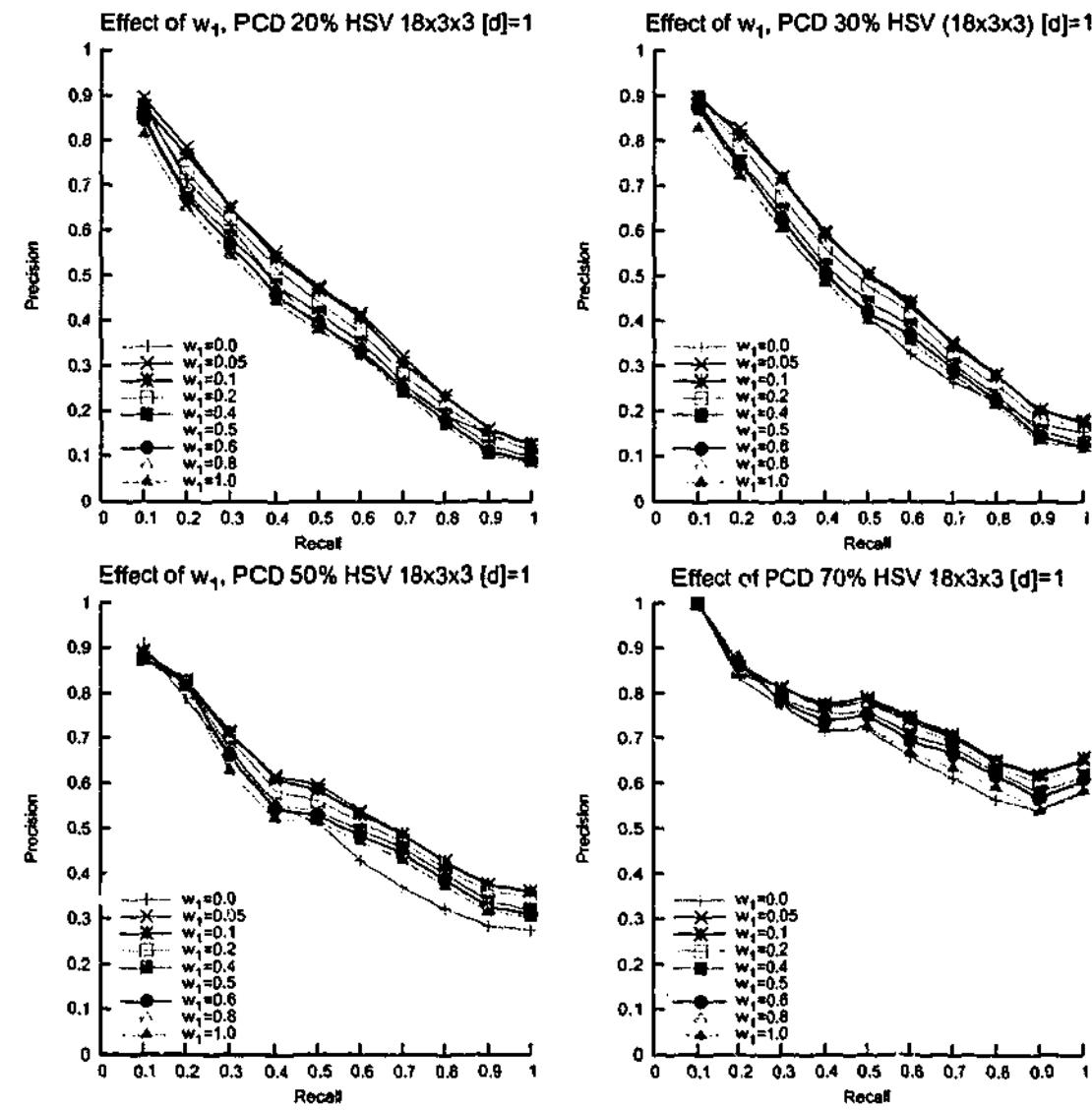
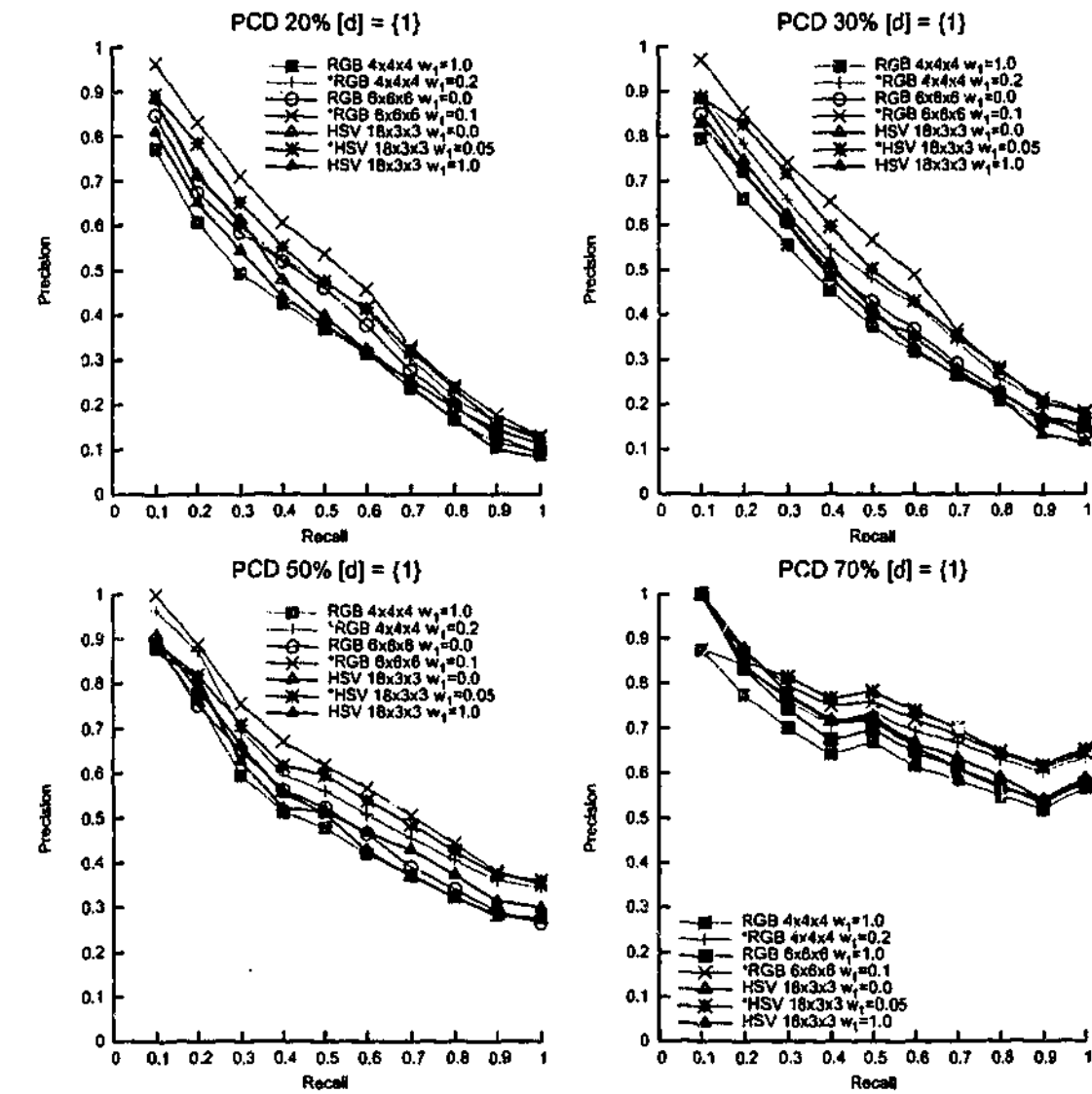


Figure C.3: PR graphs in PCD using RGB $4 \times 4 \times 4$ and RGB $6 \times 6 \times 6$ when $[d] = \{1\}$ for histogram ($w_1 = 0.0$) and autocorrelogram ($w_1 = 1.0$).

C.4 I-auto - Weighting of w_1 in PCDFigure C.4: PR graphs of the I-auto at different weight using RGB $4 \times 4 \times 4$ in PCD.Figure C.5: PR graphs of the I-auto at different weight using RGB $6 \times 6 \times 6$ in PCD.

Figure C.6: PR graphs of the I-auto at different weight using HSV $18 \times 3 \times 3$ in PCD.

C.5 Effect of Colour Spaces on I-auto in PCD

Figure C.7: PR graphs showing the effect of colour space on autocorrelation and I-auto (and histogram for HSV $18 \times 3 \times 3$).

C.6 Effectiveness of I-auto Compared to LCI and CSD in PCD

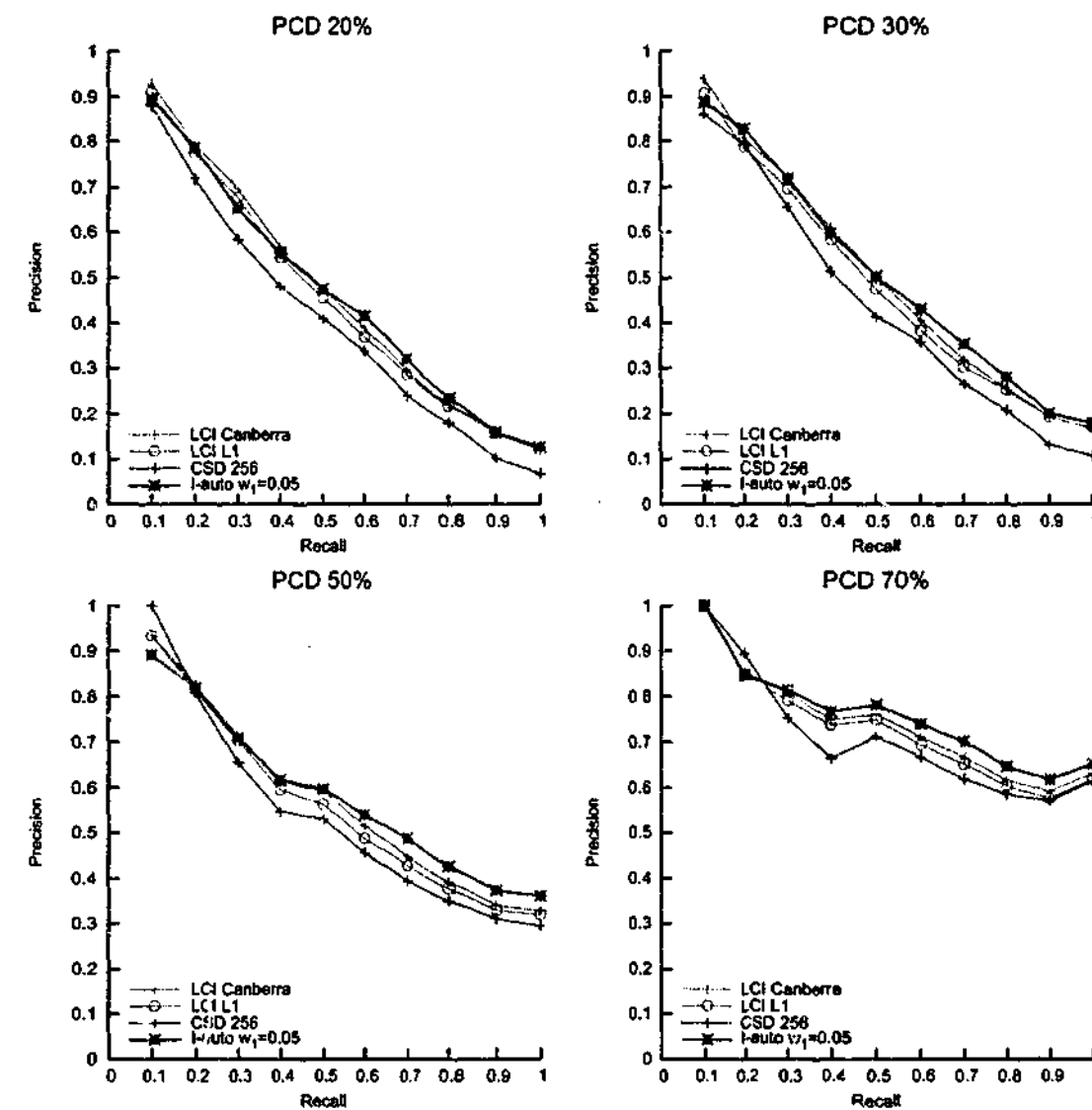


Figure C.8: PR graphs of LCI with $L1$ and *Canberra* dissimilarity metrics, CSD and I-auto in PCD using the HSV colour space.

Post Experiment Questionnaire of Usability Study Presented in Chapter 7

Each participant fill in a four page questionnaire after completing the task. To assist the participants in completing the questionnaire, the sequence for questions two and three are modified according to the order of programs and images being presented. They are also unaware of the names given to the target images on the last page of the questionnaire when performing the task.

Post Experiment Questions

1.Please describe how you performed the search? (e.g. how did you decide where to search)

Program A

Program B

Program C

Program D

Page 1

2. Please tick to answer the following questions

Strongly
Agree

Strongly
Disagree

a. Program A

a1. The task was reasonable

a2. The program was easy to use

a3. The program was enjoyable to use

a4. The image layout helps me in deciding where to start searching

a5. The program made it easy to find the target image

a6. The magnification tool was useful

a7. The magnification tool was easy to use

Strongly
Agree

Strongly
Disagree

b. Program B

b1. The task was reasonable

b2. The program was easy to use

b3. The program was enjoyable to use

b4. The image layout helps me in deciding where to start searching

b5. The program made it easy to find the target image

b6. The magnification tool was useful

b7. The magnification tool was easy to use

b8. Integration of visual query to browsing was useful

Page 2

c. Program C

	Strongly Agree			Strongly Disagree		
c1. The task was reasonable	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c2. The program was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c3. The program was enjoyable to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c4. The image layout helps me in deciding where to start searching	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c5. The program made it easy to find the target image	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c6. The focal/context view was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c7. Being able to hide images was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c8. The magnification tool was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
c9. The magnification tool was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

d. Program D

	Strongly Agree			Strongly Disagree		
d1. The task was reasonable	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d2. The program was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d3. The program was enjoyable to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d4. The image layout helps me in deciding where to start searching	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d5. The program made it easy to find the target image	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d6. The focal/context view was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d7. Being able to hide images was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d8. The magnification tool was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d9. The magnification tool was easy to use	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
d10. Integration of visual query to browsing was useful	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3. Are some images easier to search than others? If so, which ones are easy and which ones are difficult and why?



fruits & flowers



fireman



monkey



girl

4. Did you have a preference for any method? If so, which one, and why?

5. Any other comments: