Independent component filter based adaptive texture features applied to content-based image retrieval

by

Nabeel Mohammed, BCompSc

Thesis

Submitted by Nabeel Mohammed

for fulfillment of the Requirements for the Degree of

Doctor of Philosophy (0190)

Supervisor: Dr. David McG. Squire

Associate Supervisor: Dr. Peter Tischer

Clayton School of Information Technology
Monash University

April, 2014

I certify that I have made all reasonable efforts to secure copyright permissions for third-party content included in this thesis and have not knowingly added copyright content to my work without the owner's permission.

Contents

Symbols and Notations

Symbol	Description
$\|x\|$	Absolute value of x.
$\|\|Y\|\|_n$	L_n norm of vector Y.
$\bar{A}$	Mean of matrix A.
$sum(A)$	Sum of all the elements of the matrix/vector A.
$max(A)$	Maximum value of all the elements of the matrix/vector A.
$max(A)$	Mimimum value of all the elements of the matrix/vector A.
f	A filter. With a subscript f_i, it means filter number i.
$\hat{f}$	Filter f rotated by $180°$.
I	A grey-scale image. With a subscript I_i, it means image number i.
$I(x, y)$	Pixel value at location (x, y) for grey-scale image I.
V	An image feature vector. With a subscript V_p, it means feature vector for image p.
$V(j)$	Element number j of an image feature vector V.
v	A local feature vector extracted from an image region.
$C_{p,q}$	2-D normalised cross correlation of matrices p and q.
$E_{I,f}$	The response energy when image I is convolved with filter f. Sometimes the subscripts are dropped when talking about a general case.
$E_{I,f}(x, y)$	Response energy at location (x, y) of $E_{I,f}$.
$E\{x\}$	Expected value of x, not to be confused with $E_{I,f}$.
$d(\overrightarrow{a}, \overrightarrow{b})$	Euclidean distance between vectors $\overrightarrow{a}$ and $\overrightarrow{b}$.
$d_1(\overrightarrow{a}, \overrightarrow{b})$	L_1 distance between vectors $\overrightarrow{a}$ and $\overrightarrow{b}$.
$\chi^2(\overrightarrow{a}, \overrightarrow{b})$	Chi-squared distance between vectors $\overrightarrow{a}$ and $\overrightarrow{b}$.

Table 1: List of Symbols and Notations with their descriptions

List of Figures

Independent component filter based adaptive texture features applied to content-based image retrieval

Nabeel Mohammed, BCompSc
[redacted]
Monash University, 2014

Supervisor: Dr. David McG. Squire
[redacted]
Associate Supervisor: Dr. Peter Tischer
[redacted]

Abstract

In this research we concentrate on adaptive texture feature extractors which are automatically extracted from an image collection. We use independent component analysis (ICA) to extract independent component filters (ICF). ICF have previously been shown to include edge filters, having properties similar to the receptive fields of simple cells of the human visual cortex. In this thesis we evaluate the utility of ICF-based collection-specific features in the context of content-based image retrieval (CBIR), with the view to demonstrate the viability of using such automatic collection-specific features.

We find that global features extracted using a small number of ICF outperform those extracted by a bank of Gabor filters, even when very large number of Gabors are used. We also present comparisons against a variety of state-of-the-art features and show that ICF-based features perform better than these, without the need for any hand-tuning.

ICA extracts large number of filters. In order to find a useful smaller subset we evaluate a previously-published variance-based filter selection method. We identify the shortcomings of this method. Our proposed improvements to filter-based feature selection and extraction, response scaling and locally normalised convolution, was a result of trying to address some of these shortcomings. Even with these improvements the variance-based method has an intrinsic flaw of being susceptible to selecting redundant filters. We propose a new filter selection method based on normalised cross-correlation and clustering. We show that our method selects a more useful subset compared to the previously published variance-based method.

We also propose a salient-point based local feature which uses ICF and show that our proposed feature performs better than SIFT features and also the global ICF features. We further illustrate the utility of ICFSIFT by developing a grid-based adaptive local feature and we demonstrate that ICFSIFT extracts different and useful information compared to the ICF-based global and grid-based local features.

After applying the techniques proposed in this thesis to four standard texture collections, we show that, while hand-tuned features can work well for their target collection, the automatic adaptive features perform better for the general case.

Independent component filter based adaptive texture features applied to content-based image retrieval

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Nabeel Mohammed
April 10, 2014

Acknowledgments

For Allah the Almighty, sincere thanks and gratitude for granting me the opportunity to undertake this research.

For my supervisor, Dr. David Squire, deep thanks for not only guiding me through the turbulent headwaters that are part of any candidature, but the time he invested in mentoring me and helping me become a more rounded human being.

For my associate supervisor, Dr. Peter Tischer, thanks for valuable inputs to this research and the constant encouragements. Similarly, Professor Tom Drummond, thank you for your input on the part of the research based on SIFT.

For my family, who have unstintingly supported me, encouraged me, believed and trusted in me, all my love and thanks.

For Mohamud Hersi, my appreciation for all his invaluable help with the diagrams in this thesis. Further, for Leeanne Evans and Sevim Zongur for taking the administrative burden out of my candidature, as they selflessly do for all candidates.

Nabeel Mohammed

Monash University
April 2014

Chapter 1

Introduction

1.1 Motivation

Image features are an integral part of many image processing applications. They are the key to the understanding of the content of an image for an automated-system. If the features are good at describing the content, the system will have good results. Equally, if the features are misleading, then the system will also give misleading results. With the advent of the web, the affordability of digital cameras, the availability of cheap storage, and the increasing application of image processing techniques in important fields such as medicine — image processing applications are more important now than ever before, and so is the role of image features.

There are a multitude of features from which a system designer can choose. Typical feature types include colour, shape, and texture. There is, however, no consensus on what are the best features. A designer usually chooses a feature set and their configuration that is thought to serve a given purpose best. In cases where the images in question come from a particular collection and/or domain, the designer can tune the features to suit that domain. The effectiveness of such tuning is usually measured through the performance of the system. When a system has to deal with a more general problem, the features are usually selected to cater for a broad range of images. In both cases the systems end up with a set of fixed and pre-defined features.

This thesis is based on the hypothesis that:

> For many image processing applications, adapted collection-specific features would be more effective at identifying useful and pertinent image content than pre-defined and fixed features, for the target image collection.

Such collection-specific features may be determined manually using human participation. The hand-tuning process in such a scenario usually consists of trying a large set of features (maybe in different combinations and configurations) and choosing the set-up that performs best. This requires that a judgement be made for what is "best", and will require human participation in such a decision. As one can appreciate, with a large enough image collection, and a large enough set of candidate features, such a determination can be a time-consuming and exhaustive undertaking. It might be possible to automate the feature extraction and processing part, but determining "what is performing well" is not an easy task outside of the standard collections used by researchers.

So, imitating human action and going through a large list of features and combinations is not a suitable method for an automated system which wants to use adaptive collection-specific features. A method which can *learn* the collection-specific features from a sample of the images of the collection directly is more suitable, as it would not require a measurement of "what is performing well". In this thesis we use one such method, independent Component Analysis (ICA), which has been successfully used to extract adaptive collection-specific independent component filters (ICF) from image collections. These ICF are learnt automatically without any human intervention and are known to extract texture features. This brings us to the first specific concentration of the thesis: we exclusively concentrate on texture features, in particular our contributions are all based on texture features extracted by collection-specific filters.

We have just said that the advantage of using a technique like ICA is that there is no requirement to measure "what is performing well". This however depends on having confidence that features extracted using ICF can be useful. The work in this thesis aims to explore whether such a confidence is justified. To do so we used the following broad approach:

1. Extract collection specific filters from standard texture collections.
2. Extract images features using these filters.
3. Extract features using numerous texture features.
4. Compare the performance of the different features in an image processing application.

The contributions arising out of this thesis concentrate on first two steps. However we needed an image processing application through which to compare the performance of the different features. We have done all our evaluations using content-based image retrieval (CBIR). Given these two parameters, CBIR and collection-specific texture features, we can state our hypothesis as follows:

> Many CBIR systems apply a set of fixed pre-defined texture features for every image collection. It is entirely possible that such fixed pre-defined features will not identify useful and pertinent image content for every collection. Some of these features can be blind to textures present in the collection, as has been shown in previous studies. In such a case, automatically learnt collection-specific ICF, found through ICA, will be more useful than collection neutral features.

We do not claim that the automated methods presented in this thesis are always superior to human hand-tuning. It is very difficult to replace human intuition and ingenuity with an automated measure. We simply present techniques which make automatically learnt collection-specific texture features a viable option for CBIR designers when hand-tuning is not done, and which perform better than hand-chosen feature sets in many cases.

The application of ICA to extract ICF from image collections is already documented in literature. ICA and ICF has also been applied to some image processing applications, i.e. image classification, by some researchers. In this thesis we document the contributions mentioned in the next section, which enable us to demonstrate the validity of our hypothesis.

1.2 Contributions

Improvements to filter-based feature extraction, also resulting in improvement to previously published variance-based filter selection: It is common to use convolution to obtain the filter response of an image. In fact CBIR systems such as the GNU Image Finding Tool (GIFT) use the responses obtained from a bank Gabor filters to extract its texture features. Previous ICF-based image classification work used the variance of filter responses for filter selection. We demonstrate that even after doing global normalisation, the responses of ICF vary by orders of magnitudes. Any statistics gathered from these will thus be unduly affected. We introduce response scaling as a method to ensure that statistics gathered from filter responses (or their energies) are not affected by the differences in their scales. We show that response scaling improves overall CBIR performance and also improves the filter selection of the variance-based filter selection method proposed by Le Borgne and Guérin-Dugué (2001).

We have also found the standard convolution does not cope well with local image intensity differences. We introduce locally normalised convolution (LNC), an adaptation of normalised cross-correlation to deal with such local intensity differences. We show that by using LNC, it is possible to identify texture features which standard convolution would not identify. We show that it improves CBIR performance. We also show that combining LNC with response scaling gives significantly better performance for filter-based featuers (ICF and Gabors). We show that ICF-based adaptive features perform better than banks of Gabor filters, two GLCM features, HOG and PHOG when applied to images with globally consistent texture. The details of response scaling and LNC are described in chapter 4 (Mohammed and Squire, 2013c).

New filter selection technique based on normalised cross-correlation followed by clustering: ICA extracts a large number of filters, typically hundreds. It may not always be practical to use such a large number of filters. Consequently, we need techniques that select a smaller subset of useful filters. The variance-based method proposed by Le Borgne and Guérin-Dugué (2001) calculates the variance of filter responses across the image collection (or a representative subset). The filters with the highest variance are deemed to be most useful. We have already mentioned that any statistics gathered from convolution energies have the potential to be affected by the difference in scales of the responses of different filters. The variance-based filter selection method was similarly affected, and we addressed it through response scaling. However this method does not take into account the fact that ICA can extract ICF which are close to being shifted/duplicate versions of each other. Such similar filters naturally have similar responses, leading to similar variances, causing the variance-based method to select a filter subset which has redundancies.

To address this problem we propose a filter selection method which uses normalised cross-correlation as a method to judge filter similarity. We use the similarity measurements in an implementation of complete-link clustering followed by a simple heuristic to choose a filter from each cluster. We show that filters selected by this method consistently outperform the filters selected by the variance-based method. A demonstration of the shortcomings of the variance-based method is shown in chapter 3 and the clustering-based method is described in chapter 4 (Mohammed and Squire, 2011b, 2013c,a).

ICFSIFT: A salient point-based adaptive local feature: In chapter 3 we show that features which divide an image into sub-regions and extract features from each subregion performs poorly for images with globally consistent texture, if they encode local information in their feature vectors. We present ICFSIFT, which uses SIFT-like keypoints to identify locations of interest. Collection-specific ICFSIFT filters are learnt from key point patches of training images. They are then applied to image collections and the bag of words method is used to create an image descriptor from the ICFSIFT local descriptors. We show that ICFSIFT features consistently outperforms its fixed pre-defined counterpart SIFT for our test collections, which also means they outperform other features such as HOG, PHOG, GLCM features, and features extracted from a bank of Gabor filters ICFSIFT is described in chapter 5 (Mohammed and Squire, 2013b).

To further illustrate the utility of ICFSIFT, especially its keypoint-based processing, we create an ICF-based local feature extracted from a image regions after an image is divided into a fixed grid. We use the bag of words method to create a spatial location independent image descriptor. We show that this feature, ICF-BOW, performs better than other similar counterparts using fixed and pre-defined filters as well as ICFSIFT. However combining ICFSIFT with ICF-BOW results in significant performance improvements, leading us to conclude that the keypoint-based processing of ICFSIFT gives different and useful information. This is further discussed in chapter 6.

1.3 Thesis outline

This thesis has nine chapters. They can be broken down into five parts.

Part 1: This chapter which introduced the motivation behind the research and presented the hypothesis.

Part 2: Chapters 2 and 3 set the context in which the rest of the research is carried out. Chapter 2 is a review of topics which are related to our research. In chapter 3 we present the image collections we use most often, the details about ICF extraction, some global and grid-based features, concrete details about how we evaluate performance, and shortcomings of the variance-based filter selection method.

Part 3: Chapter 4, 5 and 6 document the contributions arising from this thesis. Chapter 4 documents response scaling, locally normalised convolution and the clustering-based filter selection method. Chapter 5 describes ICFSIFT, our proposed keypoint-based adaptive local features. In Chapter 6 we describe a grid-based adaptive local feature which does not encode spatially local information in its feature vector. We demonstrate the utility of the ICF-based features by showing that using all of them together gives better performance compared to using any one individually or two in combination.

Part 4: In chapter 7 we do not describe any new technique, rather we apply the adaptive features in a comparison with hand-tuned features. We compare against four texture collections and show that while hand-tuned features can work well for their target collection, the adaptive features perform better for the general case.

Part 5: In chapter 8 we briefly describe some future work and conclude this thesis in chapter 9.

Chapter 2

Background

This chapter discusses topics relevant to this research. This thesis concentrates on collection-specific adaptive features, especially related to texture. We start by describing some commonly used image features in §2.1–§2.4. We then present Independent Component Analysis (ICA), and, show how it can be used to extract collection-specific filters, in §2.5. One of the main challenges with using these adaptive filters is their large number, so we present some techniques, from existing literature, that can be used to select a useful subset of filters in §2.5.6, although they are very limited. We use Content Based Image Retrieval (CBIR) as a method to evaluate the performance of collection-specific filters. We present CBIR and methods to evaluate its results in §2.6 and §2.7

2.1 Image features

Most digital images are collections of discrete pixel data. The challenge of any image feature is to extract useful information from this. Using just the raw pixel data can be difficult due to its size, or the fact that it is usually not stable even under minute changes of position, lighting, rotation, and scale. Image features aim to extract image descriptors/feature vectors which can be useful and discriminative without being easily influenced by these changes. They should ideally be compact, and for certain applications, efficient (e.g. real time systems). We use the following three categories proposed by Marques and Furht (2002) to organise the vast number of features present:

1. Low-level features (such as colour, texture, etc). These are typically extracted directly from the raw pixel values of an image.
2. Middle-level features (such as regions or blobs). These are extracted using image segmentation.
3. High-level features. This class of features is concerned more with the semantics (meanings, role, etc.) of an image and its contents.

Currently the most widespread practice in CBIR is to use low-level features (Datta et al., 2008). Extraction of high-level features still requires human assistance, and these are the hardest to automate. In this chapter we present a brief review of shape and colour features and a more in-depth review of some features that can be employed to represent image textures.

2.2 Shape and Shape-Based Features

Shape is a very important characteristic present in certain images and can be a very useful feature. However, ascertaining it is a difficult task and is made even more so by the loss of a dimension in projecting a 3-D object onto a 2-D image plane (Zhang and Lu, 2004). One of the challenges in extracting such features is finding salient regions/blobs within images from which to extract the features (Shapiro et al., 2001; Datta et al., 2008). A discussion on image segmentation is out of the scope of this thesis, however we briefly describe some shape-based features here.

Shape-based features can be split in two broad categories, contour-based and region-based, depending on whether the features are extracted just from the boundary of the shape or from the whole region.

2.2.1 Contour-based shape features

Commonly used contour-based shape features include area, eccentricity, major axis orientation, circularity and bending energy (Young et al., 1974). These are usually global descriptors and are useful for differentiating shapes which are significantly different and are used often (Carson et al., 1999).

Belongie et al. (2001) proposed shape contexts, which extracts a shape context for each point on the shape's boundary. So, if p_i is one of N sample points lying on the contour of a shape, for every p_i, the relative position of every p_j, $j \neq i$ is computed. A histogram of these positions forms the "shape context" of p_i. The context is extracted from log-polar space to make the feature more sensitive to positions of nearby points. An example is shown in Figure 2.1.

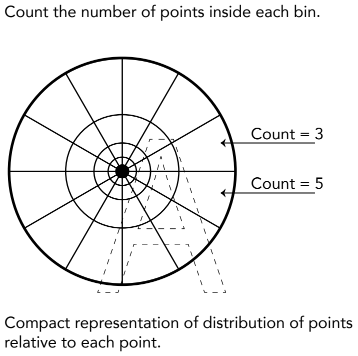

Count the number of points inside each bin.
Count = 3
Count = 5
Compact representation of distribution of points relative to each point.

Figure 2.1: Shape context being calculated for a point.

There are numerous other approaches to contour-based shape features. Asada and Brady (1986); Mokhtarian and Mackworth (1986); Daoudi and Matusiak (2000) and others have attempted to interpret interval-trees resulting from space analysis, as shape features.

Methods to break down the boundary into segments called primitives and extract features from them are mentioned in numerous studies including Pavlidis (1982).

Fourier descriptors have been used to extract shape features and have the advantage of coming from a widely used and understood Fourier theory background. El-ghazal et al. (2009); Chen et al. (2009); Ortega et al. (1997); Arbter (1989); Brill (1968); Sanchez-Marin (2000) and many others have used Fourier descriptors to represent shapes.

Wavelet-based descriptors have also been used to describe shapes (Ohm et al., 2000; Tieng and Boles, 1997; Yang et al., 1998), but they have the drawback of having complex matching schemes which make it difficult for practical use.

2.2.2 Region-based shape features

In region-based techniques the shape feature is extracted from all the pixels that lie within a shape.

Grid-based features have a grid of cells overlaid on a shape (Lu and Sajjanhar, 1999; Safar et al., 2000). The grid is scanned and if a cell overlaps the shape it is given a value of one. A cell which does not overlap the shape gets a value of zero. This bitmap is then used as the feature vector. A similar idea with a circular grid is proposed by Goshtasby (1985), called a shape matrix. However this uses sparse sampling and is sensitive to noise.

Zhang and Lu (2002) introduces Generic Fourier Descriptor (GFD), extracted by applying a two dimensional Fourier descriptor on a shape image obtained by polar-raster sampling. Compared to moment-based features (Teague, 1980; Liao and Pawlak, 1996), the GFD features perform better.

There are many other shape features and this is a very active area of research. As far as adaptive features go, the challenge would be to find shapes which are peculiar to certain image classes. This then becomes not only a shape extraction problem but also a shape discrimination and assignment problem. There is great potential in this area to develop methods to identify shapes which are the building blocks for images in a particular collection.

2.3 Colour

Colour features have been used extensively in image processing with numerous color feature extraction techniques being proposed. The choices in extracting colour features involve choosing both the colour model to use and the actual procedure of feature extraction. We first present some of the most commonly used colour models and then describe some popular colour feature extraction methods.

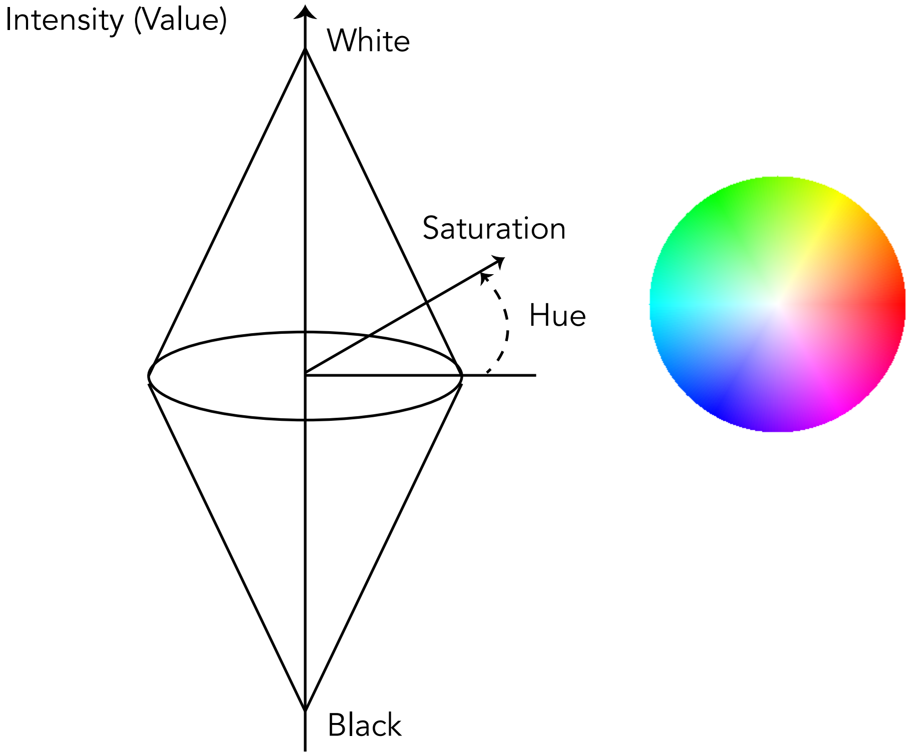
2.3.1 Colour Models:

Three commonly used colours models are presented here:

- RGB: This scheme is used mostly in hardware-based colour schemes to represent digital images. The colour space can be represented as a unit cube (Marques and

Furht, 2002). A colour is formed by using the basic colours (red, green and blue in this model) in combinations.

- HSV: HSV stands for Hue-Saturation-Value, sometimes also called Hue-Saturation-Brightness (HSB). This was developed to transform the RGB unit cube into a set of dimensions that better mimic the way an artist mixes colours (Smith, 1978). Hue-Saturation-Lightness (HSL), which is also referred to as HLS sometimes, is also similar to HSV (Joblove and Greenberg, 1978). These models are non-uniform and are often represented graphically as a double cone, an example of which is shown in Figure 2.2, representing the HSV cone. Changing H is equivalent to traversing a colour circle (Figure 2.2(b)), decreasing V increases blackness and decreasing S increases saturation. There is some evidence from experiments conducted on human subjects that these colour models are more compatible with human perception compared to RGB (Berk et al., 1982).



(a) HSV cone

(b) Color plane

Figure 2.2: The HSV colour model (Marques and Furht, 2002)

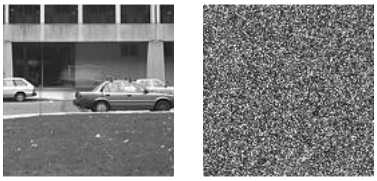
- CIELAB: The CIELAB is an international standard to ensure perceptual uniformity of colour. The motivation was to ensure that Euclidean distances in the CIELAB space map to human perception of colour difference (Schwarz et al., 1987). The coordinates of this colour model is described as $L*, a*$ and $b*$, where $L*$ is a measure of lightness, $a*$ a measure of red/green balance and $b*$ is a measure of blue/yellow balance.

2.3.2 Colour Features

Colour features are low-level features. Some commonly used colour features are described below.

Colour Histograms

The use of colour histograms in various different ways is probably the most popular colour feature. Colour histograms are obtained by counting the number of occurrences of each colour in a given image (Swain and Ballard, 1990). These histograms are useful as they remain invariant even if the images are translated or rotated about the viewing axis and change slowly when changes occur in the viewing angle, scale and occlusion (Swain and Ballard, 1990). Global colour histograms are calculated over the whole image, as implemented in the GNU Image Finding Tool (GIFT). It uses the HSV colour model and a palette of 166 colours (Squire et al., 1999). Such global colour features do not contain any information pertaining to the spatial arrangement of the colour values, therefore very different images can have identical colour histograms, such as the ones shown in Figure 2.3.



(a) (b)

Figure 2.3: Two images with identical global colour histograms.

This shortcoming can be addressed somewhat by calculating colour histograms on image regions as does the QBIC system (Faloutsos et al., 1994), where colour features are extracted on regions selected by the user, or as in the work by Vimina and Jacob (2012), where colour histograms are extracted from pre-determined sub-regions.

Another Content Based Image Retrieval system that uses region-based colour histograms is Blobworld (Carson et al., 1999). In Blobworld, an image is divided into multiple regions, and the colour histograms are calculated for each region. The histograms have a bin width of 20 in each dimension of the L*a*b* colour space. With 5 bins in the L* dimension and 10 bins in a* and b* dimensions, there are a total of 500 bins. 282 of them are never used as they don't fall in the range $0 \leq (R, G, B) \leq 1$. Colour regions are matched using a quadratic distance calculation.

Colour Moments

Stricker and Orengo (1995) describes the use of the first three statistical moments of each colour channel as a colour feature. The first moment provides information about the mean colour value. The second and third moments provide information about variance and skewness. Calculated on each of the three colours channels of a colour model like HSV, there will be a total of 9 features. Each moment has a different weight associated with it. Using these weights, the difference between images is calculated as the weighted sum of the absolute value of the differences of the moments.

Colour sets

Smith and Chang (1996) first proposed colour sets. It provides a method to extract colour features and retain some regional information. An image is split up into salient image regions. Each region is analysed to find the dominant colours. The presence of just the dominant colours is encoded in a binary vector. The criterion of choosing dominant colours is controlled by a threshold, for example: the number of occurrences of a colour must be at least 20% of the total for it to be considered a dominant colour. Note that the presence of the colours is stored, as either present or absent (hence the binary nature of the vector), as opposed to colour histograms which store the amount of colour present.

Smith and Chang (1997) describe VisualSEEK, a CBIRS which uses colour sets as a method of colour feature extraction. They leverage this technique to allow users to draw simple images and search for images with a similar regional distribution of dominant colours.

2.4 Texture

The research described in this thesis concentrates almost exclusively on texture. Giving an accurate description of visual texture is difficult, as no single definition exists for it, although it has been studied for a very long time. The following are some of the numerous definitions that exist in relevant literature.

> We may regard texture as what constitutes a macroscopic region. Its structure is simply attributed to the repetitive patterns in which elements or primitives are arranged according to a placement rule" (Tamura et al., 1978).

> The image texture we consider is nonfigurative and cellular... An image texture is described by the number and types of its (tonal) primitives and the spatial organization or layout of its (tonal) primitives... A fundamental characteristic of texture: it cannot be analyzed without a frame of reference of tonal primitive being stated or implied. For any smooth gray-tone surface, there exists a scale such that when the surface is examined, it has no texture. Then as resolution increases, it takes on a fine texture and then a coarse texture. (Haralick, 1979)

> The notion of texture appears to depend upon three ingredients: (i) some local order is repeated over a region which is large in comparison to the order's size, (ii) the order consists in the non-random arrangement of elementary parts, and (iii)the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region. (Hawkins, 1970)

> The variation of data at scales smaller than the scales of interest. (Petrou and García-Sevilla, 2006)

From the definitions above, we can deduce the following properties about texture.

- Texture is a property of a region, unlike colour which is a property of a pixel.
- For texture to be discernible it must have some kind of order or repetition.
- The visual elements which are repeating would have some appearance of similarity between each other.

Not following from the definitions — an aggregation of visual elements with the characteristic of having a shape might create a textured region, however in general a texture is not an indication of the shape of objects which may be present in an image.

Even after all these attempts at describing visual texture, it is perhaps best described by showing some examples of it. Figure 2.4 illustrates a few images with texture characteristics in them.

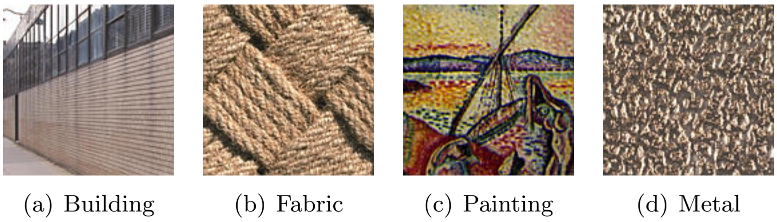

(a) Building (b) Fabric (c) Painting (d) Metal

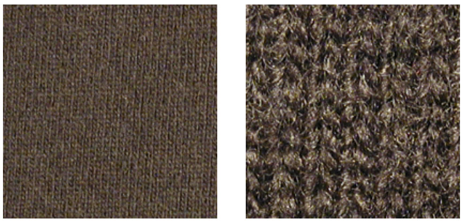
Figure 2.4: Some examples of visual texture

Figure 2.4(a) shows an obvious challenge of having multiple textures in a single image (the pavement, the brick wall, and the windows). However even for an image which has apparent global consistency, like Figure 2.4(b), it is easily seen that there are at least three different scales of texture present — (i) The granularity within the threads, (ii) the granularity of each individual thread, and then the (iii) granularity of the collection of threads woven together. The painting in Figure 2.4(c) is an interesting example where, if the textures present in the image are analysed and used for image similarity, we almost disregard the subject of the painting but concentrate on an artefact of the painting technique.

The second definition mentioned above encapsulates one main challenge with texture analysis, that of scale. Figures 2.5(a) and 2.5(b) show the same fabric photographed at different scales. From a purely visual point of view these are very different textures, but in certain situations they need to be treated as similar.

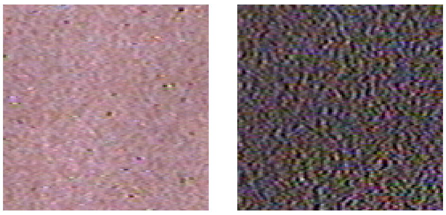
Another issue is that the same material exhibits different textures under different lighting conditions. Figure 2.6(a) and 2.6(b) show the same material photographed under two different illumination levels, and they clearly exhibit different textures.

There are other challenges, such as rotation (especially out of image plane rotation), image noise etc. The texture features described below can cope to a certain extent with some of these issues, e.g. scale, illumination, rotation. However the challenge of catering



(a) (b) Fabric

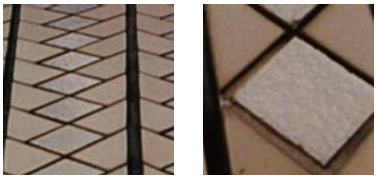
Figure 2.5: The same material photographed at different scales.



(a) (b)

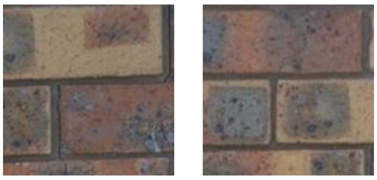
Figure 2.6: The same material photographed under different illumination conditions.

to differing user intent is still difficult. Most research gets around this issue by testing against standard image collections with pre-defined ground-truth. The results are always dependent on the relevance judgements of the collection and sometimes they can appear to be counter-intuitive. For example Figures 2.7(a) and 2.7(b) show two textures which are judged to be similar, although the scales are quite different. In the same collection the textures in Figures 2.8(a) and 2.8(b) are judged to be different although they look exactly the same in scale, rotation and illumination and also in colour. Results presented in this thesis and in fact in any texture research should be interpreted with this in mind.



(a) (b)

Figure 2.7: Two textures at different scales judged to be similar in a standard texture collection.



(a) (b)

Figure 2.8: Two textures which are visually similar in terms of scale, illumination and rotation but judged to be different in a standard texture collection.

Below we discuss some texture features which have been used quite successfully for texture analysis. No one feature exists which works best for all situations, so there exists a multitude of techniques for texture feature extraction. Different authors have made different categorizations of these techniques (Tuceryan and Jain, 1998; Gonzalez and Woods, 2007; Howarth and Rüger, 2004). Here the broad but inclusive categorisation by Gonzalez and Woods (2007) is used. This categorization includes statistical models, structural models and spectral models. These are not strict categorisations and in fact many of the features discussed here can very easily fall into multiple categories. We haved added a fourth category called “Gradient-Based Features” to include SIFT, HOG and PHOG features. These are features which do not necessarily fit into any of the above three categories, and have not been popularly employed as texture features. They can, however, be thought of as texture features as they find information similar to texture feature extractors.

2.4.1 Statistical Models

Statistical methods are some of the earliest methods proposed, as the spatial distribution of grey values in texture seems to fit this model rather well. The simplest features in this category employ first, second or high-order statistics to extract texture features, for example calculating the variance from the grey-level histogram of an image region has also been used to extract information about relative smoothness (Kelly et al., 1995). There are also the Wold features, the six Tamura features, Grey Level Co-Occurence Matrix and Local binary patterns. We presented these in more detail here.

Wold features

Rao and Lohse (1992) described “periodicity”, “directionality” and “complexity” as the most perceptually relevant dimensions of texture. As discussed is §2.4.1, the Tamura features also try to capture some of these aspects of texture.

Francos et al. (1993) proposed a texture model called the Wold decomposition model which models texture as a 2-D homogeneous random field. The Wold decomposition theorem (Liu and Picard, 1996) decomposes a real-valued, regular, homogeneous random field into three mutually orthogonal fields:

$$y(m, n) = p(m, n) + d(m, n) + c(m, n) \tag{2.1}$$

where $\{p(m, n)\}$ is deterministic, $\{d(m, n)\}$ is deterministic and generalised evanescent, and $\{c(m, n)\}$ is non-deterministic. p can be used as a model of the “periodicity” of a texture. d can used to model the directionality, and c can be used to model the complexity.

Liu and Picard (1996) analyse the local maxima of the discrete Fourier transform of a zero-mean, Gaussian-tapered image to estimate the periodicity component. A histogram of line slope angles, built from the Hough transform of the DFT, is used to estimate the directionality. The complexity is modelled using the 2-D autoregressive model.

Tamura features

Tamura features were proposed by Tamura et al. (1978) and are still widely used (Datta et al., 2008; Kaur and Mann, 2012; Ho et al., 2012). In total six features were proposed that could be extracted from pixel statistics. In the original paper the authors used four of their proposed features, as the other two utilised one or more of the first four. We will briefly discuss the six features below.

The six Tamura Features

Coarseness: Textures at multiple scales are usually present in an image, as highlighted previously in Figure 2.4(b). The coarseness measure is an estimate of the size of textures present in the images. The coarseness measure estimates a size that best characterises the scale of textures present in the image. It is designed with the intention to choose a large size as best when coarse texture is present, even if micro-texture is also present. The method will pick a small size as best if only micro-texture is present. The following steps can be used to calculate the coarseness feature F_{crs}:

- Step one: For neighbourhood sizes which are powers of two, e.g $1\times1, 2\times2, 4\times4..., 32\times32$, the average intensity of the neighbourhood is found. So, if k is the neighbourhood size, then $A_k(x, y)$ is the average of the $2^k \times 2^k$ neighbourhood around the point (x, y) and is calculated as

$$A_k(x,y) = \sum_{i=x-2^{k-1}}^{x+2^{k-1}} \sum_{j=y-2^{k-1}}^{y+2^{k-1}} \frac{f(i,j)}{2^{2k}} \tag{2.2}$$

  Where $f(i, j)$ is the gray-level at (x, y).

- Step two: For each point, for each size, differences between non-overlapping neighbourhoods on opposite directions of a point (x, y) are found. This is done for both horizontal and vertical orientations as shown below.

$$D_{k,horizontal}(x,y) = |A_k(x+2^{k-1},y) - A_k(x-2^{k-1},y)| \tag{2.3}$$

$$D_{k,vertical}(x,y) = |A_k(x,y+2^{k-1}) - A_k(x,y-2^{k-1})| \tag{2.4}$$

- Step three: For each point, $S_{best} = 2^k$ is chosen such that D_k is the highest.

- Step four: Finally the coarseness measure F_{crs} is calculated as the average of $S_{best}(x, y)$ for all x and y of the image.

Contrast: Tamura et al. (1978) describes four factors to take into considerations for any measure of contrast, which are:

1. The range of grey-levels.

2. The polarization of the distribution of black and white on the grey level histogram.

3. Sharpness of edges.

4. Period of repeating patterns.

σ or σ^2 can measure the dispersion of the gray level values, being a good indication of factor one and also to some extent factor two. But they propose the kurtosis of the distribution, as shown in Equation 2.5, as a better measure for factor two:

$$\alpha_4 = \frac{\mu}{\sigma^4} \tag{2.5}$$

Tamura et al. (1978) propose the following measure which takes into consideration factors one and two.

$$F_{con} = \frac{\sigma}{(\alpha_4)^n} \tag{2.6}$$

Experimentally the authors arrived upon $n = \frac{1}{4}$ to give the best result and did not make any changes to incorporate factors three and four.

Directionality: An orientation histogram calculated over an image region (or the whole image) is used to extract the directionality feature. The magnitude $|\Delta G|$ and local orientation θ is approximated as follows

$$|\Delta G| = \frac{|\Delta_H| + |\Delta_V|}{2} \tag{2.7}$$

$$\theta = tan^{-1}\left(\frac{\Delta_V}{\Delta_H} + \frac{\pi}{2}\right) \tag{2.8}$$

where Δ_H and Δ_V are the horizontal and vertical differences measured using the following 3×3 matrices.

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{pmatrix}$$

A threshold t is determined and every point with magnitude $|\Delta G|$ above t is counted in a histogram H_D obtained by quantising the orientation θ.

$$H_D(k) = \frac{N_\theta(k)}{\sum_{i=0}^{n-1} N_\theta(i)}, \qquad k = 0, 1, ...n-1 \tag{2.9}$$

Where $N_\theta(k)$ is the number of points at which $\frac{(2k-1)\pi}{2n} \leq \theta < \frac{2k+1}{2n}$. The sharpness of the peaks of the histogram H_D can be used as a measure of the directionality. The approach taken by Tamura et. al. is to sum the second moments around each peak from valley to valley using Equation 2.10.

$$F_{dir} = 1 - r.n_p \sum_{p}^{n_p} \sum_{\phi \epsilon w_p} (\phi - \phi_p)^2 . H_D(\phi) \tag{2.10}$$

where

- n_p: number of peaks
- ϕ_p: pth peak position of H_D
- w_p: range of pth between valleys,
- r: normalising factor related to quantizing levels of ϕ
- ϕ: quantized direction code

Line-likeness: The line-likeness feature is an attempt to describe a texture element that is composed of lines. For this, when the direction of an edge and the direction of a neighbouring edge (by a distance d) are nearly equal, they may be regarded as one line.

In reality Tamura et al. (1978) first construct a direction co-occurence matrix $P_{Dd}(i, j)$ where i and j are direction codes. $P_{Dd}(i, j)$ is the frequency at which edges with direction i and j occur at neighbouring cells separated by a distance d along the edge direction of i. Trivial edges are disregarded by using a threshold t. The following is used to give a quantitative value of line-likeness. It gives a weight of +1 for directions codes of the same direction and −1 for perpendicular combinations.

$$F_{lin} = \frac{\sum_i^n \sum_j^n P_{Dd}(i, j) cos\left((i - j)\frac{2\pi}{n}\right)}{\sum_i^n \sum_j^n P_{Dd}(i, j)} \tag{2.11}$$

Regularity: A texture is considered to be irregular if it varies over the whole image. Using the previous four features Tamura et al. (1978) devised the following measure for regularity

$$F_{reg} = 1 - r(\sigma_{crs} + \sigma_{con} + \sigma_{dir} + \sigma_{lin}) \tag{2.12}$$

Where σ_{xyz} is the standard deviation of feature F_{xyz} and r is a normalising factor.

Roughness: Tamura et al. (1978) put an emphasis on coarseness and contrast when characterising roughness. This is based on their knowledge of human perception attained from the psychological experiments conducted by them. It is defined as follows.

$$F_{rgh} = F_{crs} + F_{con} \tag{2.13}$$

The original paper used only the first four features, F_{crs}, F_{con}, F_{dir}, F_{lin}, and two distance metrics, the Mahalanobis distance and the Euclidean distance, to determine the nearest neighbour of texture images and compare the results with the human judgements. Prasad and Krishna (2011) uses three Tamura features — coarseness, contrast and directionality — on CT scan images and uses Euclidean distance as a similarity measurement. Patil and Talbar (2012) uses multiple texture features, including Tamura features, and compare their performance using multiple distance measures including Euclidean, Manhattan, Chi-squared, etc.

Grey Level Co-Occurence Matrix (GLCM)

One of the best known statistical methods for texture analysis is the use of Co-occurrence matrices. The grey levels of images are used to construct the grey level co-occurrence matrix (GLCM). This was proposed by Haralick et al. (1973) and has become a very popular technique. The basic principle of the technique is simple: for a given displacement $d = (dx, dy)$, each entry (i, j) in the co-occurrence matrix is the count of the number of occurrences of the pair of grey levels i and j at a distance d. The following simple example shown in Figure 2.4.1 involving a 4×4 image with 3 different grey values $0, 1$ and 2 may serve to explain GLCM better.

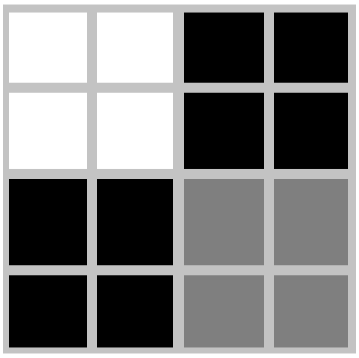


Figure 2.9: Example 4×4 image region on which a GLCM with $d = (1, 0)$ is being calculated. There are three grey scale values, here represented by white(1), black(0) and grey (2)

Given the image, and $d = (1, 0)$, the GLCM would be the following:

$$glcm_{(1,0)} = \begin{pmatrix} 4 & 0 & 2 \\ 2 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix}$$

The $(0, 0)$ element of the GLCM represents the number of times a grey level value of 0 follows (horizontally) a grey level value of 0. So, the $(0, 2)$ element of the GLCM has a value of 2 because it is only twice that a grey level value of 2 follows (horizontally) a grey level value of 0. Typically the choice of d is fixed for certain systems and for others it is hand-tuned on a case-by-case basis. Some studies use multiple values of d like Howarth et al. (2005), where multiple GLCM are extracted for $d = 1, 2, 3$, and 4 and four equally spaced orientations between 0 and π. Vimina and Jacob (2012) uses a single value of d (1) with the same four orientations.

A total of 14 texture features have been proposed by Haralick et al. (1973) using the GLCM. These include entropy, contrast, correlation, etc. La Cascia and Ardizzone (1996) describe a prototypal content-based video search system which makes use of grey-level co-occurrence matrices to extract texture features. A video sequence is split up into shots, and representative frames are chosen. These frames are used as the source to extract colour and texture features.

In the context of domain-specific CBIR, there are a few issues with the co-occurrence matrix. Haralick et al. (1973) does not seem to provide any information about how to choose d, although the coarseness measure proposed by Tamura et al. (1978) may be of some value. The usefulness of the information stored in the matrix when comparing with a query image would hinge on the correct (or as close to correct as possible) choice for the value of d and the orientations. If the image database has images which have textural properties spread over various scales, multiple values of d would be more suitable, however which combinations will work best is still an issue that need to be addressed through trial and error.

Local Binary Patterns

Local Binary Patterns (LBP) were first proposed in Ojala et al. (1996). A variation was then presented in Ojala, Pietikainen and Maenpaa (2002). Since then it has become a widely used texture features and has gained wide acceptance due to good performance in different image processing applications and also through simplicity of calculation. Here we will discuss the version presented in Ojala, Pietikainen and Maenpaa (2002).

LBP features are calculated over a circular neighbourhood around a central pixel. The size of the number of pixels in the neighbourhood (P) and its radius (R) can be varied. For this discussion we will assume that the images are grey scale.

The authors begin by defining a texture T in a local neighbourhood of an image as the joint distribution of the gray scale pixel values of $P, (P > 1)$ pixels

$$T = t(g_c, g_0, ...g_{P-1}) \tag{2.14}$$

Where g_c is the grey level value of the central pixel in the neighbourhood and $g_p(p = 0, ..., P-1)$ are the grey level values of P equally spaced pixels at a distance of radius R. This forms the circular neighbourhood. The grey level values which do not correspond to an actual pixel can be found by interpolation. To achieve invariance to luminance the value of the center pixel g_c is subtracted from the neighbourhood pixels G_p, giving

$$T = t(g_c, g_0 - g_c, ...g_{P-1} - g_c) \tag{2.15}$$

Assuming that the differences $g_p - g_c$ is independent of g_c, Equation 2.15 can be refactored to

$$T \approx t(g_c)t(g_0 - g_c, ...g_{P-1} - g_c) \tag{2.16}$$

As $t(g_c)$ here describes the overall luminance in the image, and as LBP features want to gain invariance to luminance, the g_c components is dropped and the texture of the neighbourhood is approximated by

$$T \approx t(g_0 - g_c, ...g_{P-1} - g_c) \tag{2.17}$$

This data can be used to produce a P dimensional histogram which can be used as a feature. But to achieve invariance to scaling of the grey scale values just the sign of the differences are considered.

$$T \approx t(s(g_0 - g_c), ...s(g_{P-1} - g_c)) \tag{2.18}$$

where

$$s(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.19}$$

Incorporating a binomial factor operator 2^p for each sign, Equation 2.18 can be converted to a number. This number can be used as a representation of local texture. Ojala, Pietikainen and Maenpaa (2002) and other works have referred to this in terms of P and R as $LBP_{P,R}$.

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{2.20}$$

This process is illustrated in Figure 2.4.1, showing how the LBP pattern can be calculated for an image region with the following grey scale values.

30	25	10
85	50	120
70	60	100

The calculation is performed around the central pixel, with a grey scale value of 50, with $R = 1$ and $P = 8$. Three of the eight neighbours have a pixel value less than 50, resulting in a corresponding 0 in the pattern entry. The others get a 1 in the pattern entry as their pixel value is larger than that of the central pixel. The order of how the pattern is read is immaterial as long as it is read consistently. In this example, we can start reading from the top right in a clockwise manner to end up with 01111100 as the pattern with $LBP_{P,R} = 124$ using Equation 2.20.

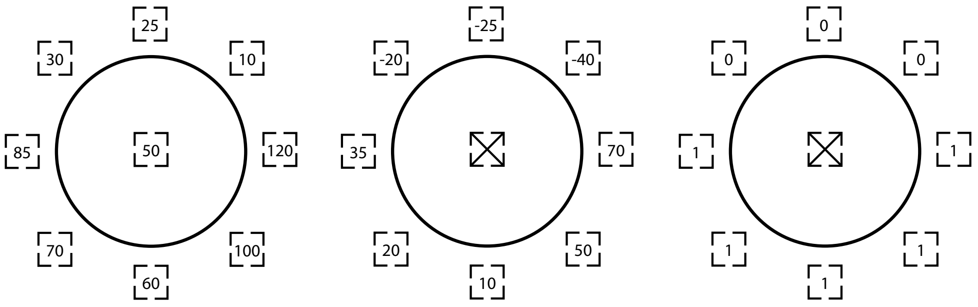


Figure 2.10: Illustration of the LBP calculation.

An extension to this is presented by the authors based on the observation that certain binary patterns occur frequently in texture images. They demonstrate this using image statistics gathered from their experimental images. They named these patterns "uniform" as they had a small number of spatial transitions in their circular neighbourhood. These can be thought of as templates which can help identify microstructures in an image such as bright spots, dark spots and edges. The following measure is used to measure the uniformity of a regions.

$$U(LBP_{P,R}) = |s(g_{P-1} - g_c)| + \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_{p-1} - g_c)| \tag{2.21}$$

In a scheme which is called LBP^{u2}_{PR}, patterns which have a U value of less than or equal to 2 are placed in a bin corresponding to their number. Other patterns are all placed in one bin. So, for a an $LBP_{P,R}$ value which has 8 bits, there are 58 uniform patterns, resulting in a histogram of 59 bins when using LBP^{u2}_{PR} .

$LBP_{P,R}$ is easily transformed into its rotation invariant version $LBP^{ri}_{P,R}$ by rotating the pattern to maximise the number of 0s in the most significant bits starting from g_{p-1}. So, the pattern 01111100 found from the illustration in Figure 2.4.1 would become 00011111. Using equation 2.22 each uniform $LBP^{riu2}_{P,R}$ pattern gets a unique label assigned to it, which depends on the number of 1s in the pattern. Every non-uniform pattern gets the label $P + 1$. This scheme produces $P + 2$ unique values. The image feature is calculated as a histogram of $LBP^{riu2}_{P,R}$ values, summed over a texture region.

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & if\ U(LBP_{P,R}) \leq 2 \\ P + 1 & otherwise \end{cases} \tag{2.22}$$

LBP is a simple and effective feature with excellent results in many applications. However, just like the GLCMs, the effectiveness of an LBP feature is dependent on the choice of the parameters dictating the window in which the pixel statistics are gathered. In this case the choices of R and P are crucial. Also, there is a choice between using the $LBP_{P,R}, LBP^{u2}_{P,R}$ and $LBP^{riu2}_{P,R}$. This needs to be determined experimentally. For example although $LBP^{riu2}_{P,R}$ seems be the most advantageous feature, Doshi and Schaefer (2012) found that the best performing feature for a texture database was one which is not rotation invariant. Their best performing results use a combination of LBP features extracted at multiple P, R. The issue is that the combinations need to be found through experimentation and are essentially hand-picked. In this thesis we aim to use techniques that will automatically give us feature extractors that are adapted to the image collection, without having to tweak them by hand as with LBP. This is explored further in chapter 7.

2.4.2 Structural Models

Some authors place this category of texture features as a sub-category of the statistical models (Zhang, Islam and Lu, 2012). It is also entirely possible to place the texton dictionary based features described below as spectral features. In general the techniques which fall under this category define texture to be composed of primitives or texture elements (texel) or textons (Tuceryan and Jain, 1998) and utilize textons and their placements to describe a texture (Materka et al., 1998; Fu and Albus, 1982; Leung and Malik, 2001).

Identifying texels/textons is one of the challenges of this field. One of the techniques filters an image with a Laplacian of Gaussian (LoG) filter at different scales (Voorhees and Poggio, 1987). The information gained from the filtering is combined to extract "blobs" from the image which were thought to be important for texture perception. Tüceryan and Jain (1990) present the use of Voronoi tessellation to determine textons from an image. They also use a LoG filter to extract textons. The local maxima of the filtered image are used to do a connected component analysis using 8 nearest neighbours.

Leung and Malik (2001) have done some interesting work on $3D$ textons. While the $3D$ aspect is not relevant to what we want to pursue in this research, their initial presentation of $2D$ textons has potential to be applied for adaptive texture features. They use the filters shown in Figure 2.4.2, now known as the Leung-Malik (LM) filter bank. There are 36 elongated filters at 6 orientations, 3 scales and 2 phases. The rest are made up of 4 low pass Gaussian filters and 8 center surround difference of Gaussian filters.

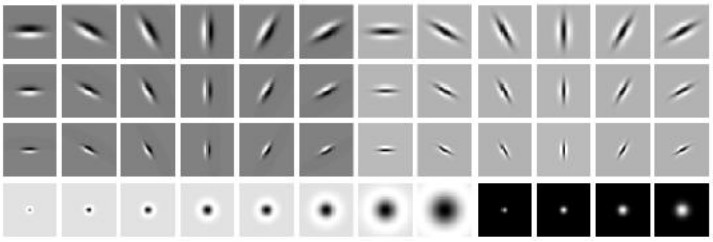


Figure 2.11: The Leung-Malik (LM) Filter Bank used to extract Textons in Leung and Malik (2001)

Similarly Schmid (2001) use Schmid(S) filters found from Equation 2.23 using 13 combinations of σ, τ.

$$F(r,\sigma,\tau) = F_0(\sigma,\tau) + cos\left(\frac{\pi\tau r}{\sigma})\right) e^{-\frac{r^2}{2\sigma^2}} \tag{2.23}$$

To extract textons Varma and Zisserman (2005) use a bank of 38 filters knows as the Maximum Response 8 filters, Ahonen and Pietikäinen (2009) uses a bank of 24 Gabor filters and also the local derivative (LD) filter set, Haddad (2005) uses ring filters and wedge shaped orientation filters in combination.

Each image is filtered with filters such as these. The responses of each filter at every pixel of an image is assembled in an n dimensional vector, where n is the total number of effective filters. These vectors are quantized using a K-means implementation into K centers. These K centers are then treated as the textons.

When using textons in a textons-based dictionary representation for images, the local descriptors of images are obtained through filtering (as above) or other techniques. Descriptors of images from each category are used to extract T textons using K-means as described above. If there are C categories, this gives a total of $T \times C$ textons to be used in a texton dictionary.

When creating an image signature the local descriptors are again found for the image and then each local descriptor is labelled by finding its closest texton using the Euclidean distance. The count of label occurrence is used to create a frequency histogram which is used as the image signature. This method is very similar to the “Bag-of-Visual-Words” representation evaluated in Yang et al. (2007).

Texton-based dictionaries have often been used in image classification using the nearest neighbour classifier or support vector machines etc. The distance between two representations can be calculated in many ways, but the Chi-square distance is quite popular (Zhang, Zhao and Liang, 2012). This is shown in Equation 2.24, where V_p and V_q are feature vectors and i is the index of the feature vector.

$$\chi^2(V_p, V_q) = \sum_i \frac{(V_p(i) - V_q(i))^2}{V_p(i) + V_q(i)} \tag{2.24}$$

Textons are particularly suited for collection-specific texture features. The issue however is how to decide which filters/methods to use to extract the local descriptors. We have only mentioned a few of the filter sets used so far to create a texton dictionary. When working with a collection it is possible to arrive at a filter set through experimentation, which would give a good texton dictionary for a collection. In this thesis we are interested in techniques which would allow us to bypass this experimentation/hand-tuning phase.

2.4.3 Spectral Models

Research has shown that the human brain performs a frequency analysis of images (Campbell and Robson, 1968), and that this kind of analysis suits texture very well. Signal processing techniques have been used to analyse texture and extract textural features because of this affinity.

Fourier Transforms (Lee and Chen, 2005; Hervé and Boujemaa, 2007), discrete cosine transforms (Lu et al., 2006), wavelets (Wang et al., 2001; Fan et al., 2004; Park et al., 2004) and Gabor filters (Tuceryan and Jain, 1998; Salembier et al., 2002; Manjunath and Ma, 1996; Zhang et al., 2005, 2000) are common methods employed to obtain texture features in this category. Fourier transforms and discrete cosine transforms are efficient to calculate but have the disadvantage of not being invariant to scale and rotation. Wavelets are limited in their orientations. Gabor-based features have the ability to capture texture patterns at multiple scales and orientations and are considered to be more robust (Zhang, Islam and Lu, 2012).

Gabor transforms are related to Fourier transforms. Fourier transforms carry out the analysis globally, but in applications where localised information is required a method called a windowed Fourier transform is used (Tuceryan and Jain, 1998). Equation 2.25 shows a one dimensional windowed Fourier transform.

$$F_w(u, \xi) = \int_{-\infty}^{\infty} f(x) w(x - \xi) e^{-j2\pi ux} dx. \tag{2.25}$$

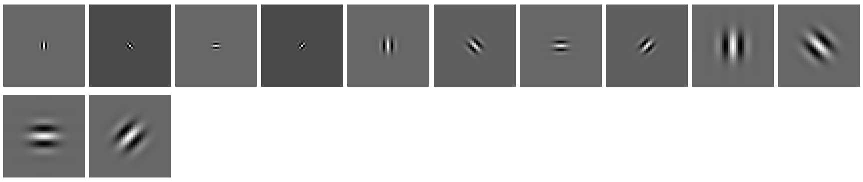


Figure 2.12: The 12 Gabor filters used by the GIFT/Viper CBIRS which gives good coverage.

When the window function $w(x)$ is a Gaussian, the resulting transform becomes a Gabor transform. Gabor filters have been proposed as a means of modelling the simple cells in the visual cortex and motivating their use in texture analysis (van Hateren and van der Schaaf, 1998; Daugman, 1980), and have been used in many image processing applications including CBIR (Squire et al., 1999), image classification (Liao et al., 2009) and others.

The GNU Image Finding Tool (GIFT) is an open source CBIRS (Squire et al., 1999). It uses a bank of 12 Gabor filters defined in the spatial domain by,

$$Gabor_{mn}(x,y) = \frac{1}{2\pi\sigma_m^2} e^{-\frac{x^2+y^2}{2\sigma_m^2}} \cos(2\pi(u_{0_m} x\cos\theta_n + u_{0_m} y\sin\theta_n)) \tag{2.26}$$

where m and n index the scale and orientation of the filters respectively. u_0 specifies the centre frequency and the half peak radial bandwidth is given by

$$B_r = \log_2\left(\frac{2\pi\sigma_m u_{0_m} + (2\ln 2)^{\frac{1}{2}}}{2\pi\sigma_m u_{0_m} - (2\ln 2)^{\frac{1}{2}}}\right) \tag{2.27}$$

Setting B_r to 1, gives σ_m to be

$$\sigma_m = \frac{3(2\ln 2)^{\frac{1}{2}}}{2\pi\sigma_m u_{0_m}} \tag{2.28}$$

the highest centre frequency is

$$u_{0_1} = \frac{0.5}{1+\tan(\frac{1}{3})} \tag{2.29}$$

For each change of scale, u_{0_m} is halved, doubling σ_m. Filters at three scales are used in GIFT. The orientations are acquired in $\frac{\pi}{4}$ steps, giving a total of 12 filters in three scales and four orientations which gave a good coverage without overlapping filters (Squire et al., 1999). These filters are reproduced and shown in Figure 2.4.3.

GIFT extracts two kinds of texture features. The global texture feature is just the mean of the filter response energy. The local features are described below.

For each filter, GIFT finds the filter response energy and divides the energy images into 16 × 16 blocks. The mean energy in each block is calculated and quantized into 10 bins. The bin centres were chosen after examining the filter response of 500 images. Unlike other systems, in GIFT every bin in the histograms are mapped to a feature.

In the context of adaptive texture features, using pre-defined Gabor filters has some drawbacks. There may be some textures to which all the filters in the bank may be blind. As like other techniques mentioned previously, it might be possible to hand-tune the filters. As we show in later chapters, it is not effective to just have filters in multiple scales and many orientations. In fact just blindly using multiple scales can lead to decreased performance as we will show in Chapter 3.

GIFT is a versatile CBIRS and in an evaluation done by Kosch and Maier (2010) it performed best among seven CBIR systems when they were tested on six different image collections representing images coming from different domains. Even so, the fact that it uses fixed pre-defined feature extractors is a weakness which was recognised by Müller et al. (2003), where they had to hand-tune the Gabor filters used for texture features to adapt to a collection of medical images.

In this thesis we will describe methods which we show perform better than banks of Gabor filters. These feature extractors are adapted to image collections without any human intervention. The process we will describe can easily be incorporated in CBIR systems like GIFT which would help bypass the type of hand-tuning done by Müller et al. (2003).

2.4.4 Other Gradient Based Methods

We present scale invariant feature transform (SIFT), histogram of gradients (HOG) and pyramid histogram of oriented gradients (PHOG) features here as a separate category as an important part of these three features is calculating gradients at image locations. However these three features could have easily been included in the category of statistical models. We first present SIFT followed by HOG and then briefly PHOG.

Scale Invariant Feature Transform

Scale Invariant Feature Transform (SIFT) was first proposed in Lowe (1999) and further explained in Lowe (2004). It has since become one of the most widely used features in object recognition (Pinto et al., 2011), image classification (Xiao et al., 2010) , content based image retrieval (Wangming et al., 2008) and many other domains. Extracting SIFT features from an image is a four-step process as described in Lowe (2004):

- Detecting keypoints
- Identifying stable keypoints
- Assigning keypoint orientation
- Creating the keypoint descriptor

We describe each step in more detail below.

Detecting keypoints: Here we discuss the keypoint detection mechanism proposed in Lowe (2004). This is by no means the only way to identify keypoints in an image. The following steps can be applied after detecting a keypoint using other methods, such as those described in Mikolajczyk and Schmid (2004), e.g. Harris corner detector.

Searching through the scale-space (Witkin, 1983) of an image, SIFT tries to find locations with properties (usually calculated from a region around the location) which remain invariant under changes of scale and viewing angle. In Lowe (2004), the method described convolves a variable-scale Gaussian function $G(x, y, \sigma)$ with an image $I(x, y)$ to produce a scale-space $L(x, y, \sigma)$ (Equation 2.30) which can then be used to calculate the Difference-of-Gaussian(DoG) function (Equation 2.31).

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.30}$$

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{2.31}$$

Lowe (2004) outlines an efficient method of calculating $D(x, y, \sigma)$ by incrementally convolving the image with Gaussians to produce images which would be separated by some factor k in scale-space. The DoGs are produced by subtracting adjacent images. The next octave can be initiated by taking the image which has twice the value of the original σ and re-sampling it to reduce the size by half. This process is depicted in Figure 2.13

The extrema in the DoGs are considered to be candidate keypoints. Every pixel value in a DoG image is compared to its eight neighbours as well as its nine neighbours from each of the adjacent scales. So, for each location there are 26 comparisons and a location is considered to be an extremum only if its pixel value is found to be the highest or the lowest. If an octave is divided in s intervals, the SIFT keypoint detection process generates $s + 3$ scales to ensure that DoG extrema can be detected from the whole octave. These candidate keypoints are further refined to select those that are stable as described next.

Identifying stable keypoints: A keypoint location is refined by fitting a $3D$ quadratic function to the local sample points. The Taylor expansion of $D(x, y, \sigma)$ shifted so that the keypoint is at the origin (Equation 2.32) is used to take the derivative of the function with respect to $\mathbf{x}$ ($\mathbf{x} = (x, y, \sigma)^T$). This is set to zero, and solved for $\mathbf{x}$. The solution is used to refine keypoint location and scale.

$$D(\mathbf{x}) = D + \frac{\delta D^T}{\delta \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\delta^2 D}{\delta \mathbf{x}^2}\mathbf{x} \tag{2.32}$$

As the DoG method is susceptible to finding keypoints lying on edges, the keypoints are further refined to eliminate those lying on an edge. This can become a problem when trying to use SIFT as a texture feature as texture elements made up of lines are well known and features like line-likeness described previously are specifically designed

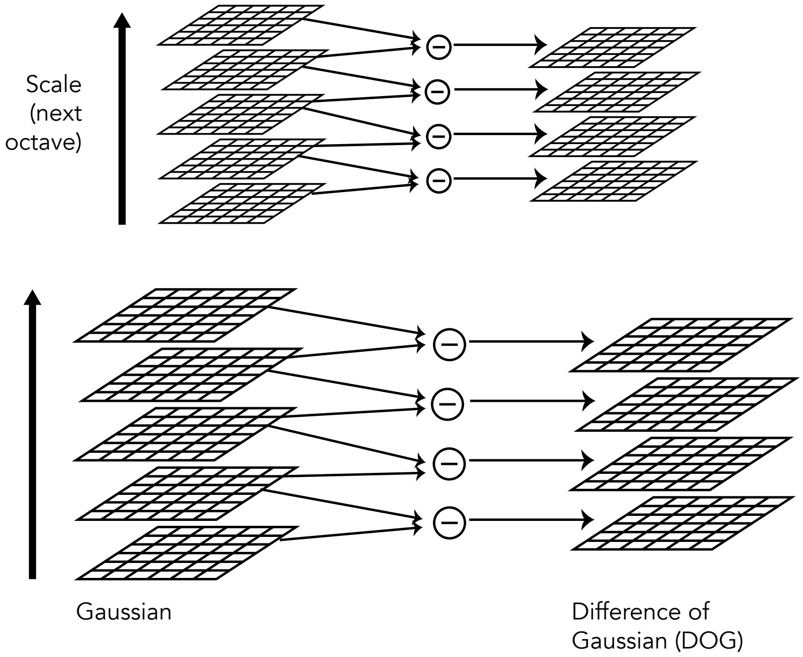


Figure 2.13: DoG calculation outlined in Lowe (2004)

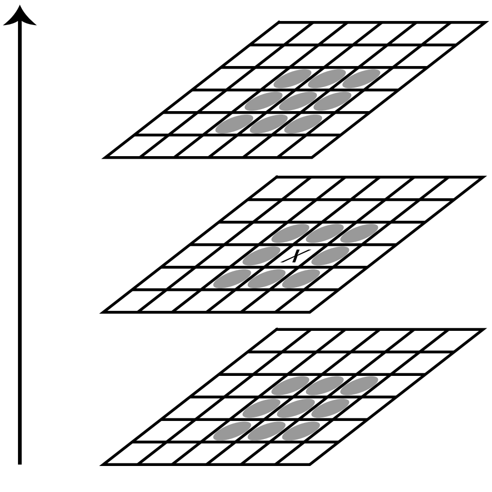


Figure 2.14: Comparing a point (marked with "x") with its 26 neighbours in from its own and its neighbouring scales

to capture them better. Figure 2.4.4 shows a pattern made up of lines where no SIFT keypoints were detected.

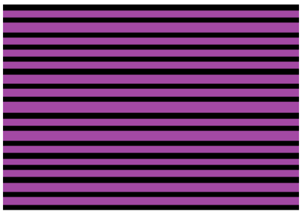


Figure 2.15: Patterned image made up of lines where no SIFT keypoints were detected.

Assigning keypoint orientation: SIFT ascribes a consistent orientation to each keypoint. It then represents the keypoint descriptor relative to this orientation, thus achieving rotation invariance. Orientation is assigned to the keypoint as follows.

The Gaussian-smoothed image L with the closest scale to the scale of the keypoint is used to perform all operations. This ensures scale invariance. For each (x, y) in a region around the keypoint at this scale, the gradient magnitude $m(x, y)$ and orientation $\theta(x, y)$ are calculated, as shown in Equations 2.33 and 2.34:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \tag{2.33}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right) \tag{2.34}$$

The magnitudes of the orientation gradients are multiplied by a Gaussian window. An orientation histogram is created and the magnitudes are accumulated in locations corresponding to the orientation values. The peak of the histogram is used as the orientation of the keypoint. All other orientations are then normalised relative to the keypoint orientation.

Creating the keypoint descriptor: The SIFT keypoint descriptor is created by assigning an orientation (normalised by the keypoint orientation) and a gradient magnitude to every sample point in a fixed region around the keypoint. To avoid abrupt changes in the descriptor these are weighted by a Gaussian window with σ one half the width of the descriptor region. Then the keypoint region is divided in 4×4 subregions. The contents of the subregions are represented using an orientation histogram where the magnitudes at each orientation bin are summed. Trilinear interpolation is used to distribute the value of each gradient sample into neighbouring histogram entries. In the experiments conducted in Lowe (2004), orientation histograms with 8 bins are used. The orientation histograms are concatenated over the 4×4 region giving a 128-dimensional SIFT descriptor.

SIFT-based features: There are quite a few variations of SIFT and several features based on it. Some of the more popular features are Dense SIFT (DSIFT) (Lazebnik et al., 2006) and Pyramid Histogram of Words (PHOW) (Bosch, Zisserman and Muoz, 2007). DSIFT, as the name suggests, is a dense feature where SIFT descriptors are calculated on the same scale and orientation at specified sample steps in the image, rather than at identified keypoints. PHOW descriptors are essentialy DSIFT done at multiple scales.

Other variations include PCA-SIFT (Ke and Sukthankar, 2004), which pre-computes an eigenspace from training patches around keypoints (extracted at the appropriate scale and rotated to align the dominant orientation). When extracting features, patches around keypoints are projected to this eigenspace. PCA-SIFT usually gives more compact descriptors but in the evaluation done by Mikolajczyk and Schmid (2005) it had lower performance compared to SIFT features.

Most of the above are usually applied on grey-scale images. Verma et al. (2010) proposes a colour SIFT descriptor where a SIFT descriptor is calculated on each of the three colour channels giving a combined SIFT descriptor of 384 dimensions, which shows improvement over grey-scale SIFT descriptors.

Histogram of Oriented Gradients

Histogram of Oriented Graidents (HOG) is a very popular feature initally used for Human detection but has been widely used in other fields (Dalal and Triggs, 2005). It utilises image gradient information extracted from images after subdividing the image into small regions which are referred to as "cells". The gradients at sampled pixels within a cell are used to compute a histogram. The combined histogram of all the cells becomes the image descriptor. We will discuss the technique in more detail below. The discussion will be based on the work presented by Dalal and Triggs (2005).

Gradient computation Dalal and Triggs (2005) reports the use of different techniques to compute the gradient at a pixel, however through experimentation they have found that using the simple masks $[-1, 0, 1]$ and $[-1, 0, 1]^T$ on a non-smoothed image works best. When dealing with colour images their proposed method is to calculate the gradients on each of the colour channels and choose the one with the largest norm as the pixel's gradient vector.

Orientation bin assignment An orientation histogram is calculated for every cell. In their work a cell is a block of 8×8 pixels. The orientation bins of the histogram are evenly spaced between $0° - 180°$ or $0° - 360°$. The gradient magnitude at each pixel is used as a weighted vote for this histogram, which is distributed using bilinear interpolation between neighbouring bin centres. Other schemes were tested including using the square of the magnitude, its square root or a thresholded value indicating presence or absence. The paper reports that the best combination for its experiments was obtained when using the gradient magnitude in a histogram of 9 bins evenly spaced between $0° - 180°$.

Descriptor blocks and normalisation To combat the issue of local intensity differences within an image, and also the issue of foreground-background contrast, the authors propose multiple strategies to do local normalisation. Most of them group multiple neighbouring cells into what they call "blocks" and normalise each block separately. They use a strategy of using overlapping blocks so that each cell response actually contributes to multiple entries of the final descriptor. They experimented with four normalisation schemes, namely L2-norm (Eq 2.35), L2-Hys (L2 normal followed by thresholding), L1-norm (Eq. 2.36) and L1-sqrt (Eq. 2.37). These are shown in equations 2.35 - 2.37, where $\overrightarrow{v}$ is the un-normalised block descriptor, $||\overrightarrow{v}||_k$ is the $k-$ norm and ϵ is a small constant. Of these L2-norm, L2-hys and L1-sqrt performed comparably and best in their experiments.

$$v_{normalised} \leftarrow \frac{v}{\sqrt{||v||_2^2 + \epsilon^2}} \tag{2.35}$$

$$v_{normalised} \leftarrow \frac{v}{||v||_1 + \epsilon} \tag{2.36}$$

$$v_{normalised} \leftarrow \sqrt{\frac{v}{||v||_1 + \epsilon}} \tag{2.37}$$

Pyramid Histogram of Oriented Gradients The Pyramid Histogram of Gradients (Bosch, Zisserman and Munoz, 2007) is an extension of the HOG descriptor which combined HOG and image pyramids as described by Lazebnik et al. (2006). It was initally utilised for shape matching. Each image is divided into smaller regions successively. A histogram of image gradients is calculated for for each region (starting from the whole image to each subregion). In the method proposed in Bosch, Zisserman and Munoz (2007) the histogram represents the number of occurrences of an orientation rather than its magnitudes. The histograms of all the regions are concatenated to create the PHOG descriptor. This process is depicted in Figure 2.4.4.

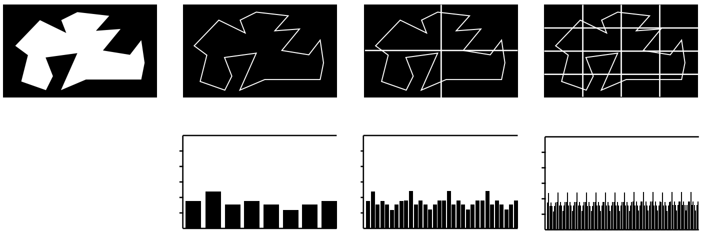


Figure 2.16: Dividing the image into sub-regions to calculate the PHOG descriptor.

SIFT and HOG as texture features

SIFT has been widely used for object recognition and HOG and PHOG have been successfully applied to human detection as well as other object recognition and classification systems. In this study we have used them to extract texture features, as at a basic level both these features calculate gradients within some image region. This we believe is not all that different from using a bank of Gabor filters on an image region. Some justification for this is presented in A.1. The main point related to this thesis is, the manner of feature extraction for both these features is fixed and pre-defined and does not automatically vary for different image collections.

It would be ideal if we could learn automatically details about feature extractors which would be adapted to the image collections. As will be discussed in the next section, ICA, a high order statistical technique, provides techniques necessary to extract filters from images which are adapted to the data (Le Borgne et al., 2004).

2.5 Independent Component Analysis (ICA)

Independent Component Analysis (ICA) is a statistical technique used to estimate underlying hidden factors from multivariate data. It uses the assumption that the underlying factors are statistically independent and non-Gaussian. The estimated factors, called independent components (IC), can be thought of as basis vectors whose weighted sum describes the data. Comon (1994) credits Herault and Jutten as the first to have addressed the problem, around 1983, and as having coined the term around 1986. The initial motivation behind ICA was to perform Blind Source Separation (BSS), which refers to the task of discovering the source signals from some observed linear mixture of the sources (Hyvarinen, Karhunen and Oja, 2001). In fact BSS is a good example to use to describe ICA as a mathematical problem. Here a simple version of ICA is presented, where we assume that the number of observed signals and the number of source signals are equal. Let $x_1(t)$, $x_2(t)$ and $x_3(t)$ represent the observed signals of some source signals $s_1(t)$, $s_2(t)$ and $s_3(t)$ at time t. Based on this information, it can be said that for $i = 1, 2, 3$

$$x_i(t) = a_{i1}s_1(t) + a_{i2}s_2(t) + a_{i3}s_3(t). \tag{2.38}$$

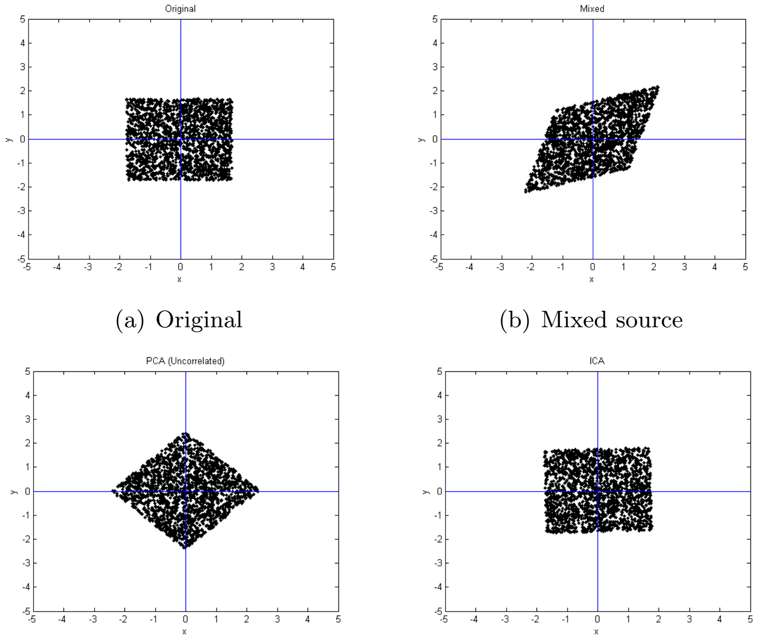
In this situation, the source signals $s_i(t)$ are unknown and so are the mixing weights a_{ij}. The only known values are the observed signals $x_i(t)$. The problem of BSS is to find the original signals $\{s_i(t)\}$) from the observed mixtures $\{x_i(t)\}$). The assumption is that the mixing coefficients a_{ij} are sufficiently different so that it would be possible to allow the inversion of the matrix formed from those values. The inference then is that there is a matrix W, with w_{ij} as coefficients, which would allow the separation of each s_i, as

$$s_i(t) = w_{i1}x_1(t) + w_{i2}x_2(t) + w_{i3}x_3(t). \tag{2.39}$$

Therefore the matrix A, formed by the mixing coefficients a_{ij}, is the inverse of W, $W = A^{-1}$. This is the basic mathematical problem that needs to be solved, and ICA provides a solution to this seemingly hard problem by the assuming that the signals are statistically independent (Comon, 1994). That is, the value of their joint probability is the product of their marginal probabilities. Independence is stronger then uncorrelatedness. Figures 2.17(a) shows the joint distribution of two random sources with zero mean and unit variance. Figure 2.17(b) shows the joint distribution of two mixed signals obtained by mixing the original sources by the following matrix:

$$\begin{pmatrix} 1 & 0.3 \\ 0.3 & 1 \end{pmatrix}$$

The data still is centered and Figure 2.17(c) shows the joint distribution of the whitened[1] mixture obtained through Principal Component Analysis. Whitening through PCA solves a large part of the ICA problem and is used quite frequently as a pre-processing step to ICA. Figure 2.17(d) shows the ICA transformation of the whitened mixture obtained by an implementation of the FastICA algorithm, which will be discussed later.



(a) Original

(b) Mixed source

(c) PCA

(d) ICA

Figure 2.17: Difference between ICA and PCA

The central limit theorem tells us that the distribution of a random variable which is the sum of non-Gaussian random variables will be distributed closer to a Gaussian compared to the original random variables. Therefore finding maxima in non-Gaussianity in a linear combination of the mixture variables ($y = \sum_i b_i x_i$) gives us the independent components (Hyvarinen, Karhunen and Oja, 2001). Two frequently used methods of estimating the non-Gaussianity of a sample from a random variable are Kurtosis and Negentropy. We discuss them further here.

2.5.1 Kurtosis

Kurtosis is the fourth-order cumulant and is defined by

$$kurt(y) = E\{y^4\} - 3(E\{y^2\})^2 \tag{2.40}$$

If y has unit variance $E\{y^2\}$ becomes 1, and Kurtosis can be simplified to $E\{y^4\} - 3$. For Gaussian variables $E\{y^4\}$ is equal to $3(E\{y^2\})^2$, so the kurtosis value in this case is zero. Kurtosis is positive and negative for super-Gaussian and sub-Gaussian random

[1] Whitening is an operation that transforms a set of random variables with a known covariance matrix to another set whose covariance matrix is the identity matrix, resulting in a data set which is uncorrelated and has a variance of 1.

variables respectively. For ICA estimation the absolute value of kurtosis is typically used. However it is not a very robust measure as it is very sensitive to outliers. From a sample of 1000 values from a whitened random variable, only one value equal to 10 is sufficient to make the kurtosis value at-least 7. Researchers have used negentropy as a more robust estimate of non-Gaussianity, which is described now.

2.5.2 Negentropy

Entropy is the fundamental concept in information theory and is a measure of the expected value of the information in a random variable. The differential entropy of a random variable is defined in Equation 2.41

$$H(Y) = -\int_Y p(y) log(p(y)) dy \tag{2.41}$$

where p is the probability density function of Y.

We know from information theory that a Gaussian variable has the largest entropy among all random variables of equal variance (Hyvarinen, Karhunen and Oja, 2001; Comon, 1994). This allows for entropy to be used as a measure of non-Gaussianity. Negentropy is just such a measure. It is zero for Gaussian variables and positive for non-Gaussian variables. Equation 2.42 defines the negentropy $J(Y)$ of a random variable Y where Y_{gauss} is a Gaussian variable with the same covariance matrix as Y:

$$J(Y) = H(Y_{gauss}) - H(Y). \tag{2.42}$$

Using negentropy as a measure of non-Gaussianity is justified by statistical theory and may be considered to be an optimal estimator of non-Gaussianity (Hyvarinen, Karhunen and Oja, 2001; Hvyarinen, 1998; Hyvarinen, 1999). Calculating it however requires an estimate of the probability density function (PDF), which makes is very difficult to compute. In practice some approximations of negentropy is used for ICA. We describe here the approximations developed by (Hvyarinen, 1998) which were shown to work better than other approaches, such as kurtosis or other cumulant based approaches by Amari et al. (1996); Comon (1994).

As a model, the approximations take the form in Equation 2.43

$$J(y_i) \approx c[E\{G(y_i)\} - E\{G(v)\}]^2 \tag{2.43}$$

where G can be any non-quadratic function, c a constant and v is a Gaussian variable of zero mean and unit variance (assuming, y_i is also zero mean and unit variance). If $G(y_i) = y_i^4$ then this model encapsulates the cumulant-based approximation shown in Comon (1994). The function G should be such that it should be locally consistent (Hyvarinen, 1999) and robust against outliers (Hampel et al., 2011). Using these Hvyarinen (1998) lists the following functions as suitable approximations of negentropy.

$$G_1(y) = \frac{1}{a_1} \log \cosh(a_1 y) \tag{2.44}$$

$$G_2(y) = -\frac{1}{a_2} \exp(\frac{-a_2 y^2}{2}) \tag{2.45}$$

$$G_3(y) = \frac{1}{4} y^4 \tag{2.46}$$

$$\tag{2.47}$$

where $1 \leq a_1 \leq 2$ and $a_2 \approx 1$ are constants.

2.5.3 FastICA

Having presented negentropy and the approximations used for its estimation, we now present a fixed point algorithm for ICA called FastICA proposed by Hyvärinen and Oja (1997). It has many desirable characteristics, including rapid convergence (Oja and Yuan, 2006) and simplicity of implementation. All experiments conducted in this research have been carried out using FastICA. The algorithm is as shown in Algorithm 1.

Algorithm 1 The FastICA algorithm

```
1: Center the data to make the mean zero.
2: Set *threshold* to some threshold value to determine convergence.
3: Whiten the data and let the whitened data be $z$.
4: Choose $m$, the number of ICs to be estimated.
5: $counter \leftarrow 1$.
6: Choose a random vector of unit norm for $w_{counter}$.
7: $temp \leftarrow w_{counter}$.
8: Update $w_{counter} = E\{zG(w_{counter}^T z)\} - E\{G'(w_{counter}^t z)\}w$.
9: Orthogonalize: $w_{counter} = w_{counter} - \sum_{j=1}^{counter-1}(w_{counter}^T w_j)w_j$
10: $w \leftarrow \frac{w}{||w||_2}$
11: **if** $||temp - w_{counter}||_2 > threshold$ *or* $||temp + w_{counter}||_2 > threshold$ **then**
12:     go back to step 7
13: **end if**
14: $counter \leftarrow counter + 1$
15: **if** $counter < m$ **then**
16:     go back to step 5
17: **end if**
```

Where G is one of the functions mentioned in Equations 2.44–2.46.

The algorithm presented here does deflationary orthogonalization, and is the one used in this research. For the symmetric version please refer to Hyvarinen, Karhunen and Oja (2001).

2.5.4 Limitations of ICA

The assumption that ICA operates under is that the underlying sources are non-Gaussian. Hence it uses maxima of non-Gaussianity as a tool to identify the independent components. This brings us to its first limitation: it does not work for Gaussian sources. Secondly unlike PCA, which orders the components by maximal variance, ICA does not give any information about the ordering of the extracted components. Neither does it retrieve the original sign or magnitude of the components. These restrictions are not prohibitive. The original direction and magnitude of the components is not important for image processing applications. An ordering of the components would have been useful. There has been some effort to estimate the relative utilities of the extracted components, when interpreted as filters. This will be presented in §2.5.6, when we discuss filter selection.

2.5.5 ICA in image processing

Barlow theorised that edge detectors in the human visual cortex are a result of a "redundancy reduction" process leading to an arrangement where the activation of feature detectors are as statistically independent from one another as possible (Barlow, 1989). Independent component analysis, as described before, is a method which can find components which are as statistically independent as possible. Bell and Sejnowski (1997) use this property of ICA to hypothesise that, if Barlow's reasoning is correct, ICA should extract localised edge detectors. In their experiments they used a training set of natural scene images which included trees, leaves etc. They took $17,595$, 12×12 patches from these images, made the patches zero mean and whitened them using a symmetrical whitening filter. These whitened patches were then unrolled into vectors and aggregated into a data matrix on which ICA was executed. Experiments were repeated multiple times to ensure that the results were not affected by the ICA initialisation conditions. The extracted independent components were transformed back into the image space and are re-formed into patches to form the filters. Figure 2.5.5 shows the 144 filters extracted from this process. Most of the filters are edge filters, localised and oriented matching some of the properties of the simple cell receptive fields in the visual cortex (Hubel and Wiesel, 1968). Their resemblance to the Gabor filters shown earlier is also noteworthy. Bell and Sejnowski (1997) also show that the ICA filters have the property of sparseness, which is each feature detector activating as rarely as possible.

van Hateren and van der Schaaf (1998) also used ICA to extract filters, calling them independent component filters (ICF), and compared them to the simple cells in the primary visual cortex. Their method is similar to Bell and Sejnowski (1997) except they used $120,000$ 18×18 patches and used PCA to reduce the dimensionality of the data matrix from 324 to 240. They report findings that by reducing the dimensionality by PCA they were able to extract only the oriented filters. In their comparison they found that while the ICF were localised and oriented, they have a fixed scale whereas the simple cells of the visual cortex have receptive fields which act on multiple scales. As far as ICF extraction goes, Hyvarinen, Karhunen and Oja (2001) describes a similar method but take $10,000$ random 16×16 patches. Each patch is whitened separately and dimensionality was reduced to 160. Similar methods are employed in many other studies (Le Borgne et al., 2004; Le Borgne and Guérin-Dugué, 2001) and is depicted in Figure 2.19.

Motivated by factors such as statistical independence, similarities with the simple cells of the visual cortex, similarities with Gabor filters etc., ICA has been used in many different

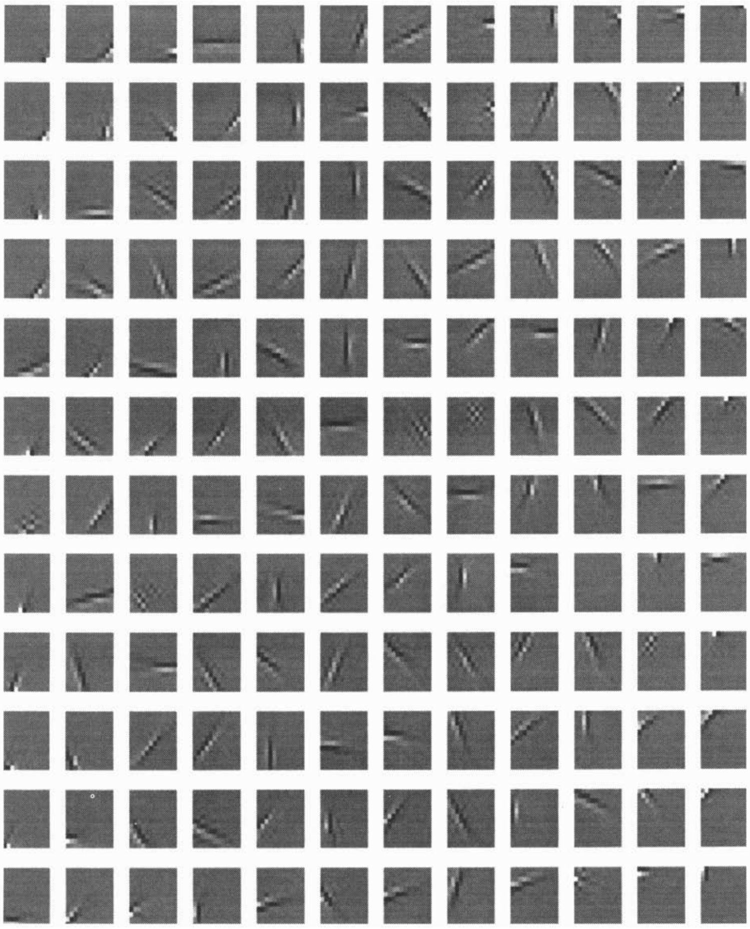


Figure 2.18: Example of filters extracted through ICA from images of natural scenes by Bell and Sejnowski (1997). The filters are localised and mostly oriented, properties in common with simple cell receptive fields

image processing applications, including image classification (Duan et al., 2009; Wu et al., 2012; Le Borgne et al., 2004), edge detection (Chen et al., 2002), blind deconvolution (Kaplan and Ulrych, 2003). It has also been used in certain medical imaging, e.g. breast cancer detection (Boquete et al., 2012), fMRI images (Calhoun et al., 2009) etc.

In terms of content-based image retrieval, Khaparde et al. (2008) compares ICs extracted from the query image and images in the database to determine the results. Sun et al. (2006) also uses ICA in conjunction with Generalized Gaussian Density for the purposes of CBIR. The results shown in the paper are very encouraging, however their method also suffers from the use of ICA in the image feature extraction process. Bai et al. (2007) uses Probabilistic ICA to extract image features and uses the z-values of ICs to find a component-wise similarity measure. Wang and Dai (2007) uses ICA features and other low-level features, along with a learning algorithm for image retrieval and they show very promising results. Li et al. (2009) employs ICA as a dimension reduction method. The study extracted eight different features from images and used ICA to extract the ICs of each feature.

Apart from Khaparde et al. (2008), these approaches do not utilise ICA components as filters. Khaparde et al. (2008) mentions the use of an ICF bank but does not clarify

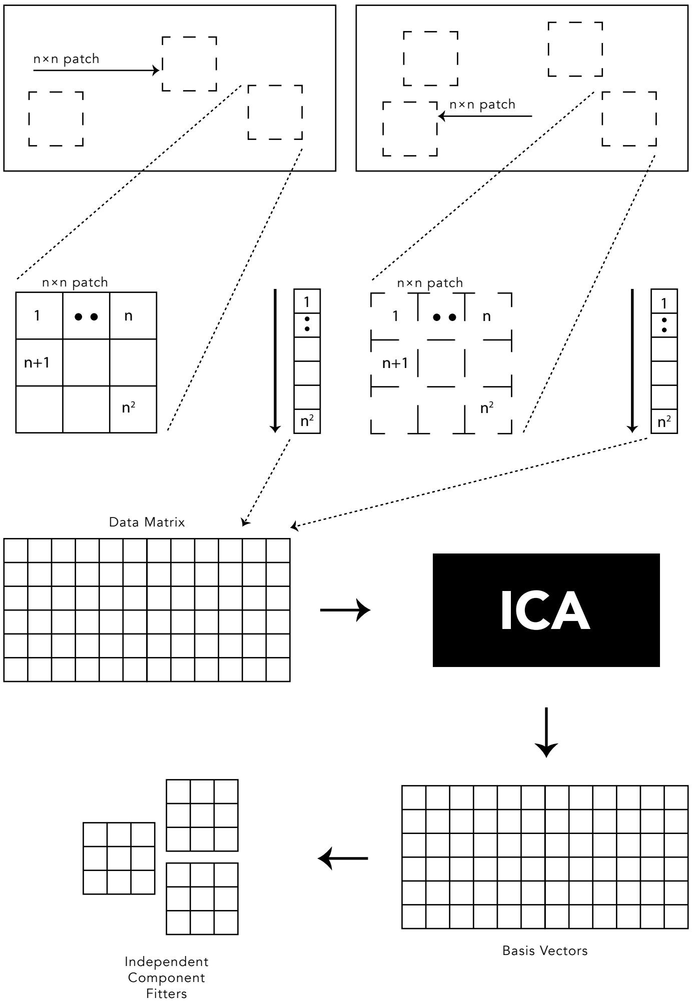


Figure 2.19: ICF extraction process following the methods of Le Borgne et al. (2004) and Le Borgne and Guérin-Dugué (2001)

how the filters were designed. It seems as if the process proposed extracts ICs from the query image, uses them as filters and collects filter responses from the database. If this is indeed the case, then it requires repetitive execution of ICA on the query image. This can be quite an expensive process. In a survey published in 2009, Vassilieva (2009) mentions that there have not been any studies comparing the performance of ICF with other texture feature extraction techniques in the retrieval space. Since then, apart from our work first published in 2011 (Mohammed and Squire, 2011a,b), there has been some use of ICF for image retrieval. Yarygina et al. (2011) employs ICF for their texture features in the context of CBIR. Unfortunately they do not provide much in the way of analysis of the performance of their ICF-based texture features as their work is not a proper evaluation of the performance of ICF but rather it concentrates on mechanisms to handle complex queries. In Khaparde et al. (2011) the authors compare the performance of ICF with

Gabor filters in a CBIR application. Their features are the mean and standard deviation of the filter response energies. However their results are harder to interpret as they use a non-standard database which has a mixture of textured images, flags, animals and trees in 18 categories, with 10 images per category for a database size of 180. Also when presenting their results they show statistics gathered from the first 32 images retrieved, when in each class they have only 10 images.

ICA can extract a large number of filters, e.g. 144 in Bell and Sejnowski (1997), 225 in Le Borgne et al. (2004) etc. and none of these address the problem of filter selection. For practical purposes using such a large number of filters is difficult. Methods need to be found which would reduce the number of independent component filters to give us the most important ones. There is also the issue pointed out by Trojan (2004) where it is shown that ICA can extract components which when re-formed into filters seem to be shifted/duplicate versions of each other. When used as filters these shifted/duplicate versions will give redundant features. None of the CBIR studies address this issue at all. In the image classfication space, Le Borgne et al. (2004) and Le Borgne and Guérin-Dugué (2001) used measurements of dispersal and sparseness to assign importance to each ICF and employed their selection strategy in an image classification context. There is however problems with this technique which will be discussed in more detail next in §2.5.6.

2.5.6 Filter selection from current literature

Variance based method

Le Borgne and Guérin-Dugué (2001) state that independent component filters have the desirable properties of sparseness and dispersal (Le Borgne and Guérin-Dugué, 2001). They used only dispersal to select filters, which was calculated as normalised variance. The idea is the variance of the response of a filter across the collection indicates how useful the filter is for discriminating between the images. Let X be a list of variances, where σ_i^2 is the variance of the responses of filter i . The list of dispersal values D is constructed from

$$D_i = \frac{\sigma_i^2}{\max(X)} \tag{2.48}$$

A subset of filters with the largest D_i is selected.

This method does not solve the problem of the shifted/near-duplicate components (Figure 2.20) mentioned in other works (Hyvärinen and Hoyer, 2000; Hyvärinen, Hoyer and Inki, 2001; Hyvärinen et al., 2000b; Hyvarinen et al., 2000a; Hyvärinen and Hoyer, 2001) and identified by Trojan (2004) as a potential problem when these components are used as filters for image feature extraction, as it will choose multiple shifted/duplicate filters so long as they have high variance, leading to redundant features.

Kurtosis of the distribution of components

Wu and Liu (2007) used ICA on Dynamic contrast-enhanced (DCE) imaging data to assess cerebral blood perfusion without the use of any prior information of underlying anatomical

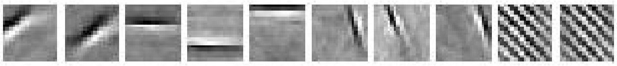


Figure 2.20: Some ICF which are seemingly shifted/duplicate versions of each other. When used as filters they would provide very similar features leading to redundancy.

information. In trying to ascertain the "meaningful" subset of the components, they used the kurtosis of the distribution of the components. Kurtosis values closer to zero would indicate that the distribution is closer to Gaussian and they deemed these components to be most likely to be noise. Arfanakis et al. (2002) and Formisano et al. (2002) use a similar criteria for choosing ICs from Diffusion tensor imaging data and fMRI data respectively. This method would also have the same problem as the variance-based method, as similar, shifted, filters will have similar Kurtosis values.

Both the variance-based and the kurtosis-based filter selection methods have the potential problem of selecting filters which would extract very similar features. We propose a solution to this problem in Chapter 4.

2.6 Content-Based Image Retrieval

Content Based Image Retrieval (CBIR) is, as the name suggests, a method to search for images. Unlike popular image search facilities like Google image search, where the search criteria is supplied by the user in a textual form, the common method used in CBIR is query by example, where the user provides an image as the search criteria. In its basic model, CBIR depends on image features extracted from the query image to perform the query. Some systems have used metadata to aid in this process. However, metadata associated with images are not always available and it is expensive to obtain annotations from humans. Even in cases where humans produce metadata, an image can mean different things to different people, so there is no guaranteed consistency. Also, there are some visual elements that are hard to describe in text, such as texture. In these cases it is easier to do a query using an example image. While using metadata/annotations associated with an image can certainly improve search results (Marques and Furht, 2002), it is not suitable to be the sole technique.

2.6.1 The semantic gap

As described in Müller (2001), and Marques and Furht (2002), and other sources, there are many difficulties which make the implementation of a universally optimal CBIR system (CBIRS) almost impossible. One of the main problems is the semantic gap, which, simplistically, could be defined as the difference between what a user perceives to be important in an image compared with the information that is extracted from image features for the same image. Compounding the problem is the fact that different users can have different judgements of what is important in an image. Indeed, the same user can have different opinions on the matter for different times, context, needs etc. (ten Brinke et al., 2006). There has been much research on trying to bridge this gap (Rho and Yeo, 2013;

Tang et al., 2012) including some which performs fMRI brain scanning on test subjects while they are exposed to multimedia content in order to understand what kind of features work best to minimise this gap (Hu et al., 2012), but a solution to this issue is still elusive.

2.6.2 CBIR architecture

Nevertheless the potential usefulness of CBIR motivates much research in the field, and any improvements are desirable. The areas in which improvements can be made are numerous, as can be seen from Figure 2.21, which depicts a typical architecture of a CBIRS.

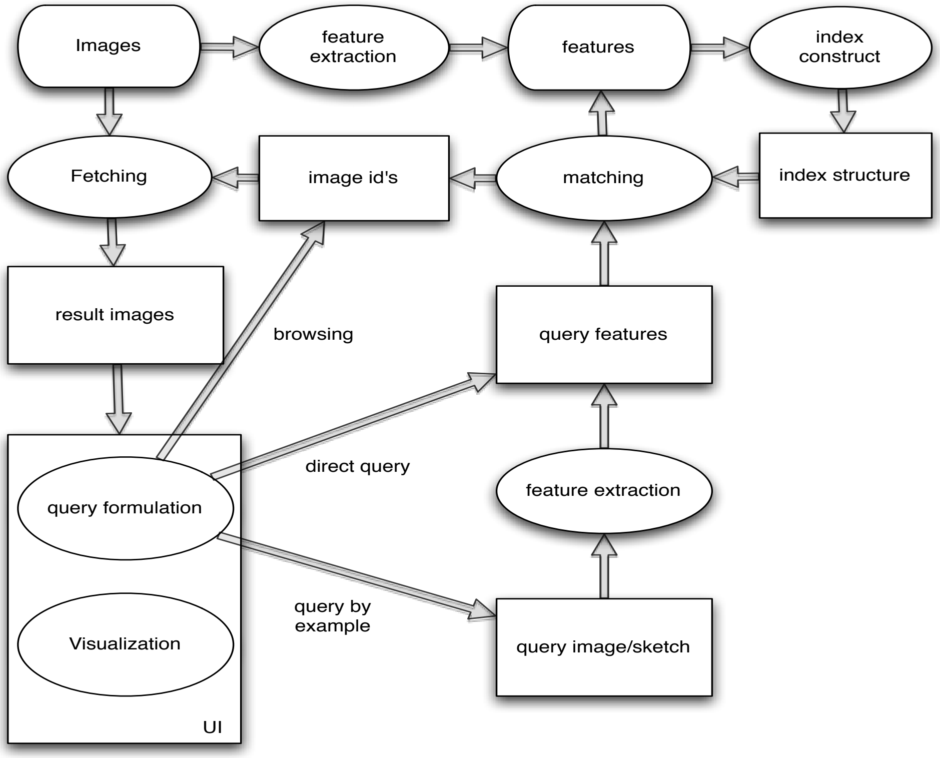


Figure 2.21: Architecture of a CBIRS, adapted from Veltkamp et al. (2001)

There are a wide range of CBIR systems, e.g. Caliph & Emir (Lux, 2009), PictureFinder (Schober et al., 2005), SIMPLIcity (Wang et al., 2001), VisualSEEk (Smith and Chang, 1997), ImageRover (Sclaroff et al., 1997), MARS (Ortega et al., 1997) and the previously mentioned GIFT (Squire et al., 1999). While the details of implementation and functionality of each system are different, almost all of them follow the architecture shown in Figure 2.21. Veltkamp et al. (2001), Kekre et al. (2011), and Marques and Furht (2002) provide a broad survey of CBIR systems, techniques, and semantics which confirms the high-level architecture shown in Figure 2.21. In this thesis we mainly concentrate on areas which are related to query by example or direct query. In particular we concentrate on feature extraction which is performed on the entire collection and at times on query images if the features are not already known, as shown in Figure 2.21. These features are used to determine image similarity and any improvements in this space can easily be used in other image processing applications. CBIR systems extract features from image databases using a variety of techniques, some of which have been discussed in §2.1.

2.6.3 Domain/collection-specific CBIR

There have generally been two ways to perform domain/collection-specific CBIR. The first method is to design such systems with the particular domain in mind. This method is very popular in fields such as medical CBIR systems. Texture features quite often play a prominent role in such systems (Deserno et al., 2008; Xue et al., 2008). Such systems deal with specialised requirements and often have specialised query interfaces Long et al. (2009). The Image Retrieval in Medical Applications (IRMA) project for example uses global texture descriptors to perform queries such as: all chest x-rays taken from a certain view (Lehmann et al., 2004; Long et al., 2009). The CervigramFinder system on the other hand uses texture and other features extracted from regions specified by the user to find similar images from a database of images related to cervical uterine cancer (Xue et al., 2008).

Another approach is to take an existing CBIRS and change it to suit the collection/domain. Müller et al. (2003) show how GIFT was modified to perform domain-specific CBIR for medical images. The modifications included changes to the colour quantisation levels for the colour feature and also changes to the texture features. GIFT uses Gabor filters to extract texture features from images and Müller et al. (2003) increased the number of filters and adjusted their orientations as part of the changes. These changes allowed for more suitable feature extraction from the images pertaining to the domain. It is notable that even a CBIRS like GIFT, which generally performs well in different domains (Kosch and Maier, 2010), needed to be customised through inspection and trial-and-error. The process of catering to collection-specific and/or domain-specific needs would be greatly simplified if human inspection could be minimised or removed. Although it might not be completely achievable currently, the part about hand-tuning texture feature extractors may be eliminated by the approaches proposed in this thesis.

2.7 CBIR performance evaluation

In order to evaluate the effectiveness of the different feature sets used in CBIR, performance measures are required. "Performance" here does not mean speed, but accuracy when compared to pre-determined human relevance judgements. A wide range of methods have been employed by researchers. Some of them are rather naive, such as simply presenting screen-shots of the query results (Flickner et al., 1995), while others use well known measures from information retrieval (IR). Others still expand on this and use a graphical representation of their results. An exhaustive description of such methods is out of the scope of this thesis. We present descriptions of a smaller subset. A more detailed description can be found in Müller et al. (2001).

There are a few basic challenges in the performance evaluation of a CBIRS. The first problem is that of finding a common image collection, which would help in critically comparing the results of the methods proposed in this thesis with other studies. One of the suggestions put forward by Müller et al. (2001) is to use image collections which are free to use, such as VisTex, a database which was created by the Vision and Modelling group at MIT. The second problem is that of obtaining judgements of similarity between images. While some databases might have agreed upon ground-truths, e.g. the VisTex database[2], that is certainly not the case with the image collections used in various other studies. Müller et al. (2001) suggest the following mechanisms:

- Use of collections with pre-defined relevance sets: This method employs standard image databases which have a pre-defined, agreed upon set of relevance judgments. Some studies might use only sub-sets of the databases depending on the particular nature of the study. These databases however reduce the amount of overhead associated with experimentation.

- Image Grouping: Employ a domain expert to categorise images according to some criteria, which may not be based on any easily perceptible visual element. This is very useful in medical CBIR (Shyu et al., 1999).

- User Judgements: A user is asked to examine the entire database or a representative sub-set to establish relevance for a given query image. However studies have shown that this method does not yield consistent results, as different users select different relevant images (Squire and Pun, 1997). However this method leads to an interesting avenue of work, which uses relevance feedback.

- Simulating users: This method assumes that the metric used in a CBIR, with the addition of noise, simulates user judgement (Vendrig et al., 1999). The amount of noise can be controlled and hence "user behaviour" can also be controlled. However modelling real users is a non-trivial problem and real users don't necessarily obey a deterministic metric, let alone one based on noise.

The performance of CBIR systems has been measured using various different methods. Some of these methods are more subjective than others, while others provide numeric evaluations of performance. Some of the methods are presented here.

- Rank of the best match: Berman and Shapiro (1999) give a description of work done in evaluating some methods to create an efficient CBIR system. To perform their experiments, the authors decided on 51 pairs of images, where each image in a pair was judged to be similar to the other. For each pair, they performed a query with one of the images, and measured the position (rank) of the other image in the results. They had also established a 'true' sequence for query images, allowing them to compare the actual rank of the target image with the 'true' rank.

- Precision and Recall: These are widely used measures in IR (Salton, 1971; Saracevic, 1995; Salton, 1992) and was first proposed by Kent et al. (1955). In the context of CBIR they are defined as:

[2] Even with a collection like VisTex, the ground-truth is not always consistent. Images belonging to different relevance classes have very similar visual texture. We describe this further in chapter 3.

$$\text{Precision} = \frac{\text{Number of relevant images retrieved so far}}{\text{Total number of images retrieved so far}}. \tag{2.49}$$

$$\text{Recall} = \frac{\text{Number of relevant images retrieved so far}}{\text{Total number of relevant images}} \tag{2.50}$$

These measures reflect the effectiveness of the system based on the fraction of the results inspected so far. Using just one of the measures is not sufficient, as it provides an incomplete picture. Recall can be made 1, by retrieving all images, and precision can be made high by only retrieving a few images (Müller et al., 2001). Average precision at recall values have been used (Belongie et al., 1998) and is useful for presenting an overall picture of CBIR performance over multiple queries. Precision after N images have been retrieved, $P(N)$, is also a useful metric and has been used to report results in this thesis.

- Target testing: In this method, users are provided with a target image. Then, the number of images that a user has to evaluate before reaching the target image is counted (Cox et al., 1996). Users are required to mark each image as relevant or not.

For this research we have opted to use quantitative evaluation for CBIR performance. Precision and recall are well established criteria and should be used jointly as the primary evaluation mechanism. Although studies such as He (1997) used *recall vs precision* graphs, we have chosen to use *precision vs recall* graphs as our method of presenting results and to avoid confusion (Müller et al., 2001). Using the appropriate methods to establish relevance, as stated earlier, followed by a numerical and graphical analysis should provide sufficient information about the performance of any new techniques. It would also assist other researchers to compare the performance of the new techniques with existing and/or other techniques.

2.8 Conclusion

In this chapter we presented topics related to the work in this thesis. We mainly concentrated on texture features and highlighted how existing approaches have fixed or predefined parameters that need to be manually adjusted to optimise for different image collections. We have also seen that even a versatile CBIRS like GIFT needs to be customised through experimentation and hand-tuning, when used for domain-specific purposes. We discussed ICA as a method to automate the customisation/hand-tuning of features and showed how it can been used to automatically find filters which are adapted to image collections. We discussed some challenges with using the filters extracted by ICA. The coming chapters will address these issues. We have discussed CBIR and and the role of features in this application. We will use a CBIRS to evaluate the performance of our proposed adaptive features so we also discussed metrics utilised in such an evaluation. In the next chapter we present our initial experiments.

Chapter 3

Comparison of global ICF-based feature with various features and an evaluation of the variance-based filter selection technique

The goal of this thesis is to demonstrate the viability of using collection-specific adaptive texture features for CBIR. Improvements to the adaptive features will be presented in chapters 4, 5, and 6. In this chapter we set the scene for the rest of the thesis. We present the image collections we used to test our methods. We also present the basic framework used to conduct CBIR, and describe how we evaluated performance.

We then present results indicating the utility of the variance-based filter selection mechanism discussed in §2.5.6, and demonstrate its shortcomings. It is vital to understand this issue, as our proposed filter selection method and the performance improvements which are presented in chapter 4 were developed in an effort to address these shortcomings.

We compare the collection-specific ICF-based features against four well known features which were discussed in chapter 2: GLCM, HOG, PHOG and features extracted from banks of Gabor filters. We divide the features into two broad categories, global features and local grid-based features. We are classifying HOG and PHOG as grid-based features. For the other methods we present both global and grid-based features.

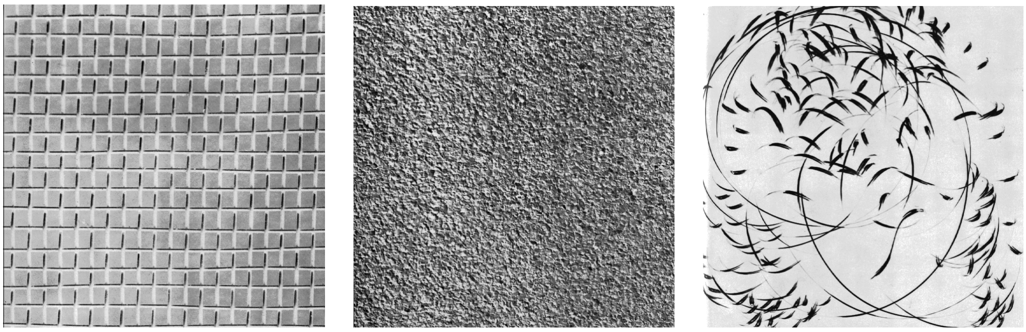
We show in this chapter that the adaptive ICF-based features outperform the other four features. We also show that grid-based features are not very effective for images with globally consistent texture. We also highlight some restriction of Gabor filters and PHOG features when applied to images with globally consistent texture.

3.1 Image Collections

As mentioned in chapter 2, it is recommended to conduct experiments on freely available image collections to facilitate comparison. As the goal of this research is exploring methods to automatically find adaptive texture features, we limited our data sources to texture collections. To demonstrate the repeatability of our findings, we have reported all results on two standard texture databases, the VisTex and the Brodatz collections, for most of the thesis. We discuss these two databases in more detail below. In chapter 7 we present results on two more collections which will be discussed there.

3.1.1 Brodatz

The Brodatz collection (Brodatz, 1966) is the most well known data set in texture analysis. It is derived from the Brodatz album and is a scanned version of glossy black and white prints obtained from the author. It contains 112, 512 × 512 images at 256 grey levels. Many studies do not report results on the entire set. Sayadi et al. (2008) uses only 25 of the Brodatz textures, Ojala and Pietikainen (1998) uses 15 textures etc. Even studies that claim to report on the entire database report on 111 classes (Lazebnik et al., 2005), and the online source[1] also has 111 images available. This collection has been criticised for its lack of intra-class variation. However, the quality of the images, the uniform lighting condition and the high exposure to relevant texture with almost no noise makes this database extremely popular for texture analysis applications. Some example original Brodatz images are shown in Figure 3.1.



(a) D1.gif

(b) D4.gif

(c) D45.gif

Figure 3.1: Examples images from the Brodatz collection

The image set used in this study is similar to the one used in Picard et al. (1993). From each of the original Brodatz images, we took nine non-overlapping 128×128 samples giving a total collection size of 999 images. In this thesis we report results on this entire collection.

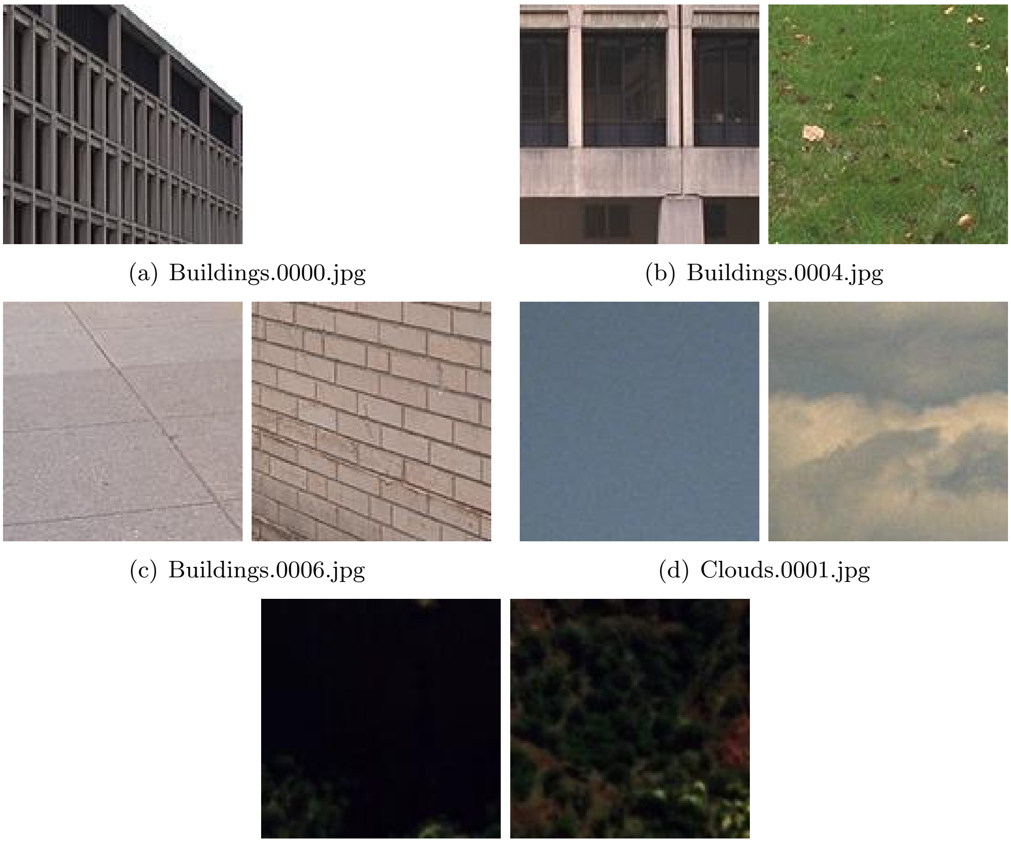
3.1.2 VisTex

The VisTex database[2] contains a set of 167, 512 × 512 images of various kinds of visual texture. Although quite old, this database is still widely used for texture studies (Hossain and Serikawa, 2013). Unlike the Brodatz collection, the VisTex images are not taken under controlled lighting conditions or camera angles. This makes this collection more challenging than the Brodatz. The common practice is to take 16 128×128 non-overlapping samples from each each 512 × 512 image, leading to a total collection of 2672 images as

[1] `The Brodatz database is available from http://www.ux.uis.no/~tranden/brodatz.html`

[2] The Vistex Database is available from `http://vismod.media.mit.edu/vismod/imagery/VisionTexture/`

done in Bai, Zou, Kpalma and Ronsin (2012). However there are a few issues with this method as illustrated in Figure 3.2, in which every pair of images belong to the same relevance class, as they are extracted from the same original 512 × 512 VisTex image. It is immediately clear that these image combinations don't have any texture similarity or even any apparent mutual relevance. To combat this problem many studies use a smaller subset of 40 homogeneous VisTex textures to report their results (Xu and Zhen, 2009; Bai, Zou, Kpalma and Ronsin, 2012; Bombrun et al., 2011; Bai, Kpalma and Ronsin, 2012).



(a) Buildings.0000.jpg

(b) Buildings.0004.jpg

(c) Buildings.0006.jpg

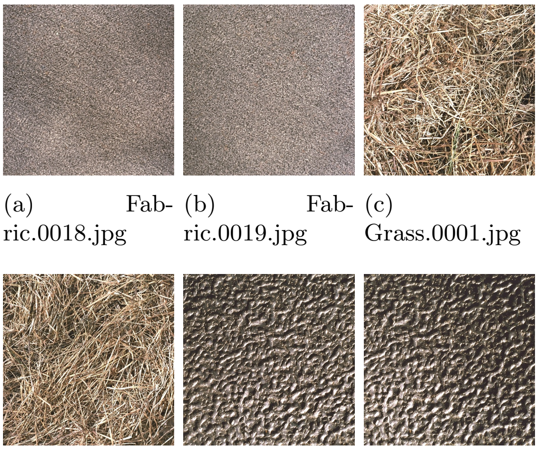
(d) Clouds.0001.jpg

(e) Terrain.0002.jpg

Figure 3.2: Examples of sub-images obtained from the original VisTex database when 16 non-overlapping samples are taken from each image. The original image names are shown as captions. One of the images in Figure 3.2(a) is a white background.

The version used for this study was first used in the development of the Viper/GIFT system at the University of Geneva (Squire et al., 1999). Ten 256 × 256 patches were sampled from random locations from the original VisTex images, and then downsized to 128×128 pixels. These leads to more of the original VisTex image being included in each sample, giving a better chance of significant similar texture being present in each sample.

Some of the images in the original VisTex database are visually very similar, as shown in Figure 3.3. Assigning relevance based solely on the 167 source image IDs is thus not always accurate. Human relevance judgements were used to classify the collection into 124 relevance classes. One query image was selected from each class. The results in this thesis are presented using these 124 texture classes.



(a) Fabric.0018.jpg (b) Fabric.0019.jpg (c) Grass.0001.jpg

(d) Grass.0002.jpg (e) Metal.0004.jpg (f) Metal.0005.jpg

Figure 3.3: Example of some 512 × 512 images from the original VisTex collection which have very similar textures yet are used as a source for difference relevance classes for the work in (Squire et al., 1999)

In Appendix D we present results comparing the performance of the features developed in this thesis on the 40 VisTex textures for comparison with the results from other studies reporting on the same texture classes.

3.2 Independent component filter extraction

The ICF were extracted from these collections using the method described in §2.5.5. We tried different patch sizes and eventually found 17×17 to be the most suitable size. So, for each image collection $10,000$ random 17×17 patches were taken from a subset of the collection containing one example of each texture class. Each patch was converted to a 289 dimensional vector, yielding a $289 \times 10,000$ data matrix. As is common in the application of ICA in image processing, Principal Component Analysis (PCA) (Hotelling, 1933) was used to reduce the dimensionality of the dataset, by dropping the components responsible for the least significant 1% of the variance. An implementation of FastICA in MATLAB was used to extract over 200 ICF for each image collection, examples of which are shown in Figure 3.4 and Figure 3.5.

3.3 Variance-based independent component filter selection

We followed the method described in §2.5.6, which was proposed by Le Borgne and Guérin-Dugué (2001). We created a set of training images T consisting of one training image from each texture class. For each filter f_j and image I_i, $I_i \in T$ we found E_{I_i,f_j}, the response energy image of the filter with training images I_i. The variances of the response energies were calculated across the training images I_i for each filter j, giving $\sigma_j^2 = \sigma^2([\bar{E}_{I_1,f_j}, \bar{E}_{I_2,f_j} ... \bar{E}_{I_n,f_j}])$, where $\bar{E}_{I_i,f_j}$ is the average filter response energy of filter

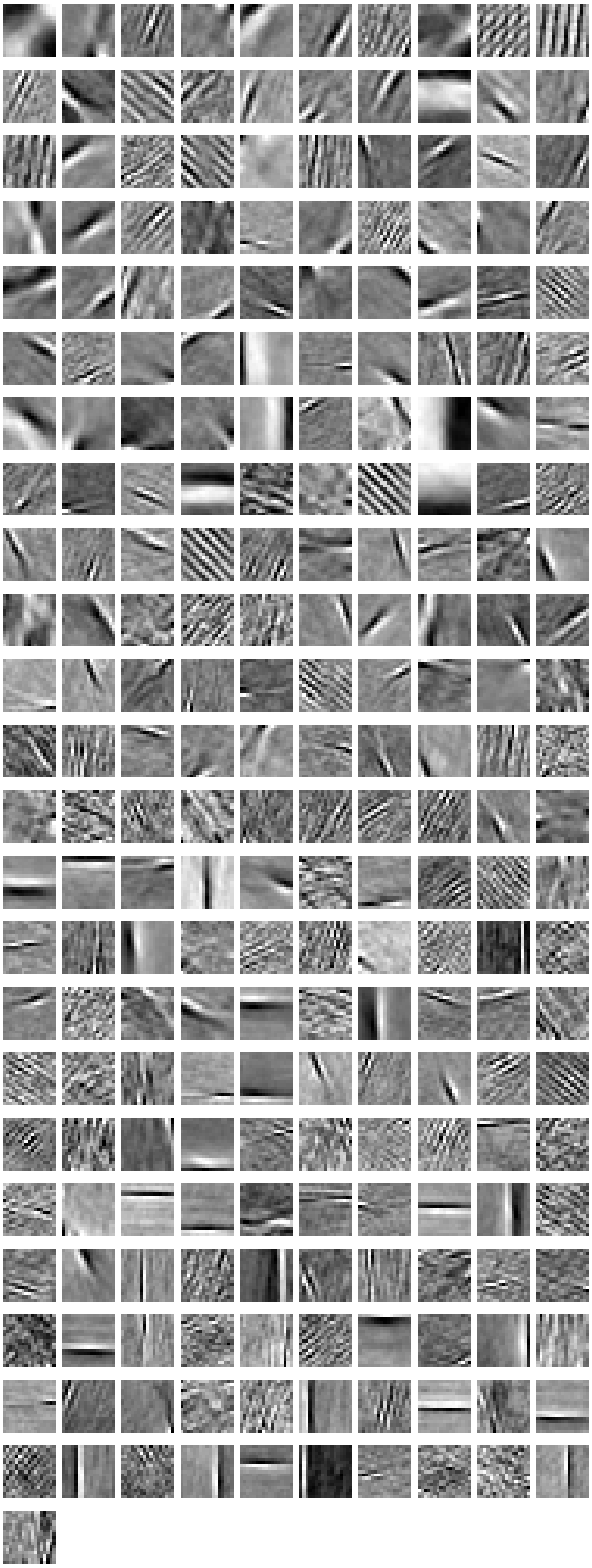


Figure 3.4: Sample ICFs extracted from the VisTex collection.

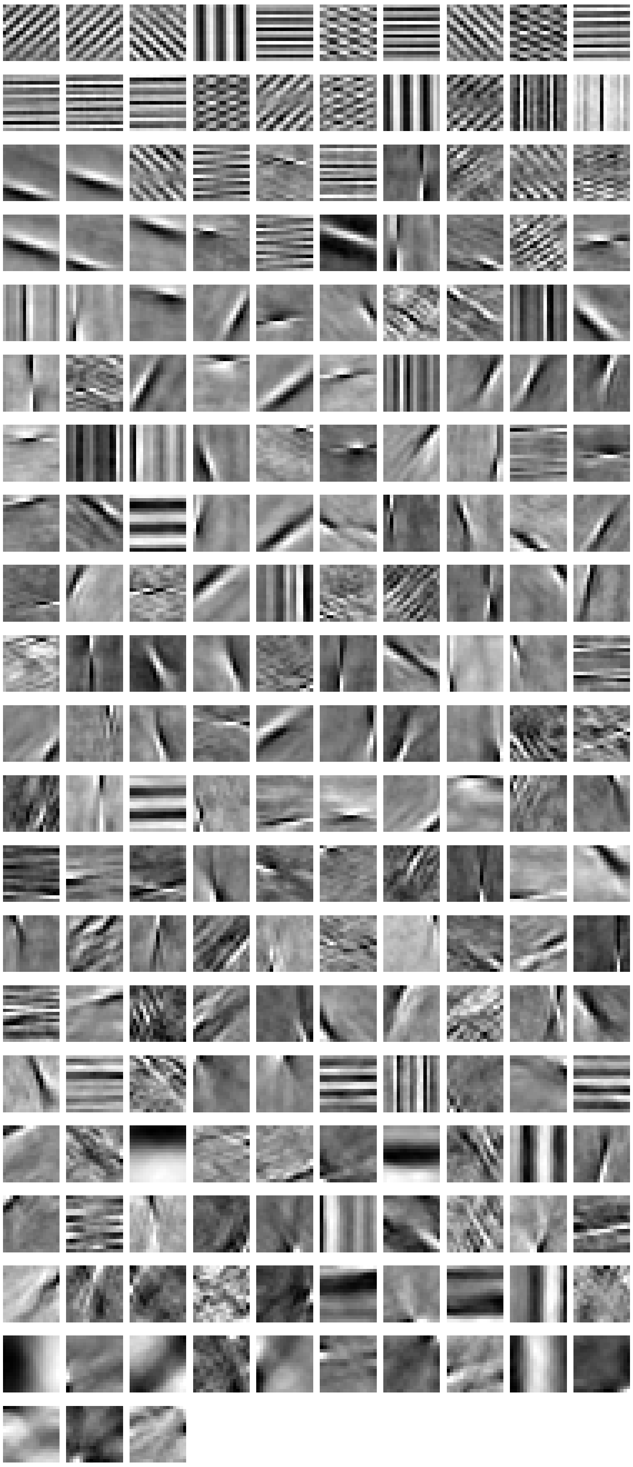
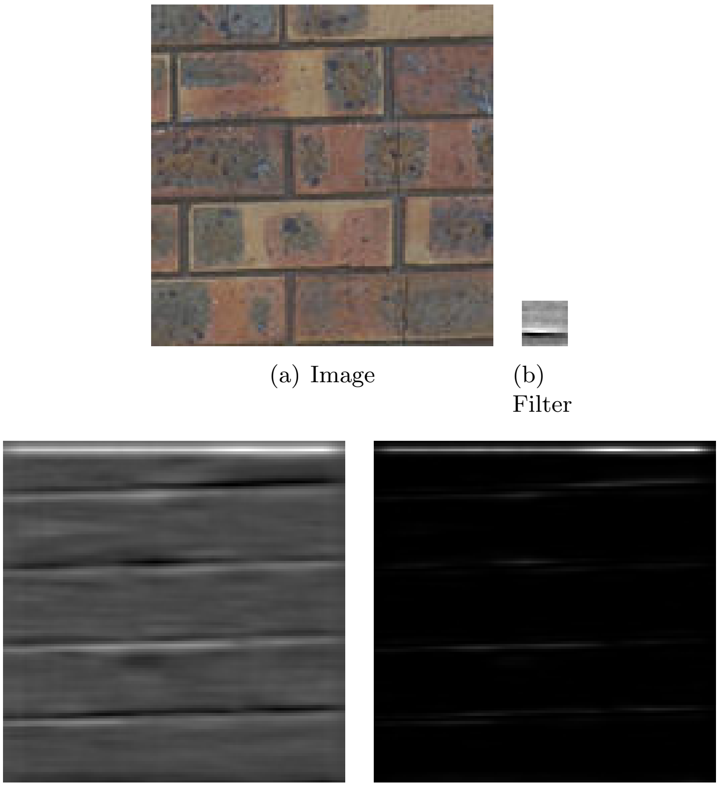


Figure 3.5: Sample ICFs extracted from the Brodatz collection.



(a) Image

(b) Filter

(c) Response, scale for visualisation

(d) Response energy, scaled for visualisation

Figure 3.6: An example of an image, its filter response and the response energy

f_j with image I_i. We then choose the top m filters with the highest σ_j^2 values. The normalisation step to calculate dispersal done in Le Borgne and Guérin-Dugué (2001) is not needed for our purpose as all it does is scales all the variances between 0 and 1.

3.4 Feature Extraction

3.4.1 Global features

For each image I_i in the database, the energy of the response of each filter f_j is calculated as

$$E_{I_i,f_j} = (I_i \otimes f_j)^2 \tag{3.1}$$

where $\otimes$ is the two dimensional convolution operation.

The feature vector V_i for image I_i holds N entries, where N is the total number of ICF/filters. $V_i(j)$ is the average energy of filter j with image i found by:

$$V_i(j) = \frac{\sum_x \sum_y E_{I_i,f_j}(x,y)}{\text{size}(E_{I_i,f_j})} \tag{3.2}$$

where $\text{size}(E_{I_i,f_j})$ is the number of pixels in E_{I_i,f_j} and $E_{I_i,f_j}(x,y)$ is the value of the response energy at location (x,y). This process is depicted in Figure 3.7.

The distance between two images I_a and I_b is calculated to be the Euclidean distance between their feature vectors.

$$d_{ab} = \sqrt{\sum_n (V_a(n) - V_b(n))^2} \tag{3.3}$$

3.4.2 Grid-based local feature

When extracting grid-based features using filters, we follow a method closely related to that used in the GIFT system (Squire et al., 1999). For each filter f_j, the filter response energy for image I_i is calculated at each pixel. The energy image is then divided in a 16×16 grid resulting in 256 blocks. The average energy in each block of the grid is calculated AE_{ijk} where $1 \leq k \leq 256$. For each block of image I_i we form a vector b_{ik} which is the concatenation of the average response energies of all the filters. So, for each block, b_{ik} would be an N dimensional feature vector, where N is the total number of filters used. The image feature is simply the concatenation of all the b_{ik} for image I_i.

3.5 Features for comparison

3.5.1 Bank of Gabor filters

Initially in these experiments we used a bank of 12 circularly symmetric Gabor filters using the same configuration as used in the GIFT system (Squire et al., 1999). The filters are generated at 3 scales and 4 orientations and seem to give good coverage. We tried different combinations of scales and orientations, and we found that increasing the number of scales to more than 3 actually decreased performance (Figure 3.8). In the coming sections of this chapter we will present results for filters at 3 scales and $4, 8, 25, 50$ and 80 orientations resulting in $12, 24, 75, 150$ and 240 filters. Features are extracted following the steps described in §3.4.

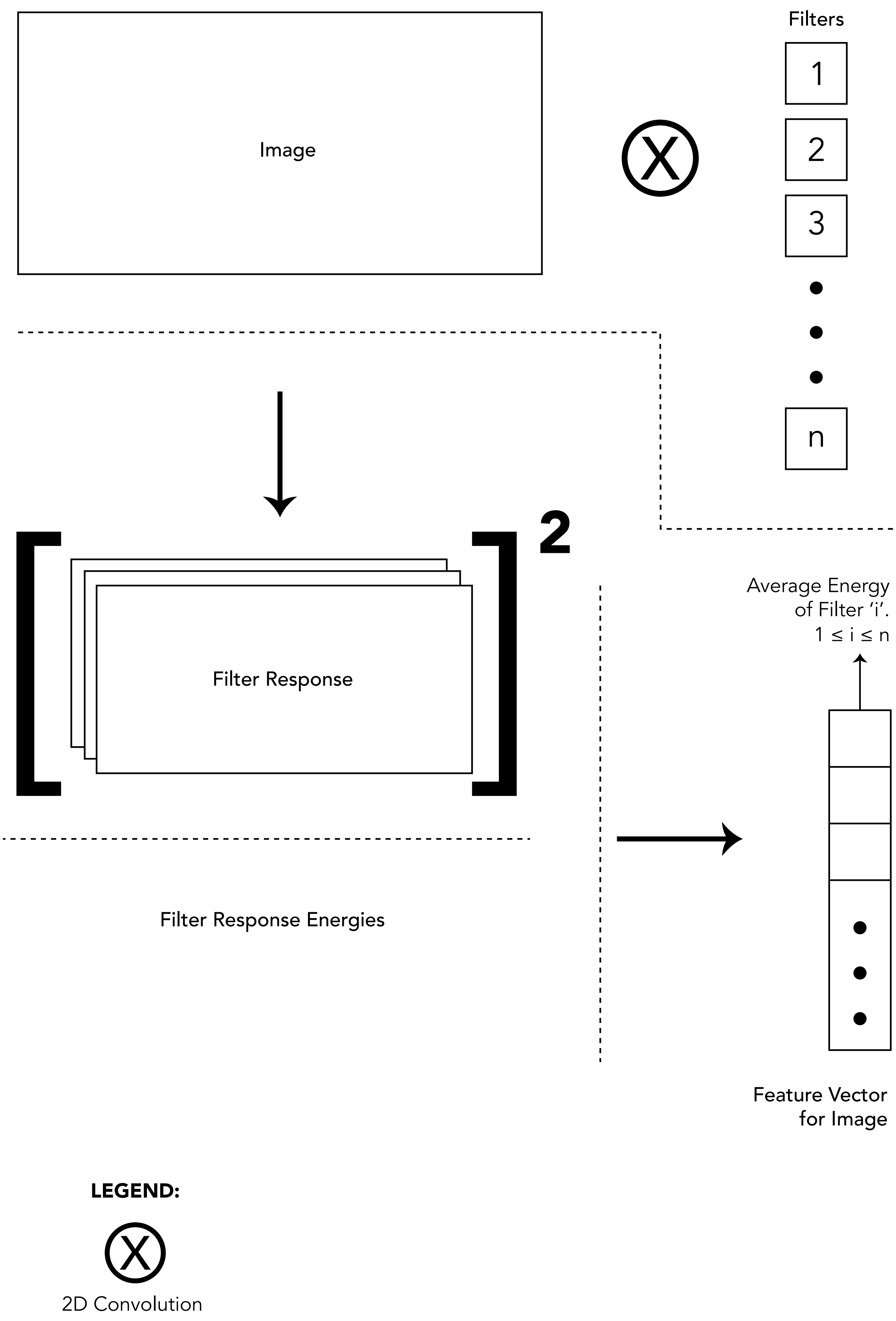
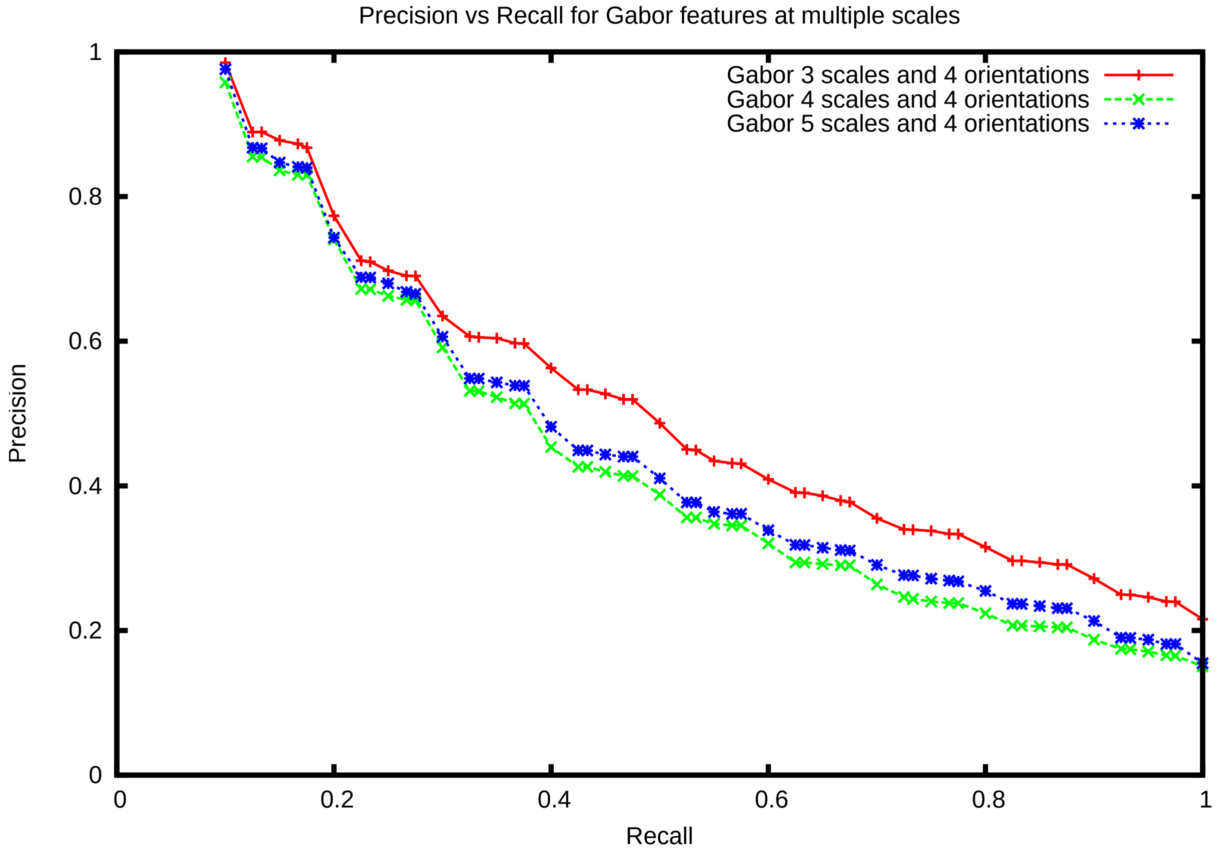
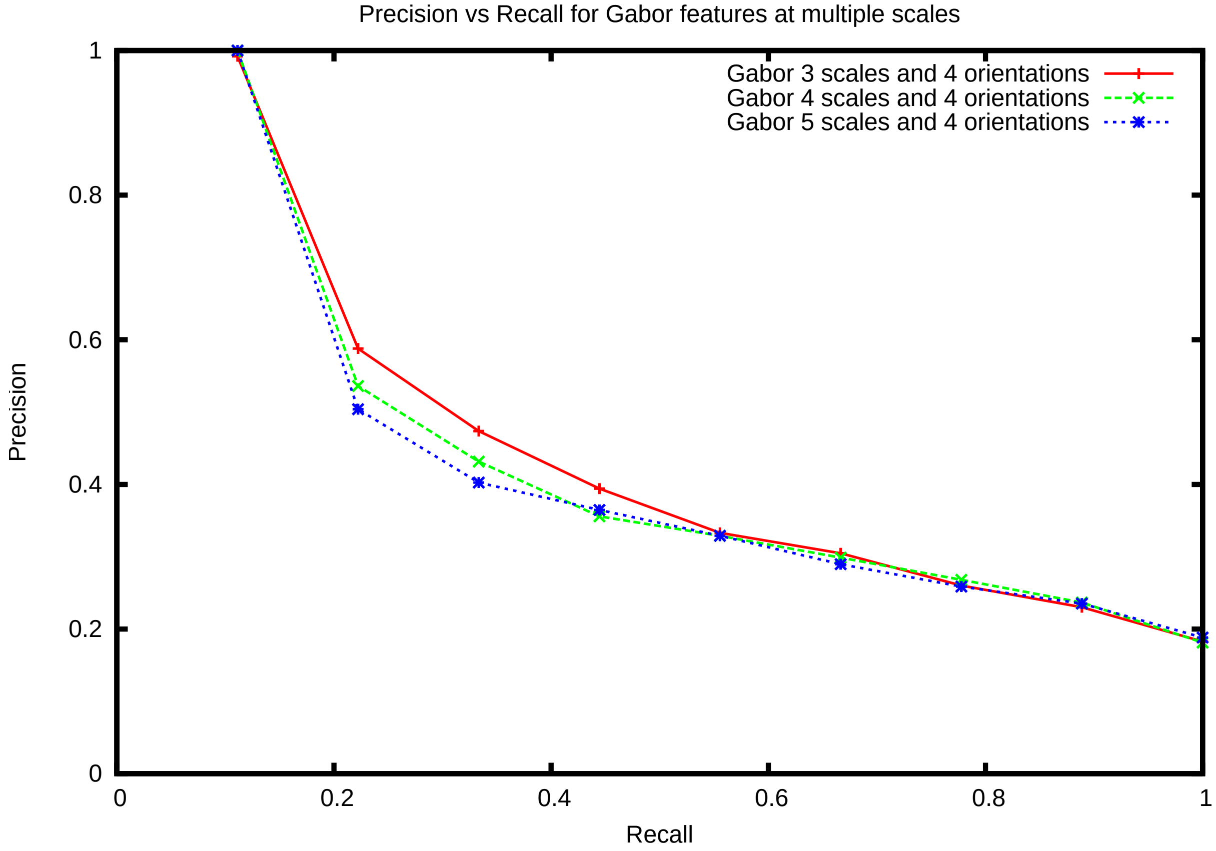


Figure 3.7: Global feature extraction process. This method is used for both the ICF and Gabor filters.



(a) VisTex



(b) Brodatz

Figure 3.8: Increasing the scale of the Gabor filters seems to have a detrimental effect on performance on the VisTex and the Brodatz collections.

3.5.2 Grey level co-occurence matrix

We have described grey level co-occurence matrices (GLCM) in §2.4.1. Here we give the details of the implementation we have used in the experiments. We chose to implement the method used by (Howarth et al., 2005) where GLCMs were extracted for medical images, which could be classified as a specialised collection.

The study in Howarth et al. (2005) divided each images in 7 × 7 sub-regions and calculated 16 GLCMs for each. The GLCMS were generated for vectors of lengths 1,2,3 and 4 pixels and orientations 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$ and $\frac{3\pi}{4}$. For each GLCM $P(i, j)$, they calculated a homogeneity feature H_p,

$$H_p = \sum_i \sum_j \frac{P(i, j)}{1 + |i - j|}. \tag{3.4}$$

Using the Manhattan distance between these features they calculated the similarity between the query images and the images in the database.

For our work, we implemented their scheme and used it as local GLCM features. For global GLCM features, we calculated GLCMs for whole images, rather than breaking it up into patches.

3.5.3 HOG and PHOG

We treat HOG features as a grid-based feature. The HOG features we compared against do not vary much from the description given in §2.4.4. A nine bin histogram is calculated for every 8 × 8 cell. Cells are merged into blocks and normalised in an overlapping manner as described in §2.4.4. Each block thus contains a 36 dimensional feature vector. The vectors from each block are concatenated to create the image feature vector. This is similar to the HOG features extracted in Zhu et al. (2006); Wu and Nevatia (2008); Paisitkriangkrai et al. (2008).

Most PHOG results are shown for image descriptors extracted at 3 levels and 8 bins, as done in Bosch, Zisserman and Munoz (2007). The method is described in more detail in §2.4.4. Results with smaller levels are also shown at the end.

3.6 Performance Evaluation

The ICF experiments were repeated 10 times for each image collection, as patch selection and other elements of the algorithm have some randomness. The ICF results are presented using error-bars indicating the standard deviation of the precision values at each recall value. For each unique recall value, the precision values are averaged over all query images. One query image was chosen from each relevance class. This gave us 124 and 111 query images for the VisTex and Brodatz collections respectively. For the VisTex database, we have relevance classes with different numbers of relevant images. In this case, where a recall value does not exist for a particular query image, the corresponding precision value is inferred through interpolation.

3.7 Results and Discussion

3.7.1 Feature set labels

Table 3.1 summarises the labels used to present the results.

Label	Description
ICF-ALL	Features extracted using all the ICF
ICF-Var-N	Features extracted using N ICF filters, where the filters are selected using the variance-based method as desribed in §3.3
ICF-Local	Grid-based ICF features using all the extracted ICF
Gabor-N	Feature extracted using N Gabor filters. We have employed Gabor filters at three scales. So, the number of orientations is $\frac{N}{3}$.
Gabor-Local	Grid-based Gabor features using 24 Gabor filters at three scales and eight orientations
GLCM-Global	Grey level co-occurrence matrix based global features
GLCM-Local	Grey level co-occurrence matrix based local features
HOG	Histogram of oriented gradients feature
PHOG N level(s)	PHOG features extracted at N levels and eight bins

Table 3.1: Feature set labels used to present the results.

3.7.2 Initial experiments

We have already shown in Figure 3.8 that increasing the number of scales of the Gabor filters from three seems to have an adverse effect on performance. In Figure 3.9 we present the performance of the Gabor filters at different orientations. We can immediately see a large improvement in performance going from four orientations to eight. Increasing the number of orientations after eight, however does not seem to have any extra benefit. This is consistent also with the findings of Lowe (1999) and Dalal and Triggs (2005) where they found that increasing the number bins in the orientation histograms from eight and nine respectively did not yield any extra benefit.

In Figure 3.10 we see a comparison between the performance of the features extracted by all the ICF filters contrasted with the performance of 240 Gabor filters and also GLCM-Global. We note that using 240 Gabor filters results in very similar performance as 24 Gabor filters. In this case the ICF features perform better than the other two global features.

3.7.3 Performance of filters selected using the variance-based method

200 plus is a rather large number of filters, consequently we used the variance-based method to select a smaller subset of filters. In figure 3.11 we show how 10 selected filters perform against 12 Gabor filters and also global GLCM.

For the VisTex database the Gabor and GLCM features outperform the 10 ICF features. In fact from the errorbars we can see that the variation of results for the 10 selected

ICF is quite significant. For the Brodatz collection the global GLCM features outperform the ICF features.

In Figure 3.12 we can see that progressively adding further ICF actually improves the performance of the ICF-based features. In fact for both the collections there is still an improvement in performance going from using 160 ICF to 200 ICF. This is at odds with the original intent of using variance as a measure of the discriminatory ability the filters. Le Borgne and Guérin-Dugué (2001) state that $20\% - 25\%$ of the filters with the highest variance should have results similar to using all the filters. The performance of 40 ($\approx 20\%$ of all filters) filters selected using the variance-based method is closer to the performance of using all the ICF for the Brodatz collection than it is for the VisTex collection. However, for both the collections the difference is significant enough to translate into an inferior user experience. There is scope to investigate the types of filters chosen by the variance-based method and improve upon the selection procedure. This is discussed further in chapter 4. For now, in a comparison to Gabor filters, where adding any new filters beyond 24 was fruitless, the ICF filters chosen using the variance-based method kept on adding value untill all the filters were selected.

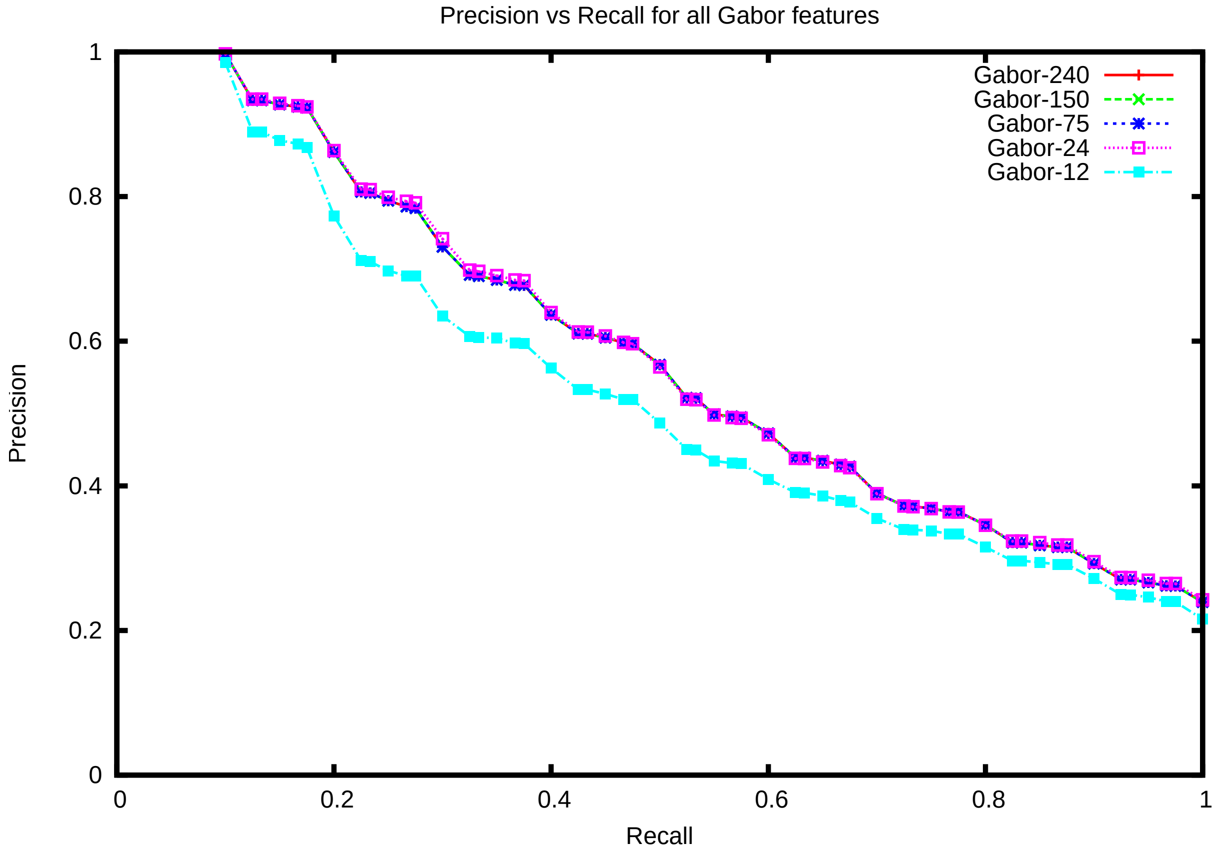
3.7.4 Comparison with GLCM, HOG, PHOG features

In Figure 3.13 we compare the ICF global features with other global features and also some grid-based features: HOG, PHOG, and the grid-based features using ICF and Gabor filters described in §3.7. In summary, we can say that none of the grid-based features perform as well as the global features for images with globally consistent texture. This is not surprising, as all the grid-based features used here encode spatially local information by assigning each grid block to fixed places in the feature vector. This creates an image descriptor which would work quite well if we were interested in comparing images where the shape of the object is significant or where the texture is different in different regions. However for images with globally consistent texture the local variations might not be fixed in particular regions of the image and using such a grid-based approach is actually counter-productive. This is highlighted even more in the performance of the ICF-Local and PHOG features. From the results in Figure 3.13 we have seen that, in this set up, the ICF filters extracted the most discriminatory features when applied globally. That is, the filters responded to texture artefacts within an image which the other features did not. However when encoding the same discriminatory features in an image descriptor which enforces spatial locality we see that the ICF features lose their effectiveness. The difference in the performance of the global and local features is highest for the ICF features. Among the grid-based features, the PHOG descriptor performs best. One possible reason is that it actually encodes a global descriptor at the base level of the pyramid and then incrementally extends the image descriptor by adding the other levels of the image pyramid. The image pyramid employed is again fixed sub-regions of the image. Figure 3.14 shows support of this hypothesis where we show the results of PHOG descriptors extracted at different pyramid levels. The descriptor extracted from only level 0 alone is the best performing

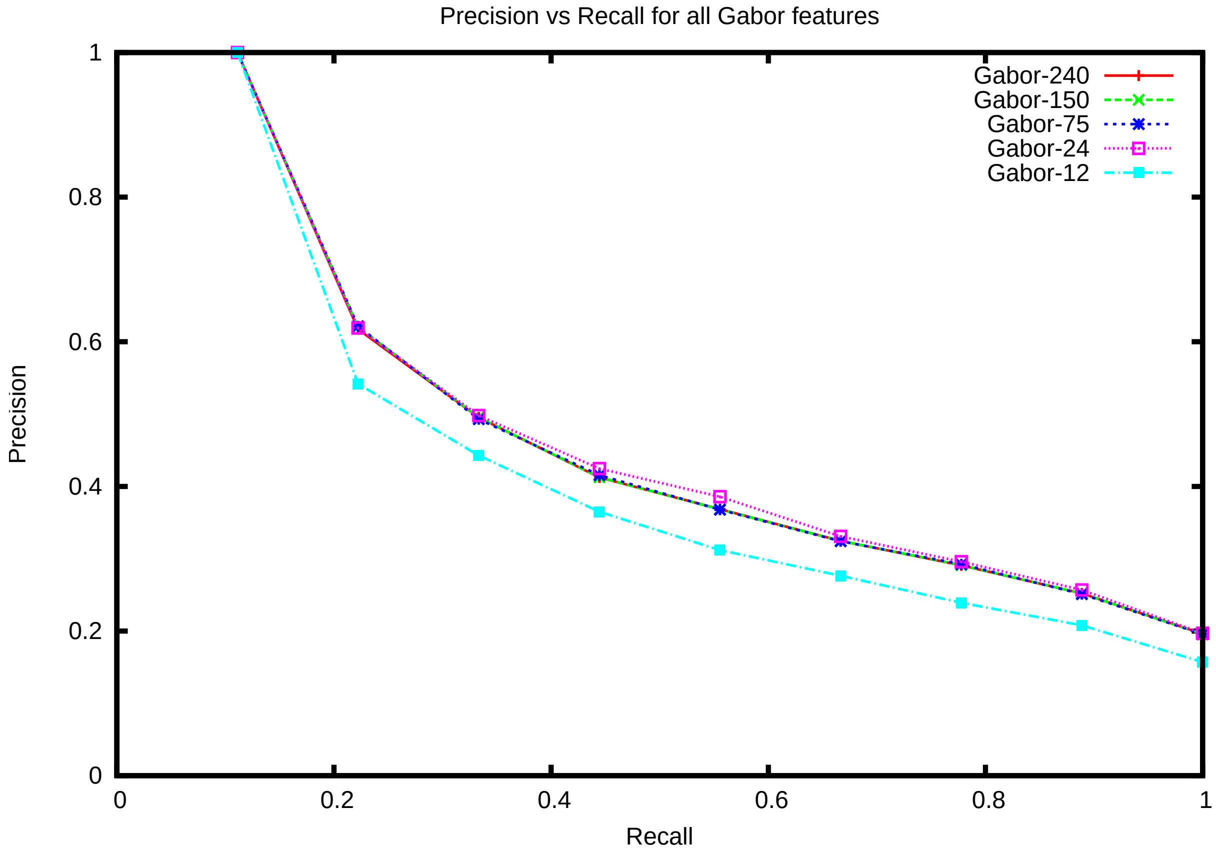
PHOG descriptor. This essentially is a global HOG descriptor extracted by summing the orientation histograms rather than concatenating them.

3.8 Conclusion

In this chapter we have attempted to set the scene for the rest of the thesis. We have presented the image collections we will be using, and outlined our feature extraction and comparison process. We have presented results using two image standard texture collections, which show that ICF-based features outperform the four features we have chosen for comparison. We have also shown that local features extracted using a grid-based method perform poorly for images with globally consistent texture. We have demonstrated this experimentally using the PHOG feature at multiple levels. We have experimented with the variance-based filter selection method proposed by Le Borgne and Guérin-Dugué (2001) and found that the ICF being selected, especially when it is a very small number (≈ 10) do not perform as well as some of the other features (GLCM, 12 Gabors, PHOG with 0 levels). In the next chapter we examine the reasons for this poor performance, address some of the issues with the variance-based selection method, and propose an alternate clustering-based method which outperforms the variance-based method in terms of the effectiveness of the selected filters.

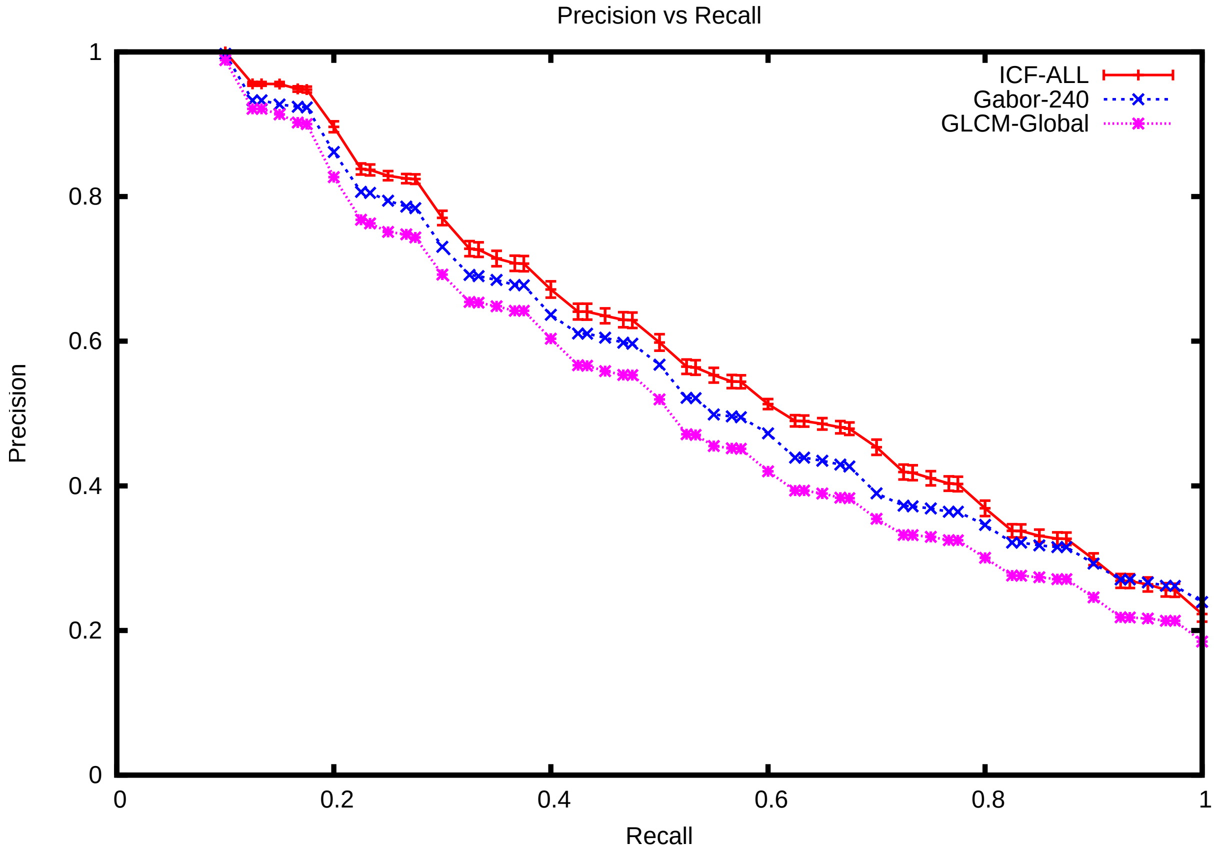


(a) VisTex

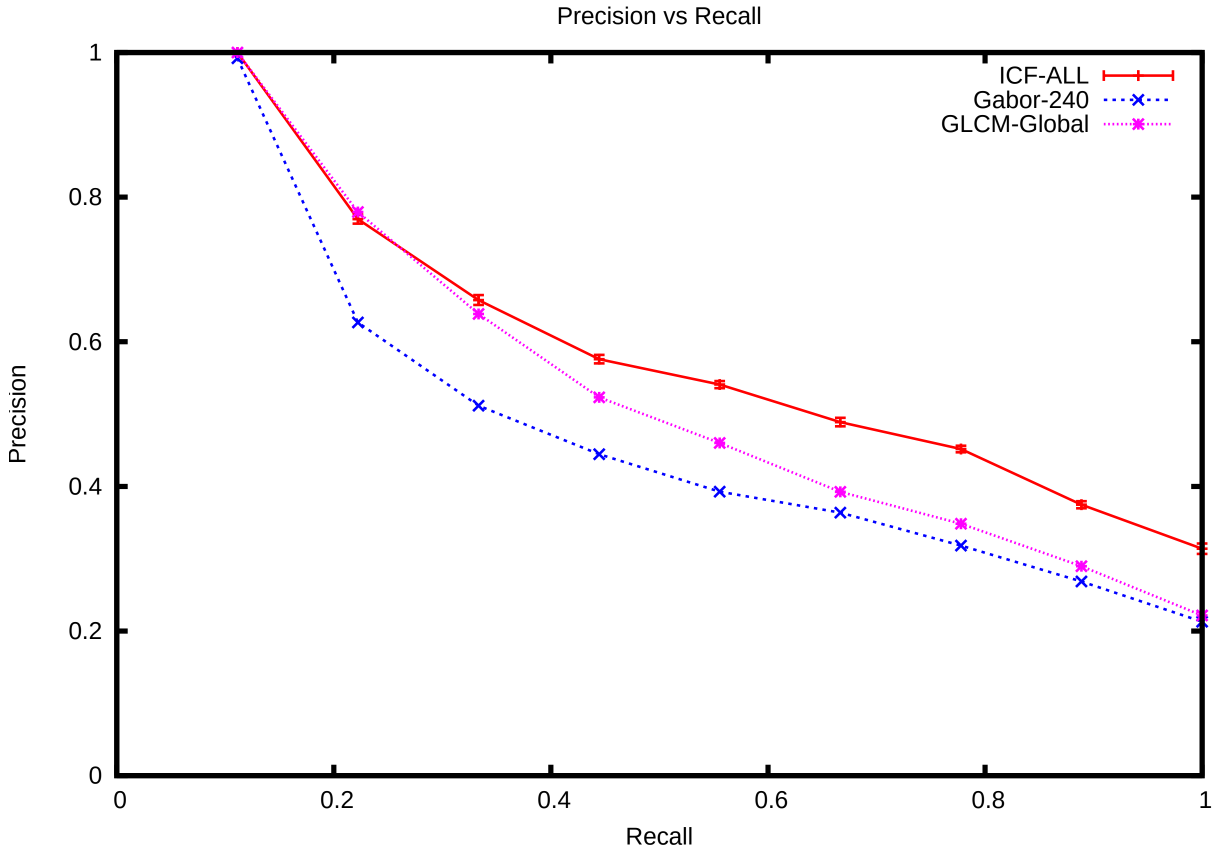


(b) Brodatz

Figure 3.9: Comparison of the perfromance of the difference Gabor filter banks.

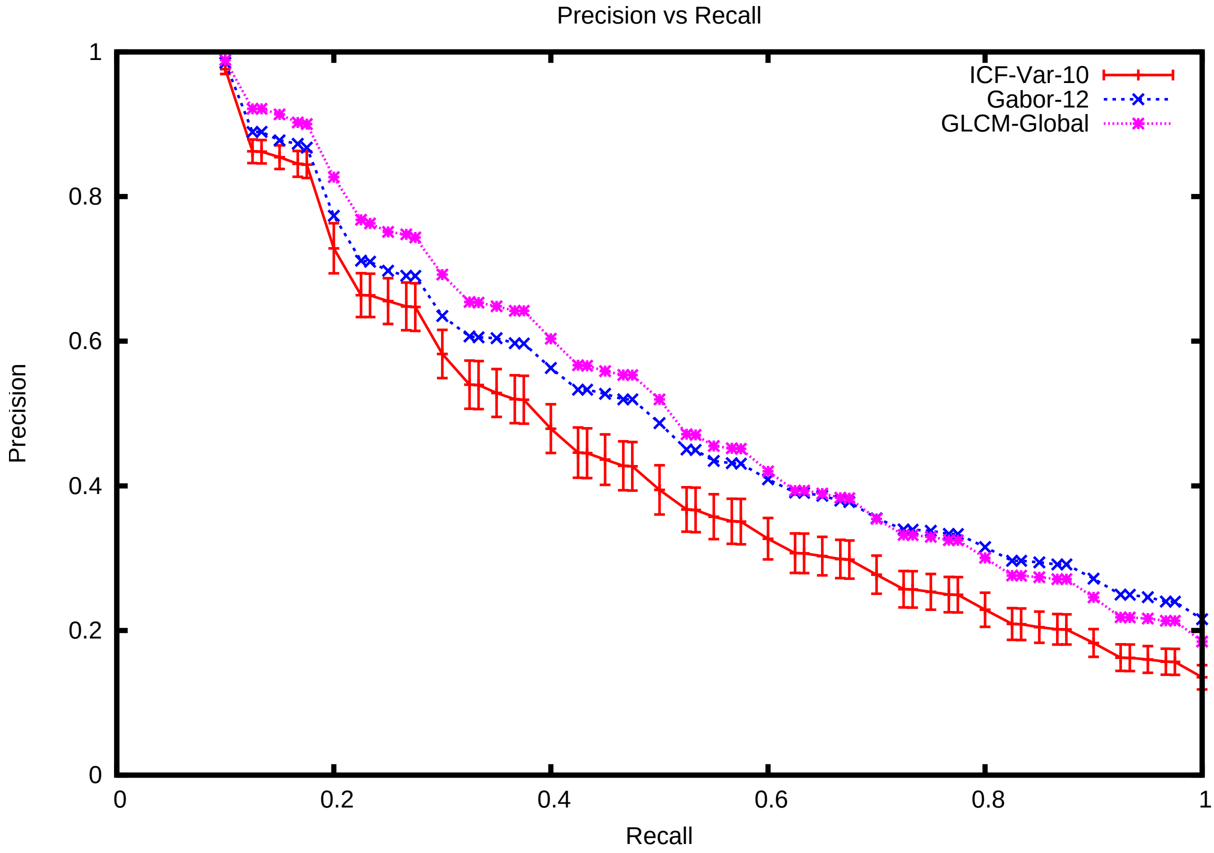


(a) VisTex

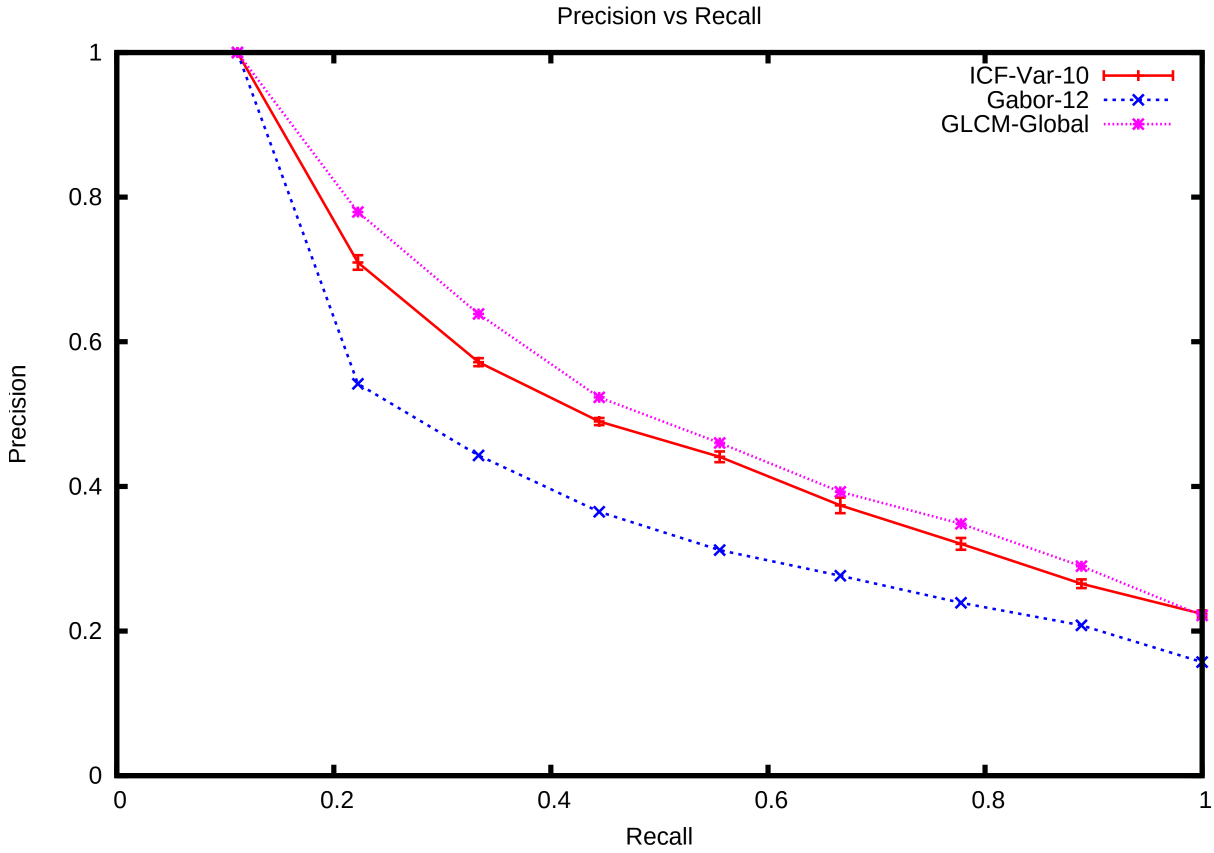


(b) Brodatz

Figure 3.10: comparison of the performance of all the ICF filters, 240 Gabor filters, and Global GLCM

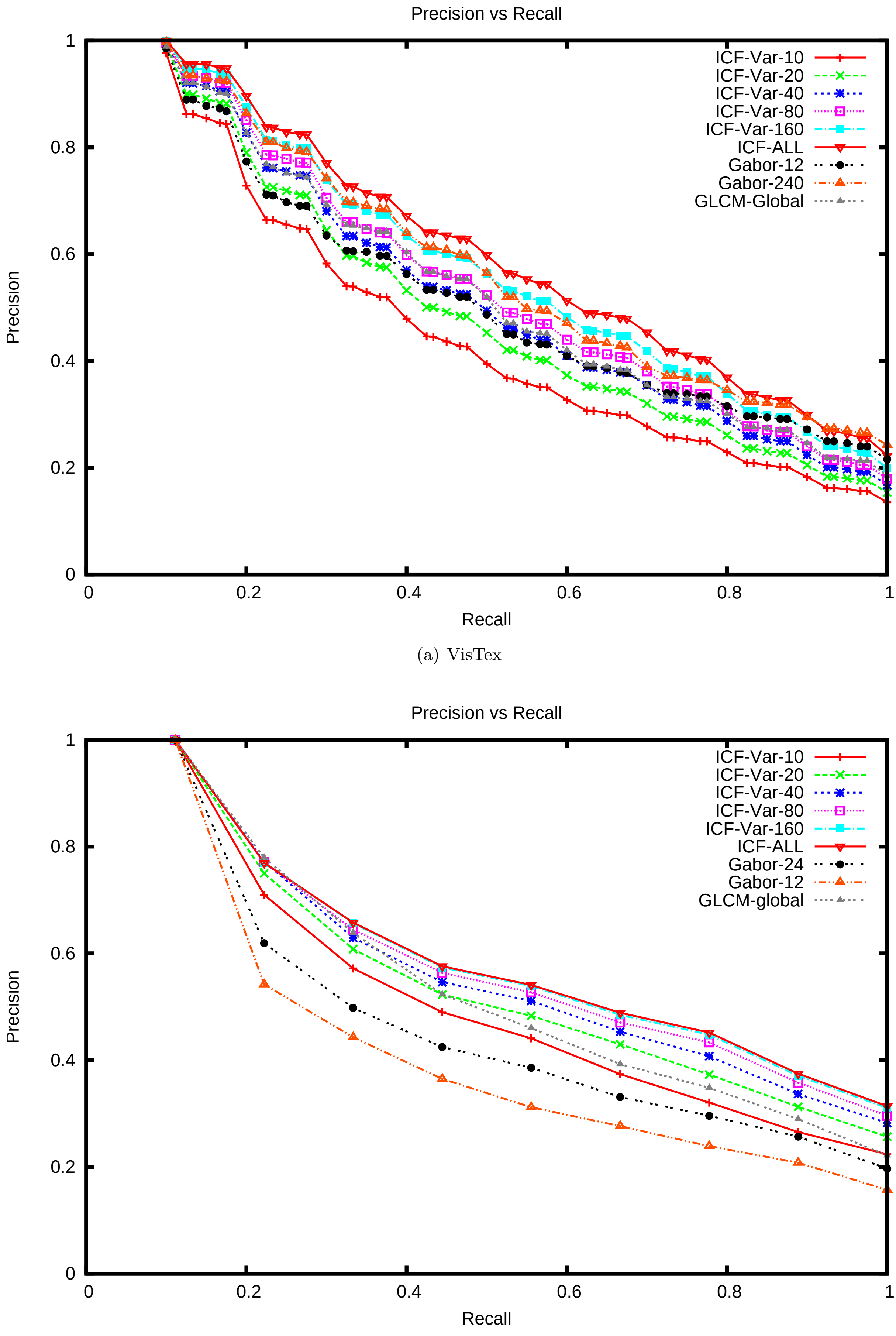


(a) VisTex



(b) Brodatz

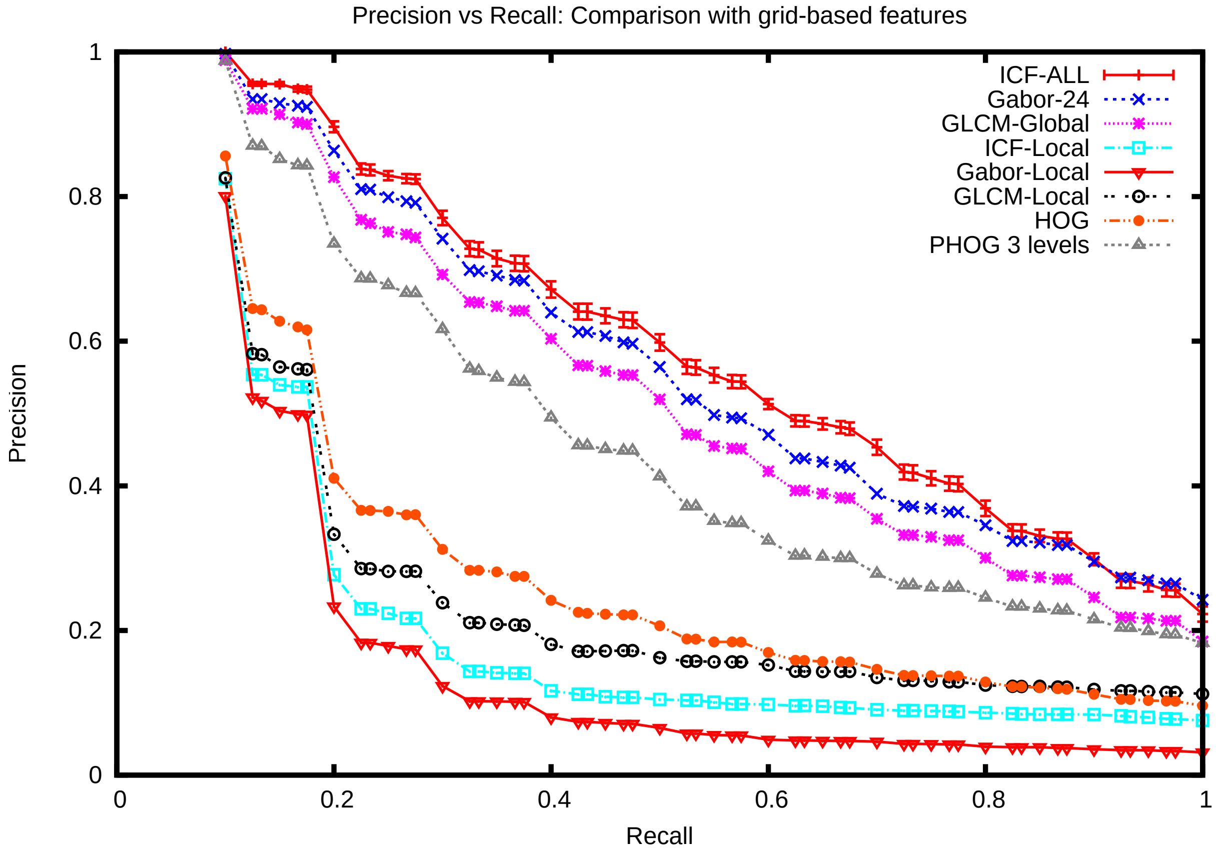
Figure 3.11: Comparison of the performance of 10 ICF filters, 12 Gabor filters, and Global GLCM



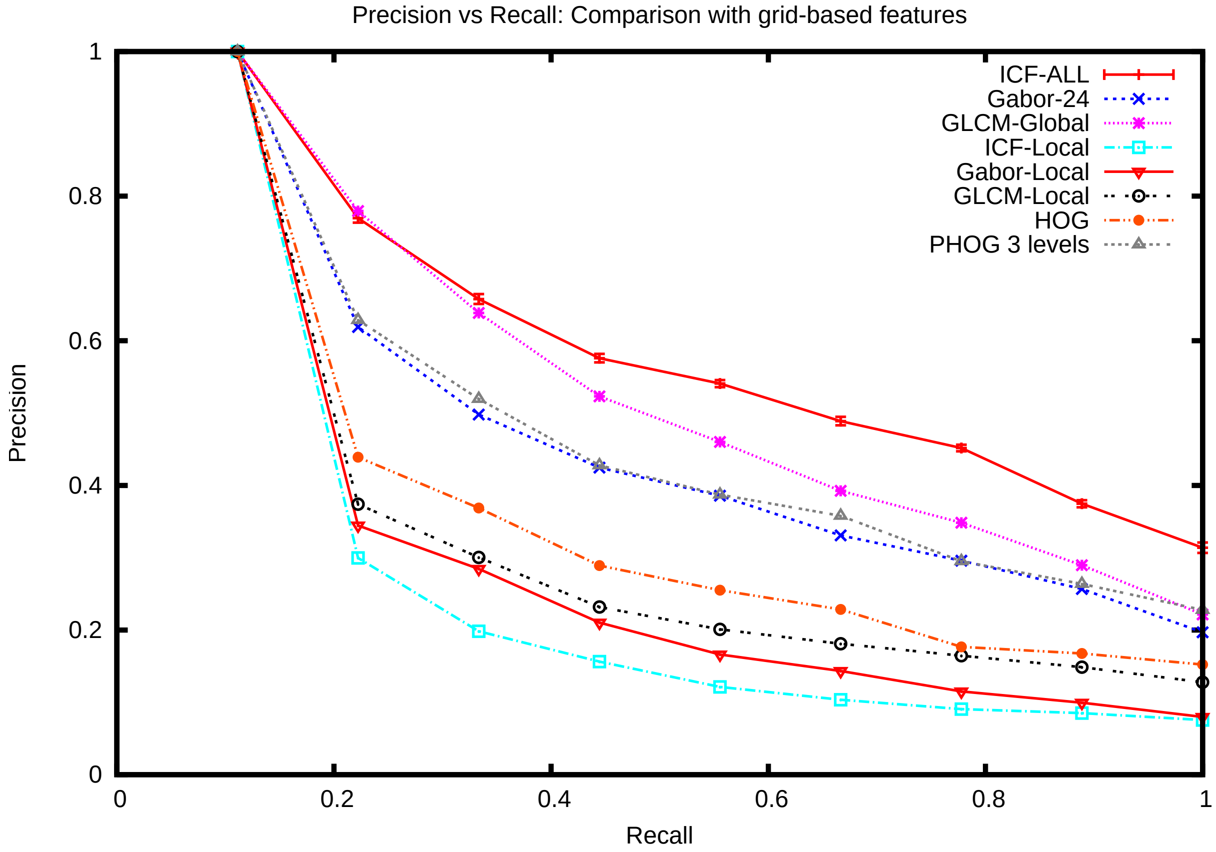
(a) VisTex

(b) Brodatz

Figure 3.12: Performance of the variance-based filter selection method as more and more filters are selected.

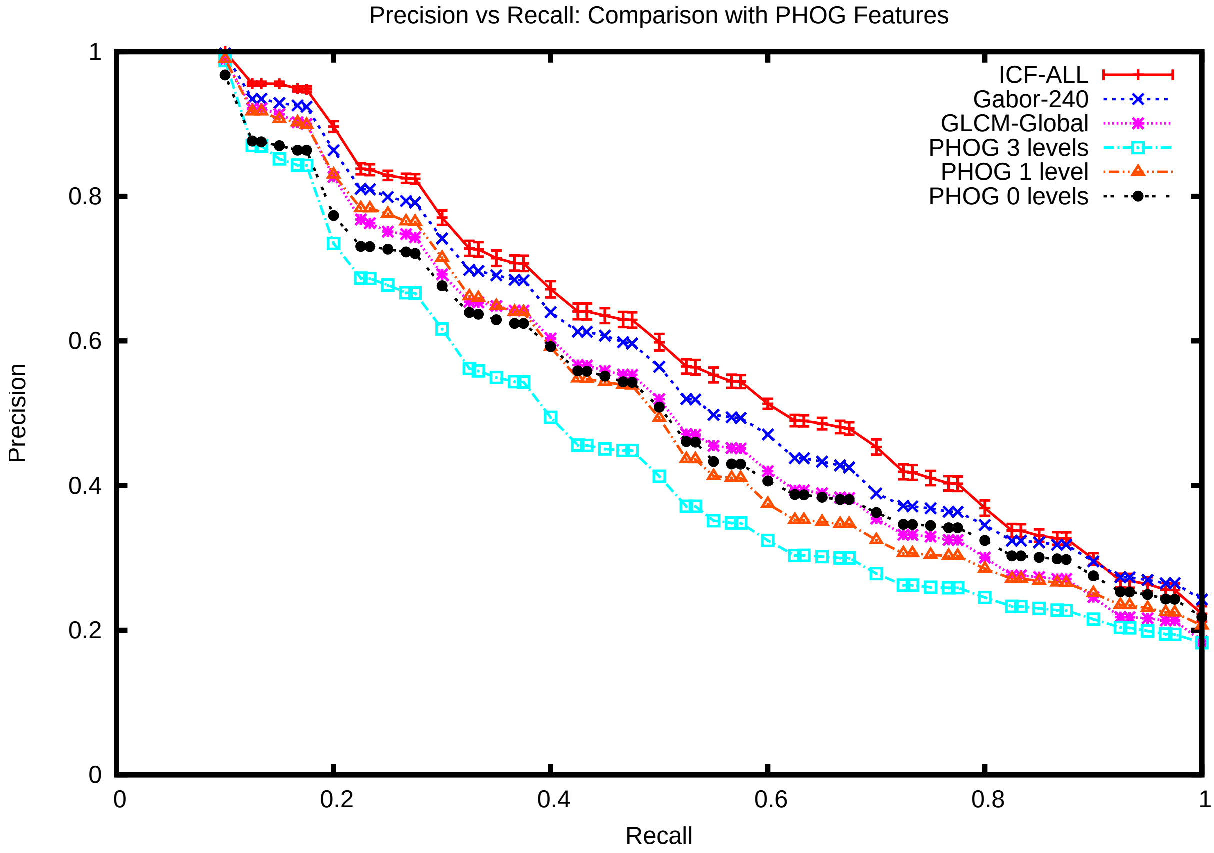


(a) VisTex

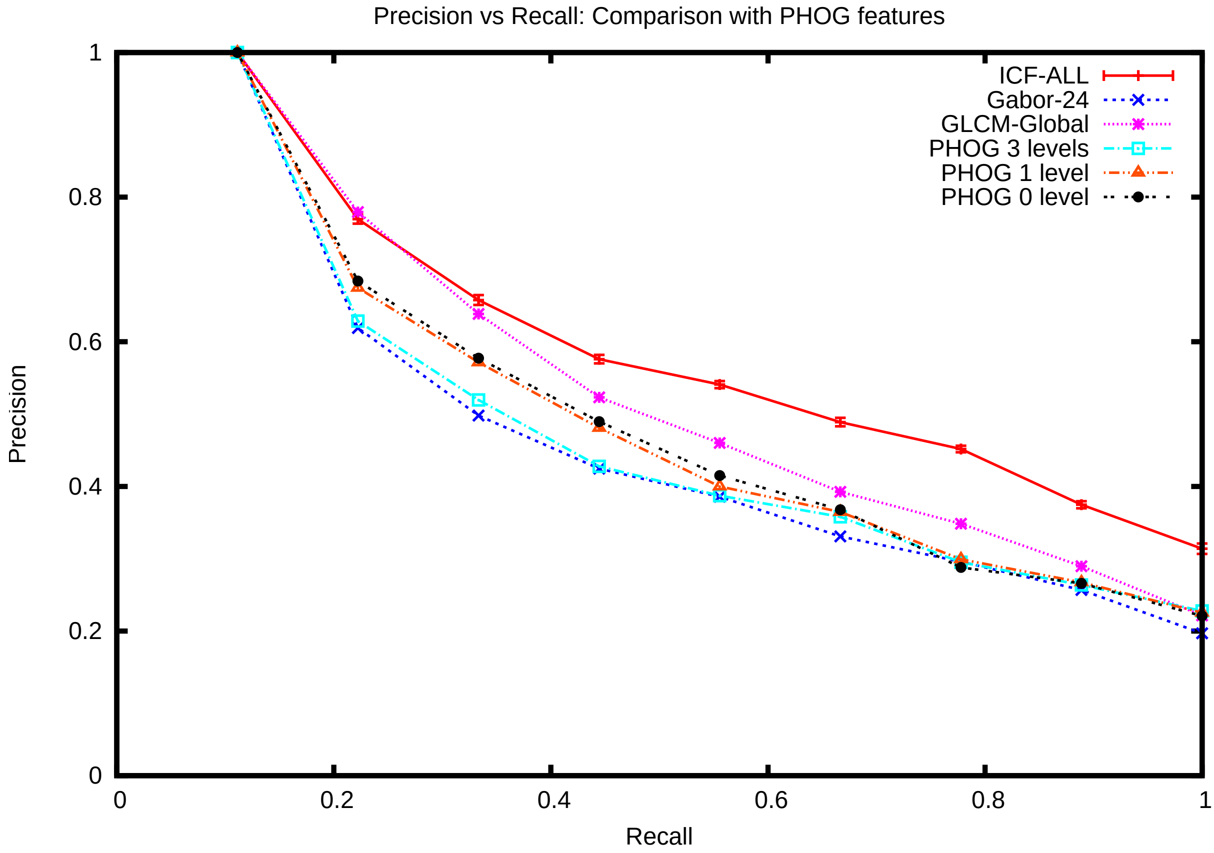


(b) Brodatz

Figure 3.13: Results of Global features and fixed grid-based features.



(a) VisTex



(b) Brodatz

Figure 3.14: Comparing the results when the pyramid of PHOG is reduced.

Chapter 4

Improved performance of ICF features with response scaling, locally normalised convolution and clustering-based filter selection

In the previous chapter we presented the results of performing CBIR using a smaller subset of ICF, selected using the variance-based method proposed by Le Borgne and Guérin-Dugué (2001). We showed that this method of filter selection has some shortcomings. It does not perform as well as a similar number of Gabor filters when very small number of filters are chosen. It seems that it does not choose the most effective filters initially, as the performance improves as more and more filters are added. In this chapter we present an improved filter selection technique based on our investigation of these results. We describe response scaling, which deals with ICF responses differing by two orders of magnitudes. We also describe our use of locally normalised convolution (LNC) to deal with local variations of illumination within an image that cannot be handled by global normalisation. We show that both response scaling and LNC individually give better performance for ICF-based features compared with the method that was presented in chapter 3. Combining them however gives even more significant improvement. We then show that these techniques also increase the performance of the Gabor filter-based features. We present results that show a clear improvement in the performance of the variance-based filter selection strategy resulting from applying our proposed changes.

Further investigation reveals that the variance-based filter selection method is susceptible to selecting certain filters which are very similar in nature and excluding others. We propose a method to choose a more varied filter set with fewer similar and/or shifted filters that uses two-dimensional cross-correlation as a measure of filter similarity followed by complete-link clustering. We show that filters chosen using this method consistently outperform the filters chosen by the variance-based filter selection method of Le Borgne and Guérin-Dugué (2001).

4.1 Response scaling

Figure 4.1 shows a set of ICF extracted from the Brodatz database sorted by the variance of their response energies, calculated as described in §3.3. The method proposed by Le Borgne and Guérin-Dugué (2001) would select n filters row-wise and down from the top left in order. From visual inspection alone it is clear that the initial filters selected would respond to texture elements which are at large scales. Also many of the selected filters would be shifted/near-duplicate versions of each other (e.g. filters five and six and also filters ten and eleven). Also, the filters which have low variance are largely those that have smaller scales.

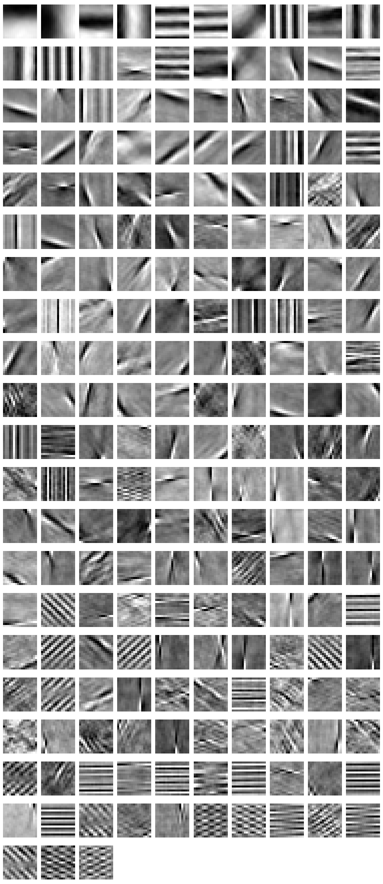
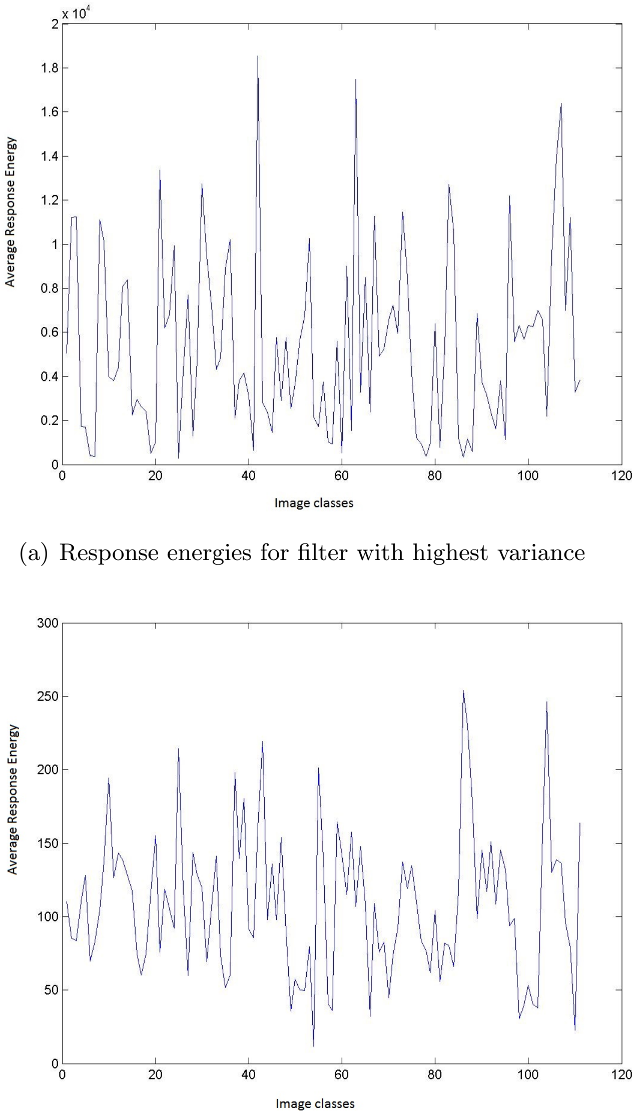


Figure 4.1: Filters sorted by the variance of their response energies. The filter with the highest variance is at the top left, with the one with the second highest variance to its right and so on.

Inspecting the average response energies of these filters reveals another problem. Figure 4.2(a) shows the average response energy of a filter with the highest variance and Figure 4.2(b) another with the lowest variance in their response energies, when applied to a set of training images of the Brodatz database. It is clear that the response energy of the filter with the highest variance is orders of magnitude larger than that of the filter with the lowest variance. Figure 4.3 shows the average response energy of each filter across the same training images, showing that the difference in scale of the response energies is not isolated to these two filters. Selecting filters based on the variance values obtained from such different scales of response energy will not select filters based on the significance of their variance across the image classes, but rather based simply on the scale of their response energies. We propose response scaling, as described below, as a method to solve this problem.



(a) Response energies for filter with highest variance

(b) Response energies for filter with lowest variance

Figure 4.2: Response energies for filters with the highest and lowest variance.

From each relevance class we select one or more training images. For each filter f_j and training image t_i we calculate the average response energy $V_i(j)$ as described in §3.4.1. We then find m_j, the maximum average response energy of f_j. We then scale each of the responses

$$V_i'(j) = \frac{V_i(j)}{m_j} \tag{4.1}$$

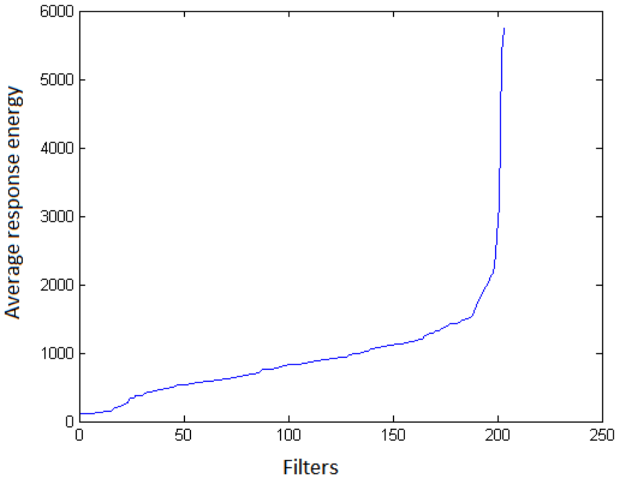


Figure 4.3: The average response energy of filters

These $V_i'(j)$ are then used for filter selection and feature calculation. This ensures that filter selection is based on the degree to which each filter response varies across texture classes, rather than their overall scales.

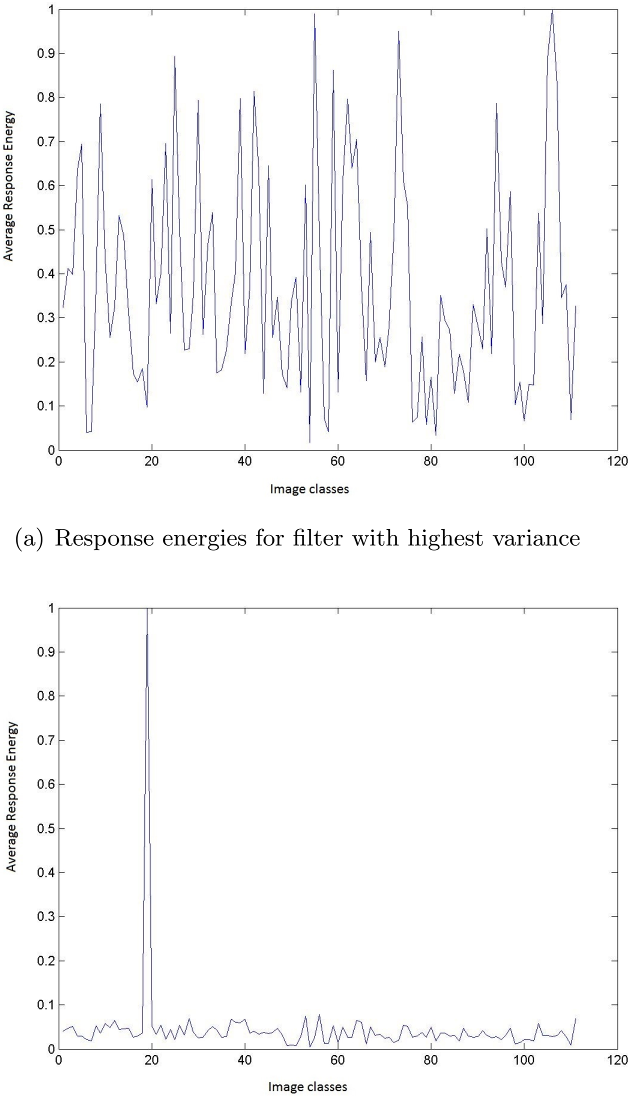
Figure 4.4 shows the scaled response energies of the filters with the highest and lowest variances. The energy range is now limited between zero and one and the filters can be selected based on the variances of their response energies, rather than the scales of the responses[1].

The utility of response scaling is shown in Figure 4.5 where we present CBIR results comparing the performance of all the filters with and without response scaling. For both the Brodatz and the VisTex collections, the improvement is significant without any filter selection being employed. This is easily understood by the fact that we are using Euclidean distance and by using response scaling we now let each entry of the feature vector contribute equally to the distance measure, whereas previously the distances would be skewed by certain entries of the feature vector due to their larger scales.

Figure 4.6 shows how response scaling a ffects the variance-based filter selection. We see the same trend that we saw in the previous chapter, which is the performance improving as more and more filters are selected. However comparing to the old results we can now see the features extracted using only 40 ICF selected with response scaling perform as well as the features extracted from all the ICF without response scaling. Features extracted from 80 ICF selected with response scaling has performance very close to using all the ICF with response scaling. Using the same filter set, just by doing a simple scaling of response energies we have been able to achieve a significant improvement in CBIR performance.

As can be expected, the variance-based filter selection technique selects a very different filter set when response scaling is used, as shown by the ordering in Figure 4.7. The results presented thus far shows that variance calculation after response scaling gives overall better CBIR performance. However the issue with this filter selection method is also well

[1] Figure 4.4(b) shows that the filter with the lowest variance has a higher sparsity compared to the filter with the high variance. We compare the performance of these sparse filters in Appendix E.



(a) Response energies for filter with highest variance

(b) Response energies for filter with lowest variance

Figure 4.4: Response energies for filters with the highest and lowest variance after response scaling.

illustrated in Figure 4.7. At least three of the first ten filters chosen are very similar but shifted (filters 3, 5, and 6) and it can be argued that there are two more that would also provide similar features (filters 4 and 9). Also, if a small number of filters are chosen ($\approx$ 40) the filters with the lowest variance, which may respond to small scale textures, are completely ignored. So, not only does this method lead to the selection of redundant features, but it also leads to the exclusion of potentially useful features. Our proposed clustering-based filter selection method described in §4.3 overcomes these issues. Now we discuss another change to the feature extraction method, introduced to deal with local intensity differences within an image which affects filter-based feature extraction.

4.2 Locally normalised convolution

Figures 4.8(a) and 4.8(b) show images belonging to the same relevance class from the VisTex collection. They have different intensities due to shadow. The energies of the responses to the filter in Figure 4.8(c) are shown in Figure 4.9. Parts of the image in Figure 4.8(a) that are in the shadow have a low response despite having similar texture leading to a large disparity in average filter response energy. The average energy value of the energy images shown in Figure 4.8 is 108 and 2.57×10^4 respectively. This is a very large difference due mainly to the local differences in image intensity.

We used Equation 4.2 to do global normalisation of our images, to make the pixels values zero mean and unit variance. The same normalisation was also applied to the filters.

$$I' = \frac{I(x,y) - \bar{I}}{\sigma(I)} \tag{4.2}$$

Where I is an image, and I' is the normalised image. This is a standard method of normalising for global image intensity, however it does not compensate for local intensity differences. What we need is to do normalisation on local image regions as the filter is being convolved with them. Normalised cross-correlation as defined in Equation 4.3 does just such a normalisation.

$$C_{I,t}(a,b) = \frac{\sum_{x,y}[I(x,y) - \bar{I}_{u,v}][t(x-u,y-v) - \bar{t}]}{\sqrt{\sum_{x,y}[I(x,y) - \bar{I}_{u,v}]^2 \sum_{x,y}[t(x-u,y-v) - \bar{t}]^2}}, \tag{4.3}$$

where t is the template and $\bar{t}$ is the mean of the template. I is the image and $\bar{I}_{u,v}$ is the mean of the region of the image under the template (Lewis, 1995).

We perform locally normalised convolution (LNC) by doing normalised cross-correlation, but by rotating the filter by 180°. So, the LNC energy image of image I_i and filter f_j can be found by:

$$E_{I_i,f_{j_{LNC}}} = C^2_{I_i,\hat{f}_j} \tag{4.4}$$

where $\hat{f}_j$ is f_j rotated by 180°.

Figure 4.10 shows that by using this locally normalised convolution we can indeed detect texture patterns that were missed earlier.

We now modify the feature extraction steps mentioned in §3.4.1, where for filter f_j and image I_i, instead of calculating the average response energy using two dimensional convolution we use the average LNC response energy,

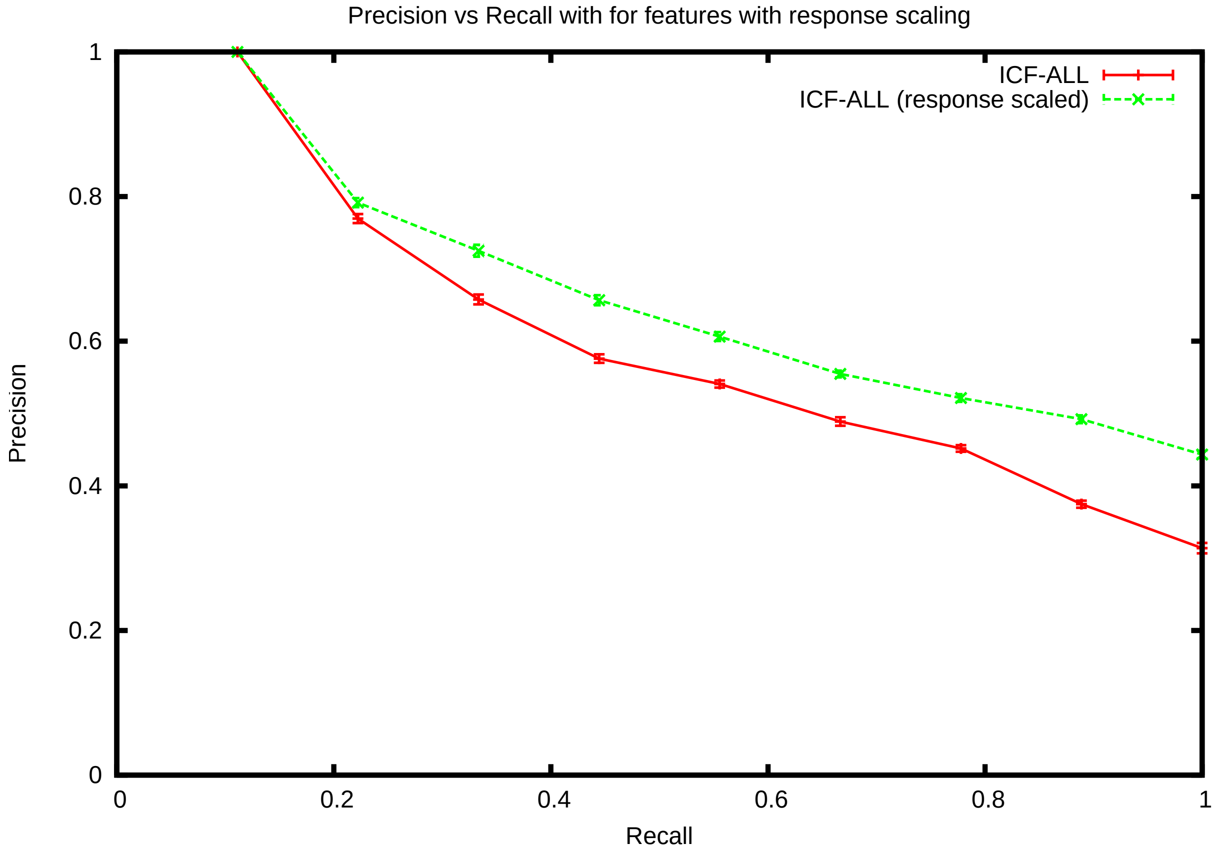
$$V_{i_{LNC}}(j) = \frac{\sum_x \sum_y E_{I_i,f_{j_{LNC}}}(x,y)}{size(E_{I_i,f_{j_{LNC}}})} \tag{4.5}$$

Figure 4.11 shows the results of using LNC to extract features using all the ICF, instead of two dimensional convolution. Note that the LNC results shown here do not include response scaling. It is evident that just by using LNC it is possible to achieve a similar amount of improvement as using response scaling. This can be attributed to both identifying texture features which were not found by two dimensional convolution, and to each entry in the feature vector being within the range $0 - 1$ and therefore not being skewed by differences in the scales of the response energies.

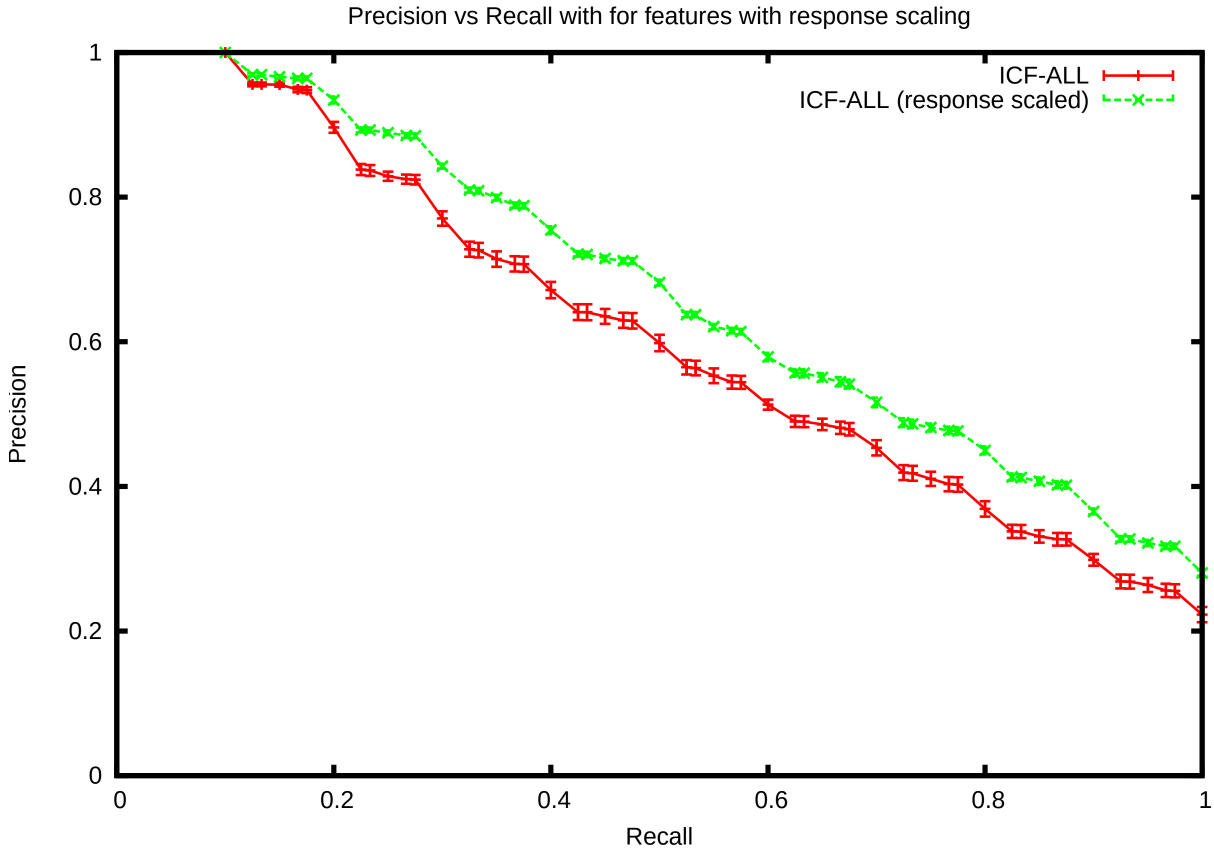
The most significant improvement however comes when we change the feature extraction to use LNC and response scaling as shown in Equation 4.6, where $m_{j_{LNC}}$ is the maximum average LNC response energy of filter j found from the training images.

$$V'_{i_{LNC}}(j) = \frac{V_{i_{LNC}}(j)}{m_{j_{LNC}}} \tag{4.6}$$

Figure 4.12 shows that combining these two steps gives consistently good results across the two image collections were are testing against. From Figure 4.13 we see that the improvements are not limited only to ICF, but are equally applicable to Gabor filters.

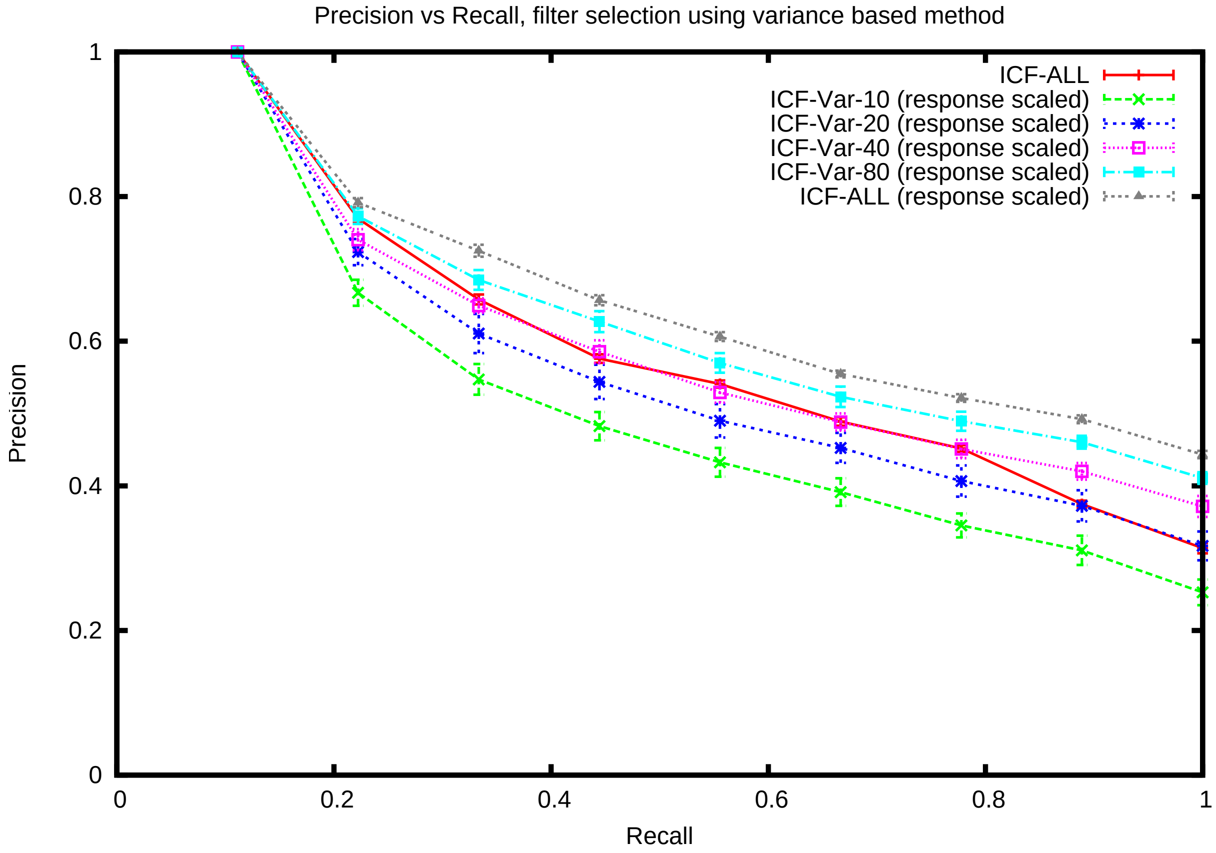


(a) Brodatz

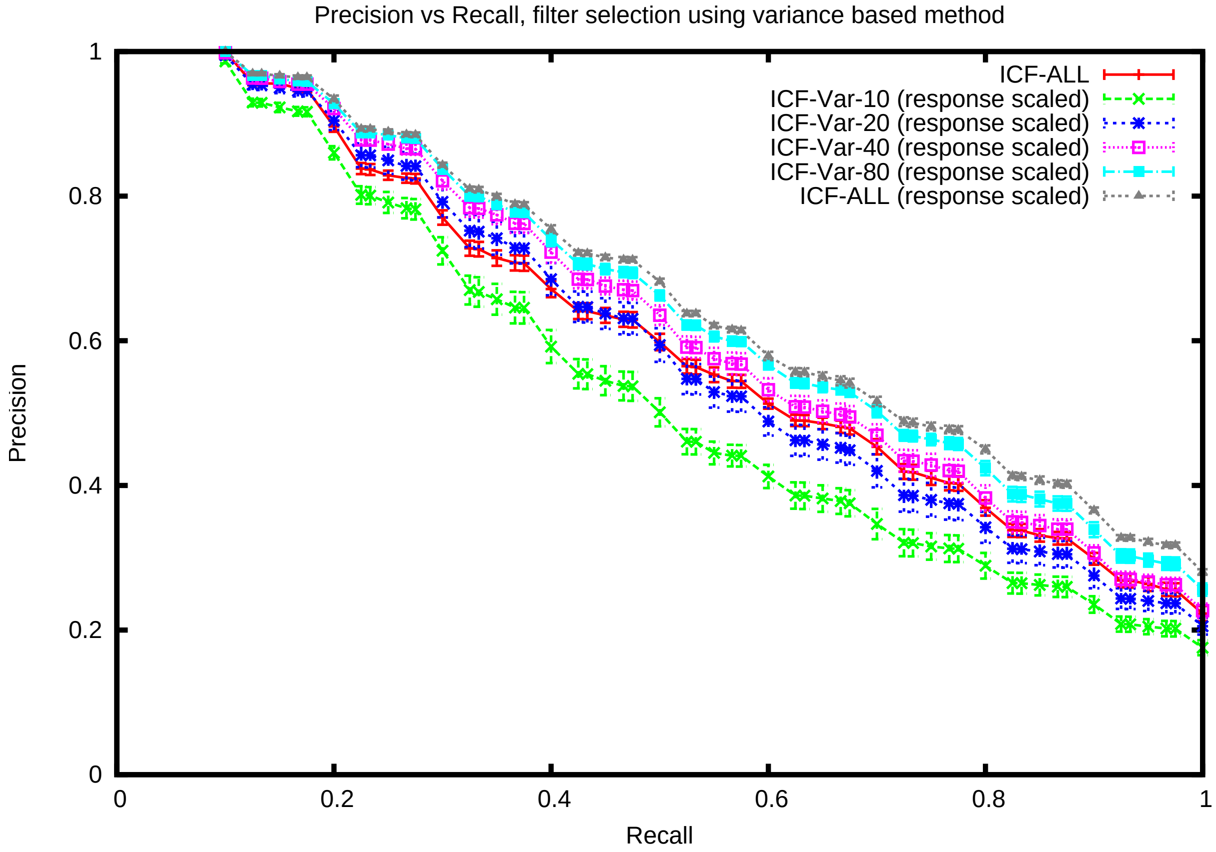


(b) VisTex

Figure 4.5: Increased performance after response scaling.



(a) Brodatz



(b) VisTex

Figure 4.6: Better performance of the selected filters after response scaling.

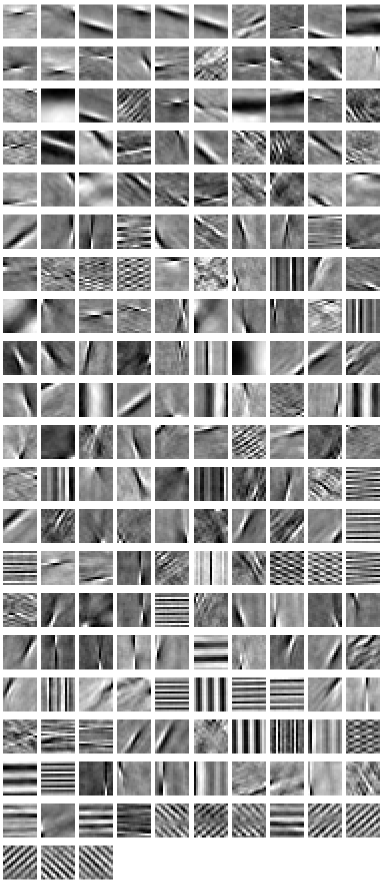
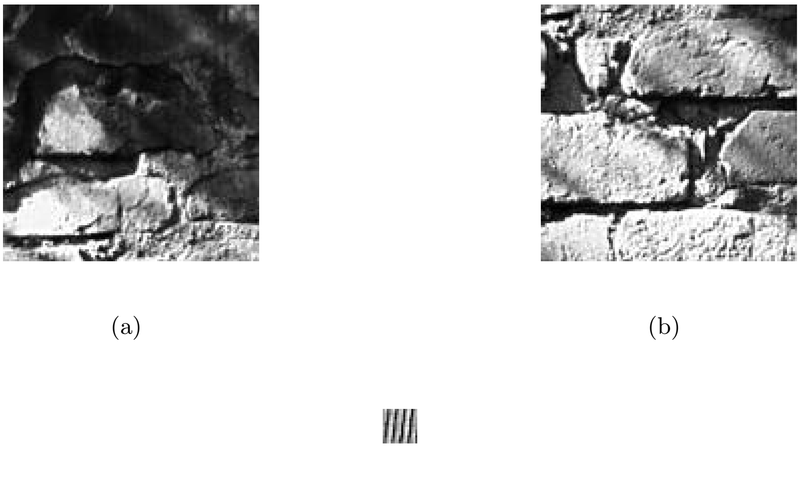


Figure 4.7: Filters sorted by the variance of their response energies after response scaling.

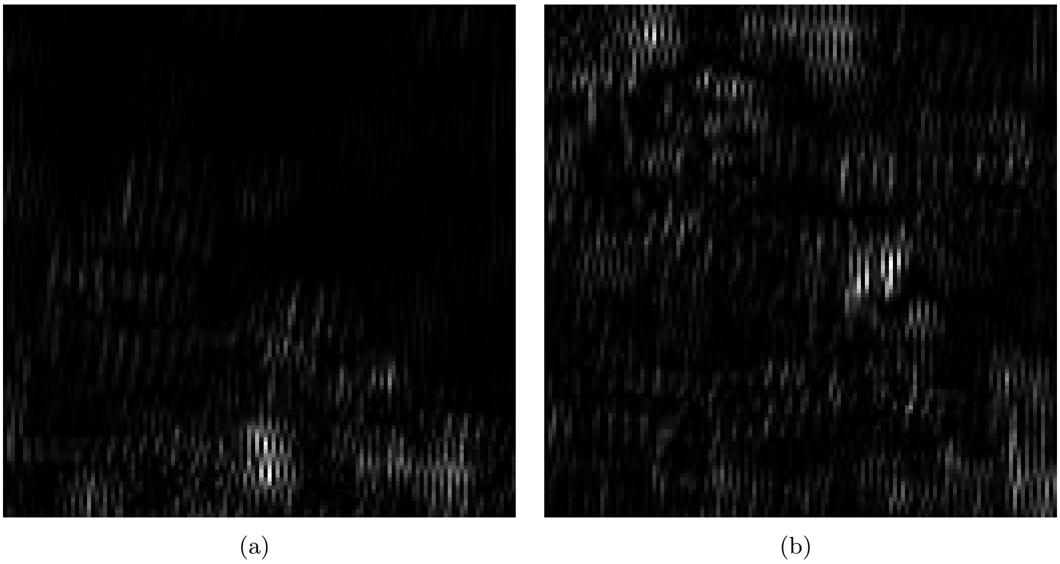


(a)

(b)

(c)

Figure 4.8: Two relevant images with localised differences in intensity. Figure 4.9 shows the response energy when these two images are convolved with the filter shown in Figure 4.8(c)

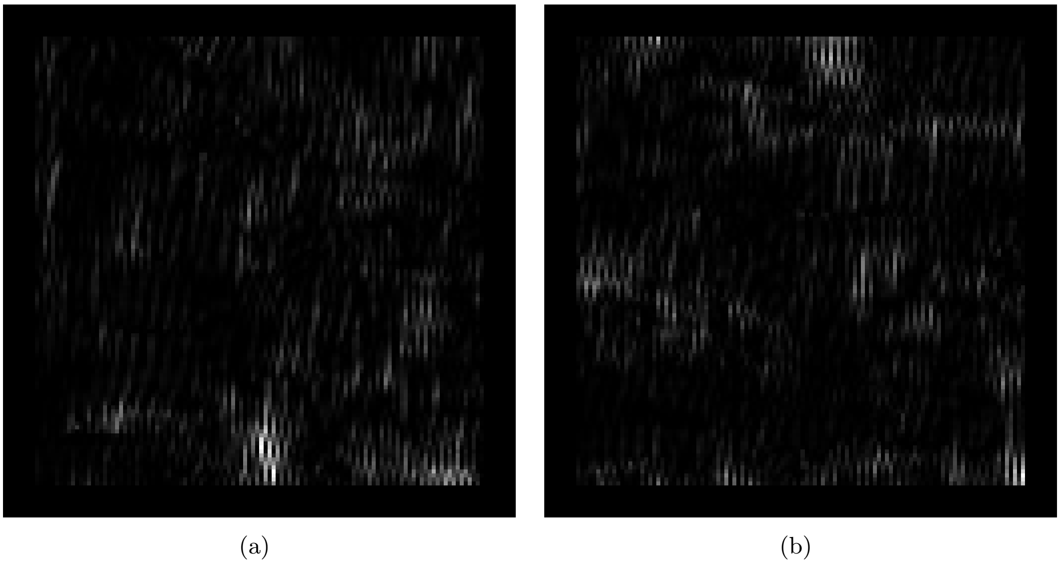


(a)

(b)

Figure 4.9: Filter energies when the images in Figures 4.8(a) and 4.8(b) are convolved with the filter in Figure 4.8(c)

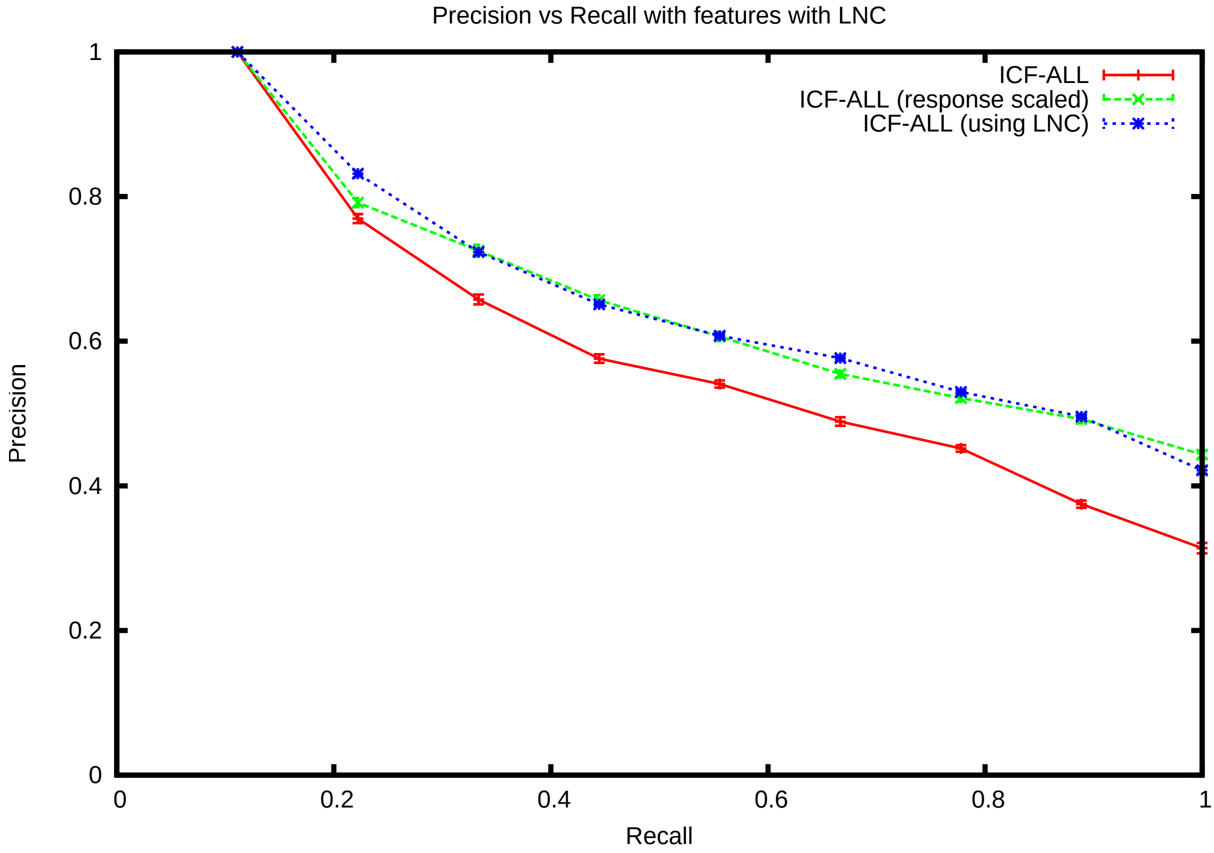
. Resized for visualisation.



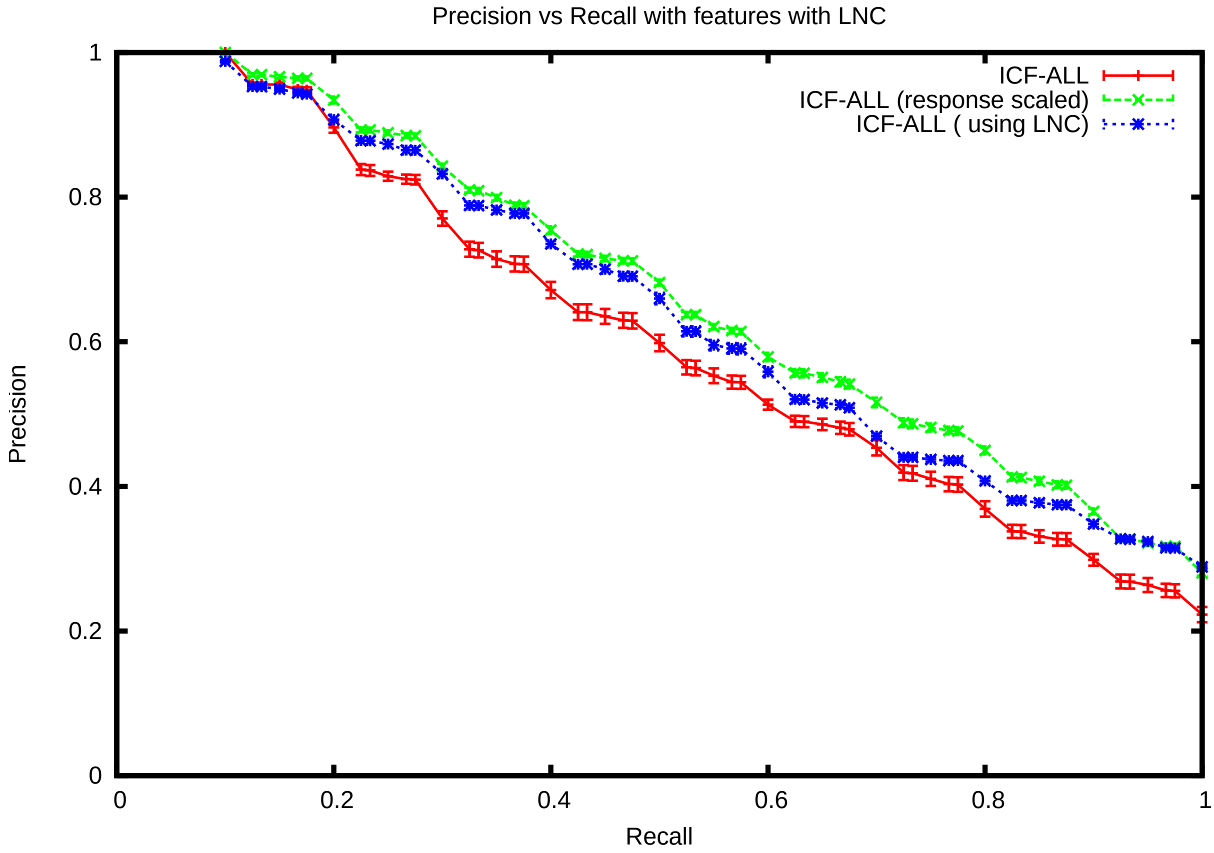
(a)

(b)

Figure 4.10: Filter energies when the images in Figures 4.8(a) and 4.8(b) are convolved with the filter in Figure 4.8(c) using locally normalised convolution. Resized for visualisation.

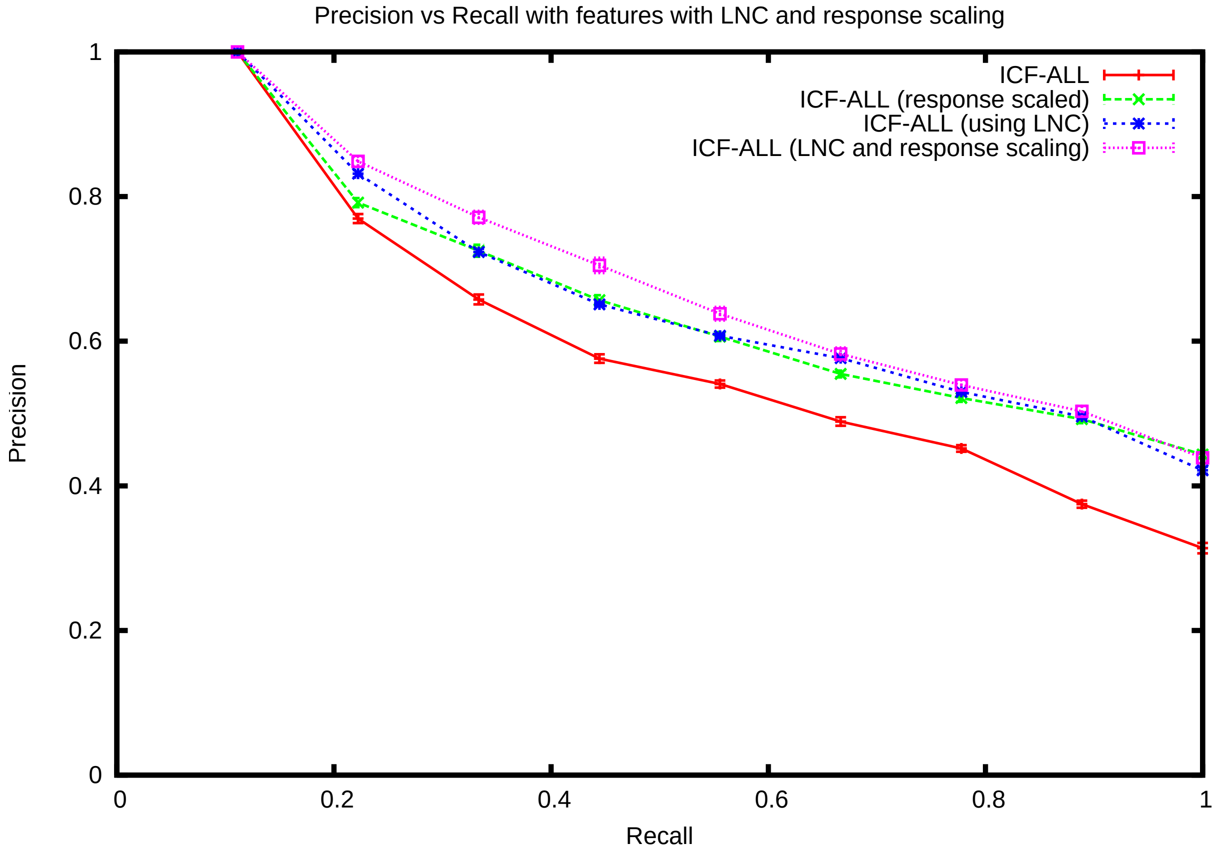


(a) Brodatz

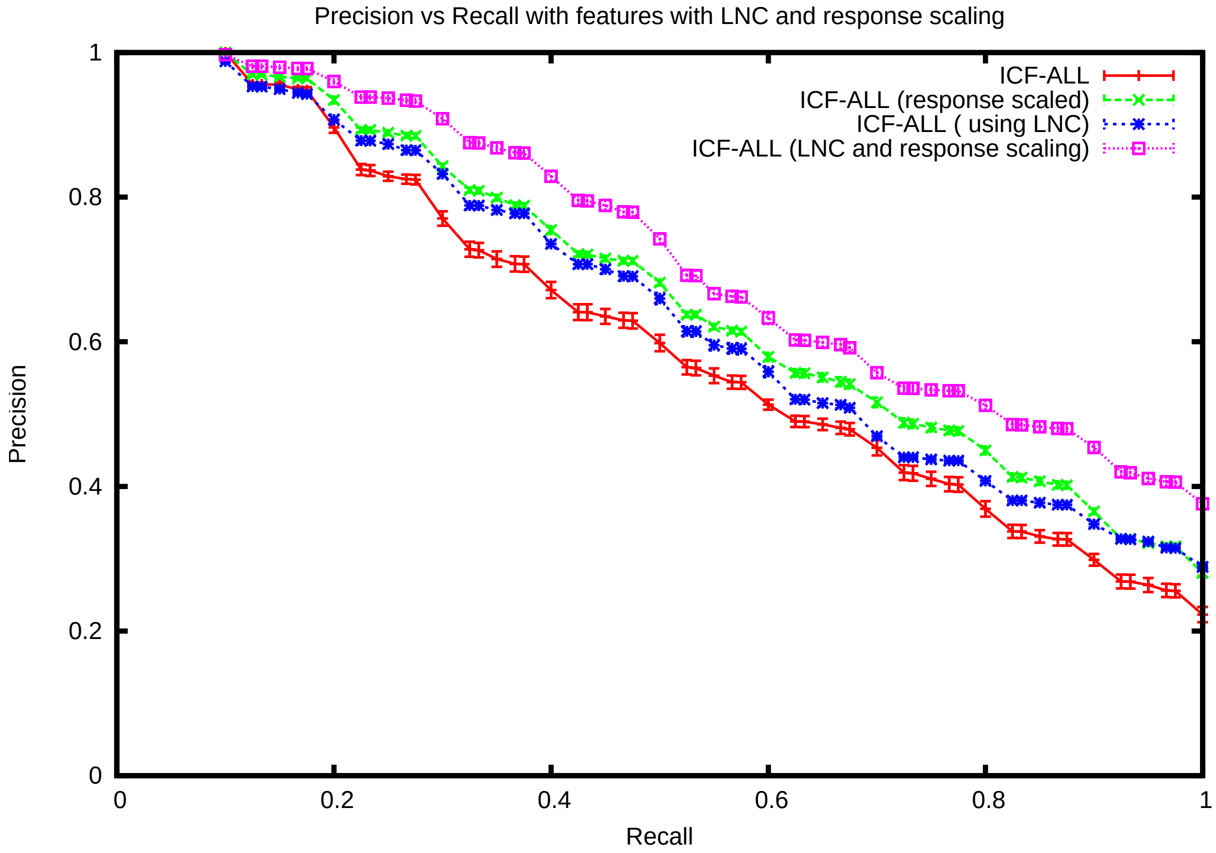


(b) VisTex

Figure 4.11: Increased performance after using LNC.

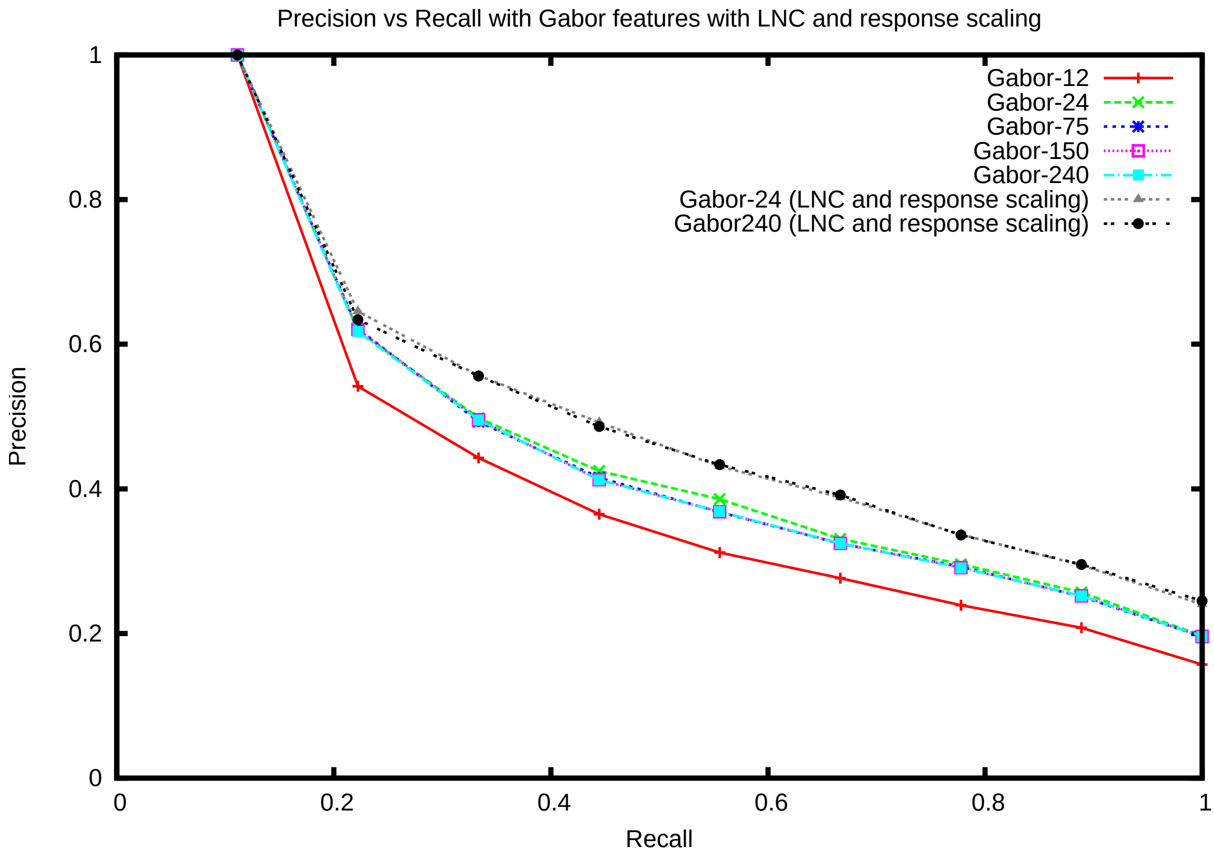


(a) Brodatz

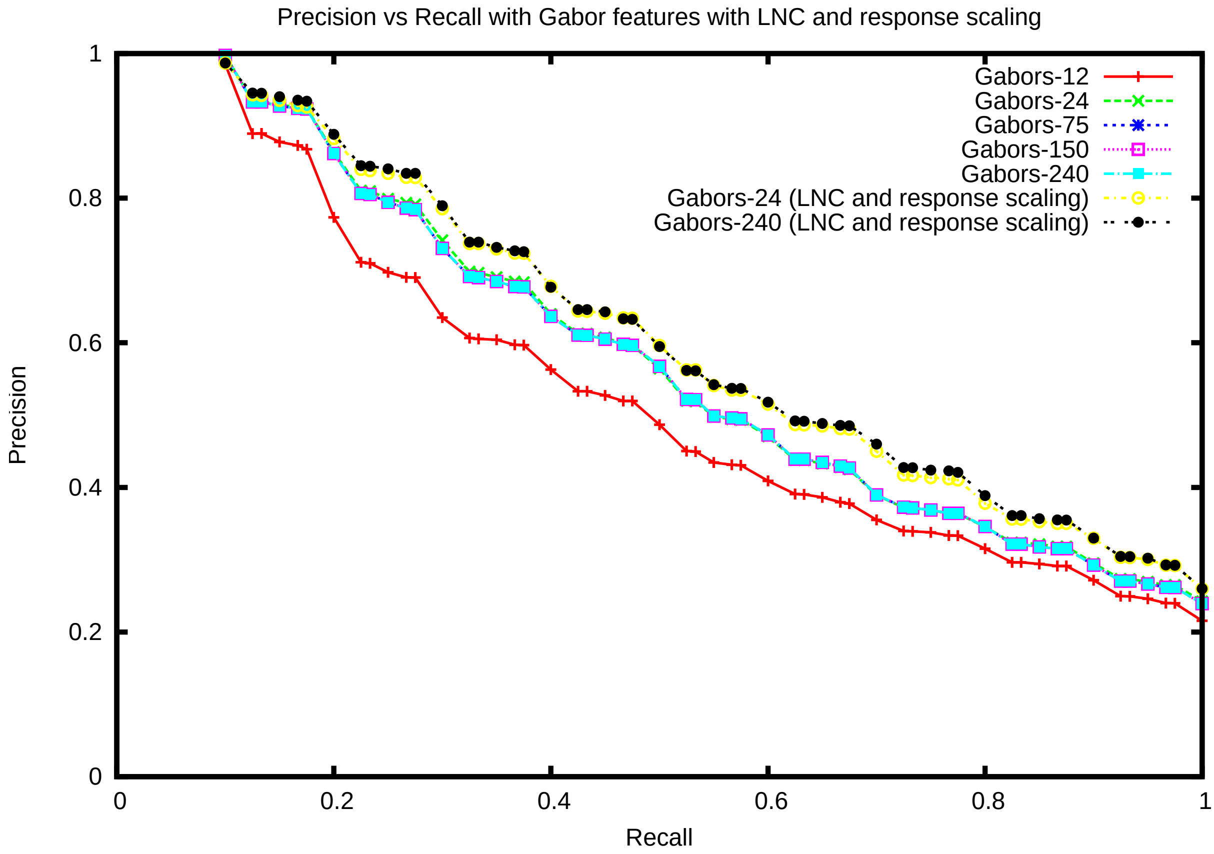


(b) VisTex

Figure 4.12: Increased performance after using LNC and response scaling.

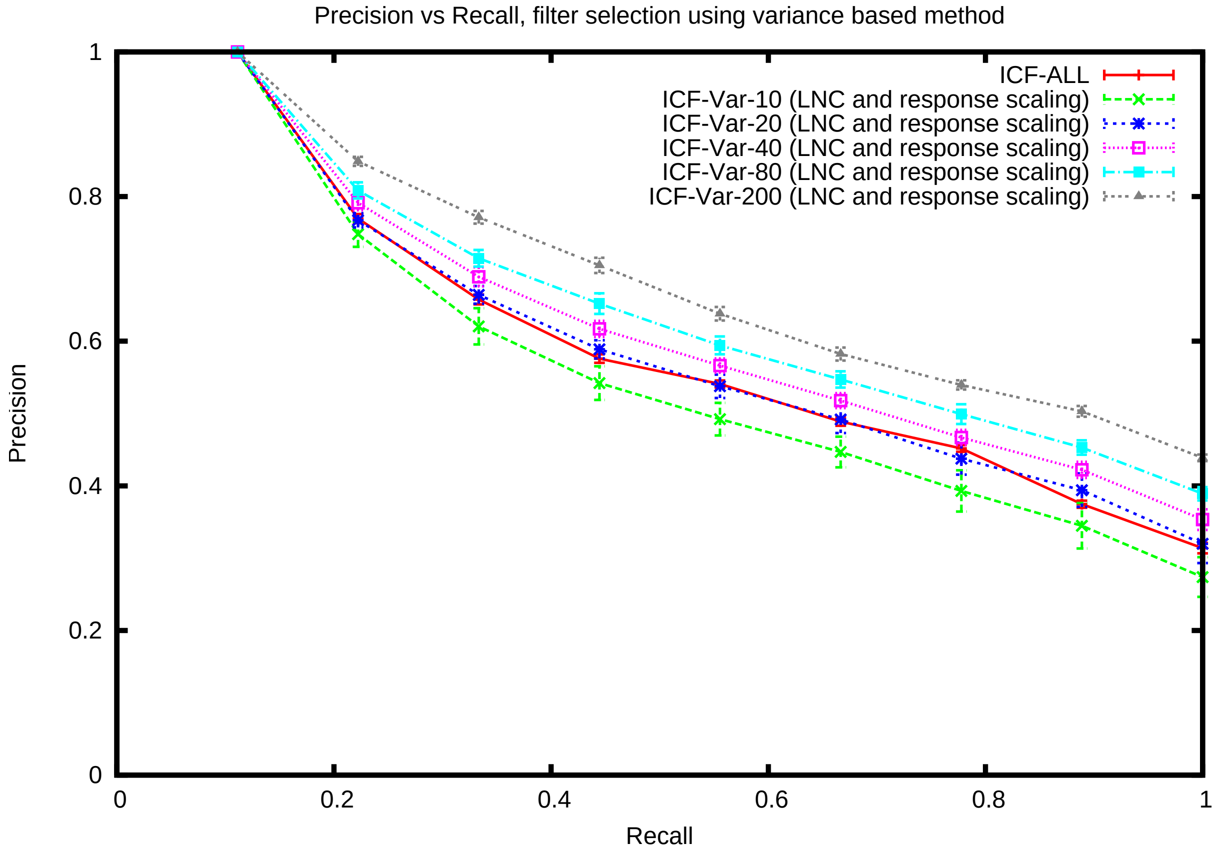


(a) Brodatz

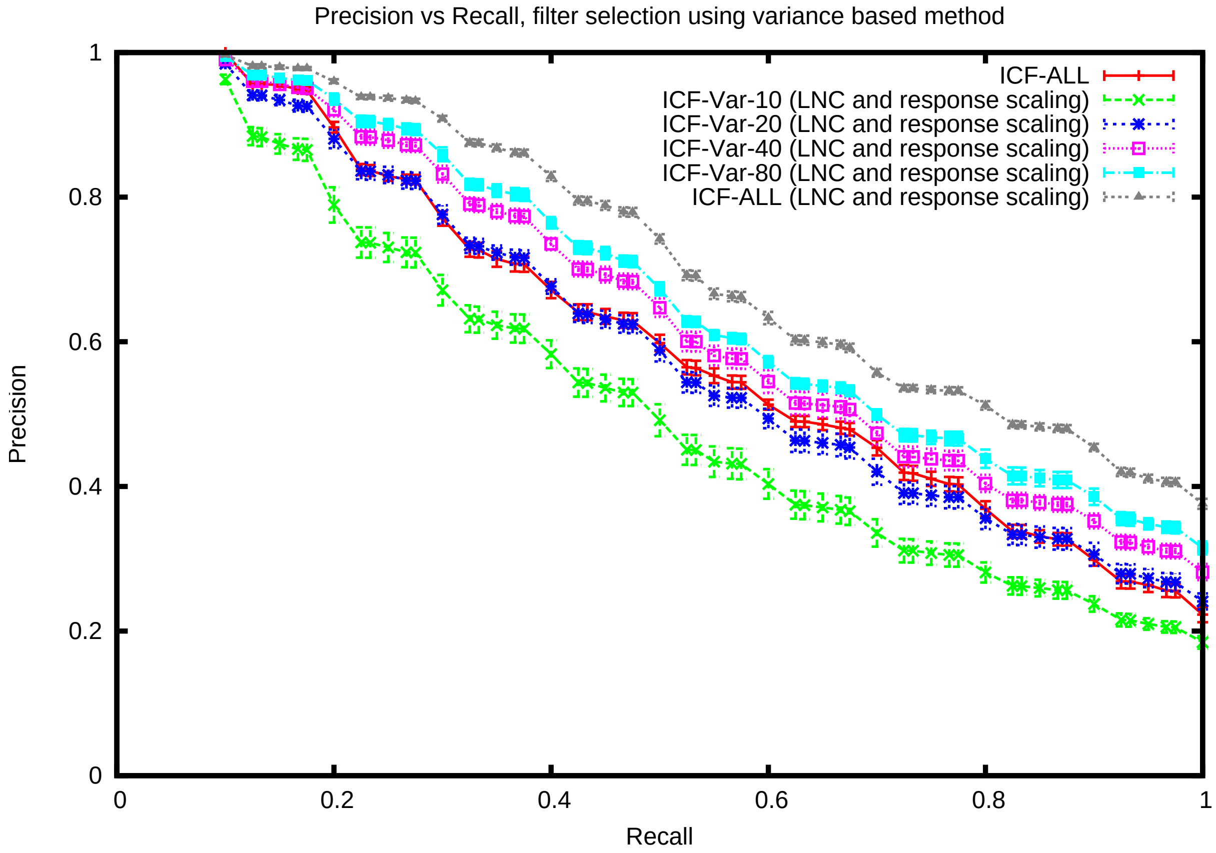


(b) VisTex

Figure 4.13: These changes also increase the performance of the Gabor filters.



(a) Brodatz



(b) VisTex

Figure 4.14: LNC and response scaling resulting in improved performance for the variance-based filter selection.

4.3 Cross-correlation and clustering based filter selection

We concluded §4.1 by explaining a shortcoming of the variance-based filter selection method. To recapitulate, Figure 4.7 shows filters sorted by the variance of their response energies from highest to lowest. The variance-based filter selection method would choose the top N filters, where N is the number of filters that needs to be chosen from the whole set. We have discussed in previous chapters how ICA is known to extract ICF which are approximately shifted/duplicate versions of each other. When used as filters, these duplicate versions give very similar responses, leading to redundant features. We also pointed out that by choosing filters in order from a sequence sorted by the variance, the method is susceptible to leaving out a class of filters which may extract texture features at different scales which may prove to be useful. The goal then is to find a method which will choose a smaller subset of filters without having shifted/duplicate versions and which will include as many filters as possible that lead to different types of non-redundant features. Our method described here concentrates on attaining the first goal and as a result also achieves the second one to a certain extent.

4.3.1 Filter similarity measure

Any attempt to reduce the selection of shifted/duplicate filters would require some kind of metric indicating the similarity of filters. The filters are initially components extracted using ICA, so, as vectors they are statistically as independent as possible. However the problem of the shifted/near-duplicate filters manifests when we re-form the components into patches and interpret them as filters. We need a similarity metric that will take into account the filter shapes when interpreted as filters. From the examples shown in Figures 4.7 and 4.1 we see that it would be possible to use the variance of response energies as a potential measure. However this metric depends on the filter responses rather than the actual impulse response structure of the filter. We define similarity between filters as the maximum absolute value of their normalised cross-correlation matrix as this will be high for filters that are approximately shifted versions of each other. For filters f_i and f_j, their similarity measure $S(i,j)$ is found from the following

$$S(i,j) = max(|C_{f_i,f_j}|) \tag{4.7}$$

where C_{f_i,f_j} is the two dimensional normalised cross-correlation matrix of f_i and f_j as defined by Equation 4.3.

4.3.2 Clustering and filter selection

Clustering filters

If similar filters are clustered together, it makes it easy to choose one from them. We wanted a clustering algorithm that would impose a strict within-cluster similarity requirement. We also wanted a clustering algorithm which would give repeatable results every time it was executed with the same group of filters. K-means has a random initial condition and also requires a sensible definition of a mean, which introduced some difficulties. Given that we already had a similarity matrix, we can easily use one of the agglomerative hierarchical clustering techniques, which would give us repeatable results . Between single-link, average-link and complete-link, complete-link clustering imposes a stricter inter cluster distance measure, where the distance between two clusters is the maximum of all the pairwise distances of the cluster members (Jain et al., 1999). As the process of merging clusters looks at clusters with the minimum distance, imposing this strict criteria gives a better chance of each cluster having filters which are highly similar as judged by Equation 4.7.

If ICA extracts N filters, then for each filter f_i where $1 \leq i \leq N$, we calculate its similarity with every other filter using normalised cross-correlation as mentioned above. Using these values we construct a $N \times N$ similarity matrix S which is used as a measure of the similarity between f_i and f_j. The distance matrix is easily calculated as $D = 1 - S$. When filters are grouped together in clusters, the distance between two clusters CL_1 and CL_2 is measured as the maximum of all pairwise distances between the filters of CL_1 and CL_2. As our implementation uses a similarity matrix, the similarity between two clusters CL_1 and CL_2 is measured as the lowest of all pairwise similarities between the filters of these clusters. The implementation of complete-link clustering, shown in Algorithm 2, is an adaptation of a version shown in Jain et al. (1999). It uses a threshold similarity value to group filter clusters together. The appropriate similarity threshold that will group the filters into the desired number of clusters is found by a simple binary search of values between 0–1. The algorithm has an upper limit of 1000 attempts, but in reality we find the desired number of clusters within a few iterations.

Figure 4.15 shows examples of some filter clusters when the entire filter set was grouped into 40 clusters. For the first three clusters we see that our method has successfully grouped very similar filters together. In the fourth example we see filters which are mostly edges at an angle, however there is a visual difference between some of the filters within the cluster.

Filter selection from each cluster

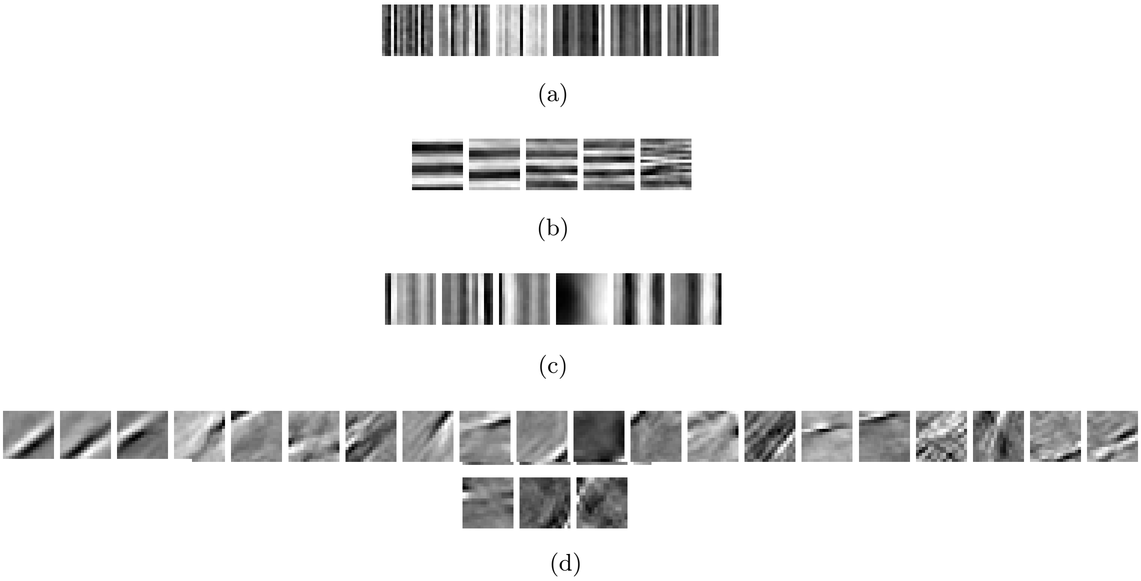
Once the filters have been separated into clusters we need to choose one filter from each cluster. Our consideration was to choose a filter that would be representative of the cluster when viewed as a filter (instead of a vector, as in the original independent component). We again made use of the similarity measure shown in Equation 4.7. From each cluster, we calculate the average similarity of each filter with other filters in the same cluster, as shown in Equation 4.8:

$$AS_i = \frac{\sum_{j|f_j \epsilon CL_k, j \neq i} S(i,j)}{size(CL_k) - 1} \tag{4.8}$$

We simply choose the filters with the highest average similarity from each cluster.

Algorithm 2 The complete-link clustering algorithm

```
$maxThreshold \leftarrow 1$
$minThreshold \leftarrow 0$
$desiredNumberOfFilters = the\ required\ number\ of\ filters.$
$S = Filter\ similarity\ matrix.$
$count \leftarrow 0$
while ($count < 1000$ $and\ number\ of\ clusters \neq desiredNumberOfFilters$) do
    set each filter into its own cluster
    set cluster similarity matrix to $S$
    $threshold = \frac{maxThreshold+minThreshold}{2}$
    repeat
        merge clusters together that have a similarity higher than the threshold.
        recalculate the similarity matrix for clusters.
    until There were no merges
    $numberOfClusters \leftarrow new\ number\ of\ clusters$
    if $numberOfclusters < desiredNumberOfClusters$ then
        $maxThreshold \leftarrow threshold$
    else
        if $numberOfclusters > desiredNumberOfClusters$ then
            $minThreshold \leftarrow threshold$
        end if
    end if
    $count \leftarrow count + 1$
end while
```


(a)

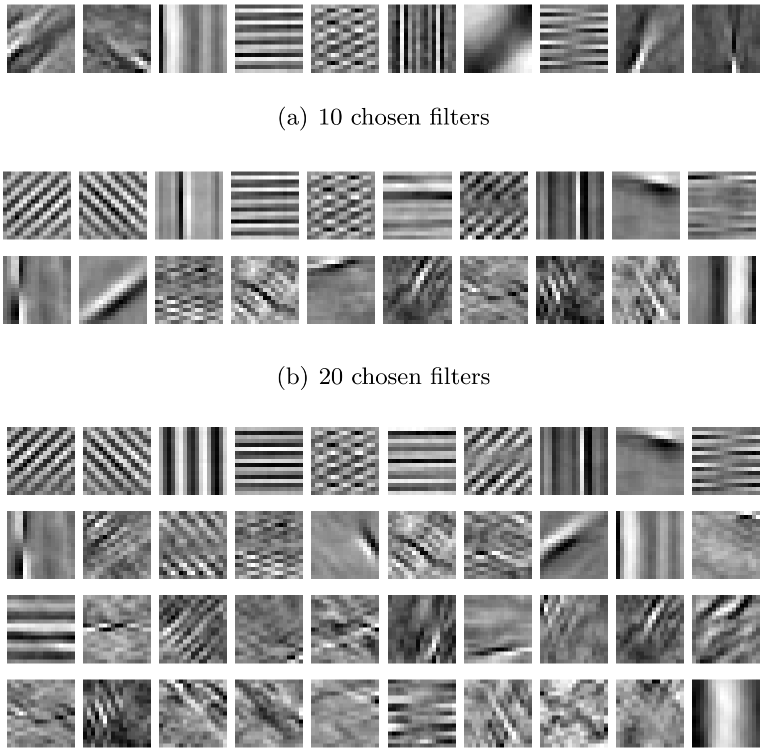
(b)

(c)

(d)

Figure 4.15: Examples of some filter clusters found by our implementation of complete-link clustering which uses the filter similarity measurement shown in Equation 4.7

Figure 4.16 shows $10, 20$ and 40 filters chosen using this method. We can see that there is a significant reduction in duplicate/similar filters being selected compared to the results of variance-based filter selection (Figure 4.7). Also, more filters that respond to textures at small scales are included.



(a) 10 chosen filters

(b) 20 chosen filters

(c) 40 chosen filters

Figure 4.16: 10, 20, and 40 filters chosen using our proposed clustering approach. A comparison with Figure 4.7 shows that the filters chosen have a lot more variation.

Figure 4.17 shows a comparison of the performance of $10, 20$ and 40 filters selected using the variance-based method and our clustering-based method. The results from the clustering-based method is marked as "ICF-CLC-N", where N is the number of filters chosen. Other labels follow the format described in Table 3.1. All results displayed use LNC and response scaling, so there is no indication shown for these techniques.

Our proposed clustering-based method consistently outperforms the variance-based method for both the image collections. In fact we can see that 10 filters chosen using our method performs comparably with 40 filters chosen using the variance-based method. This leads to significant computational savings in feature computation and comparison.

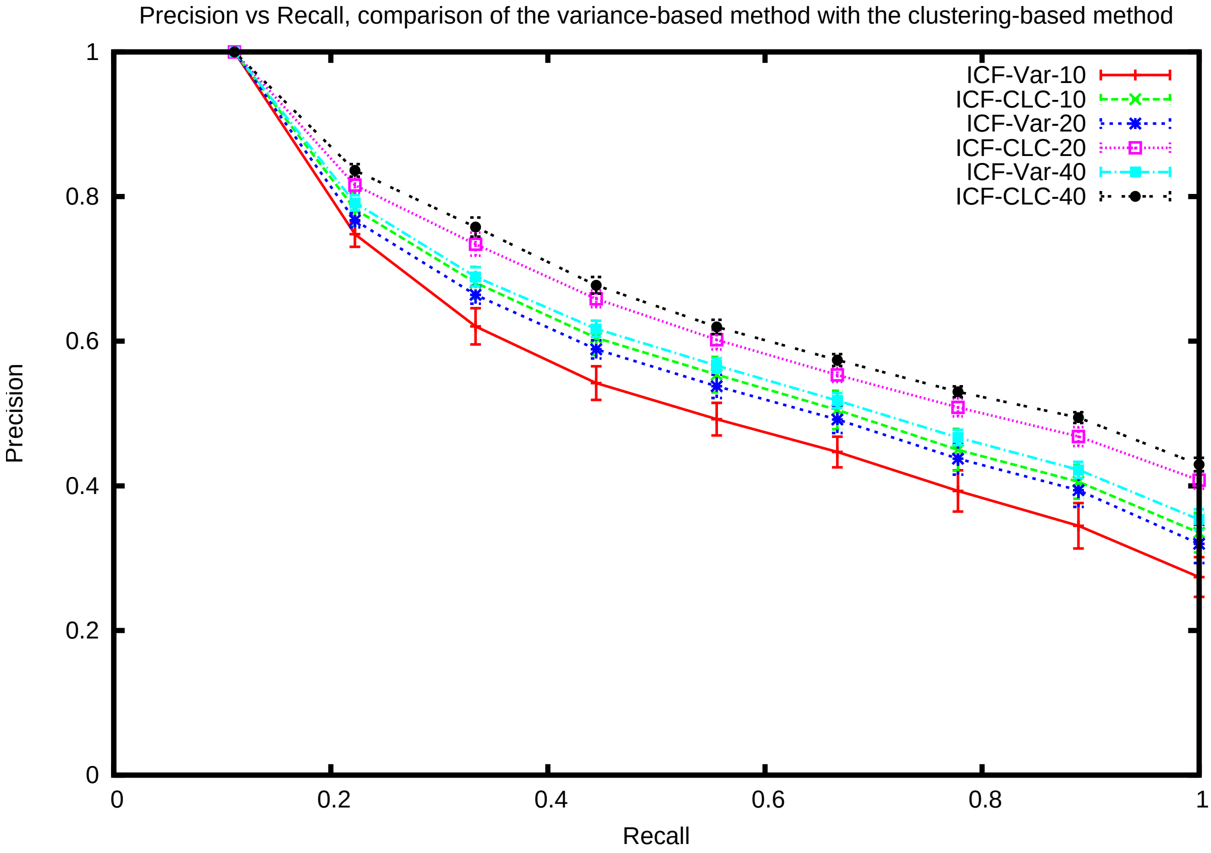
Figure 4.18 compares the the performance of 40 and 80 filters chosen using our clustering-based technique with that of using all the ICF. We see that ICF-CLC-40 perform comparably with ICF-ALL. For the VisTex database ICF-CLC-80 seem to have a slightly better performance compared to ICF-ALL. Compared with the results of the 40 and 80 filters chosen using the variance-based method, this is a large improvement. The results match our expectations, as our method chooses a filter set with less redundancy. These selected filters respond to more varied textures compared to the variance-based filters, thus leading to improved performance.

4.4 Conclusion

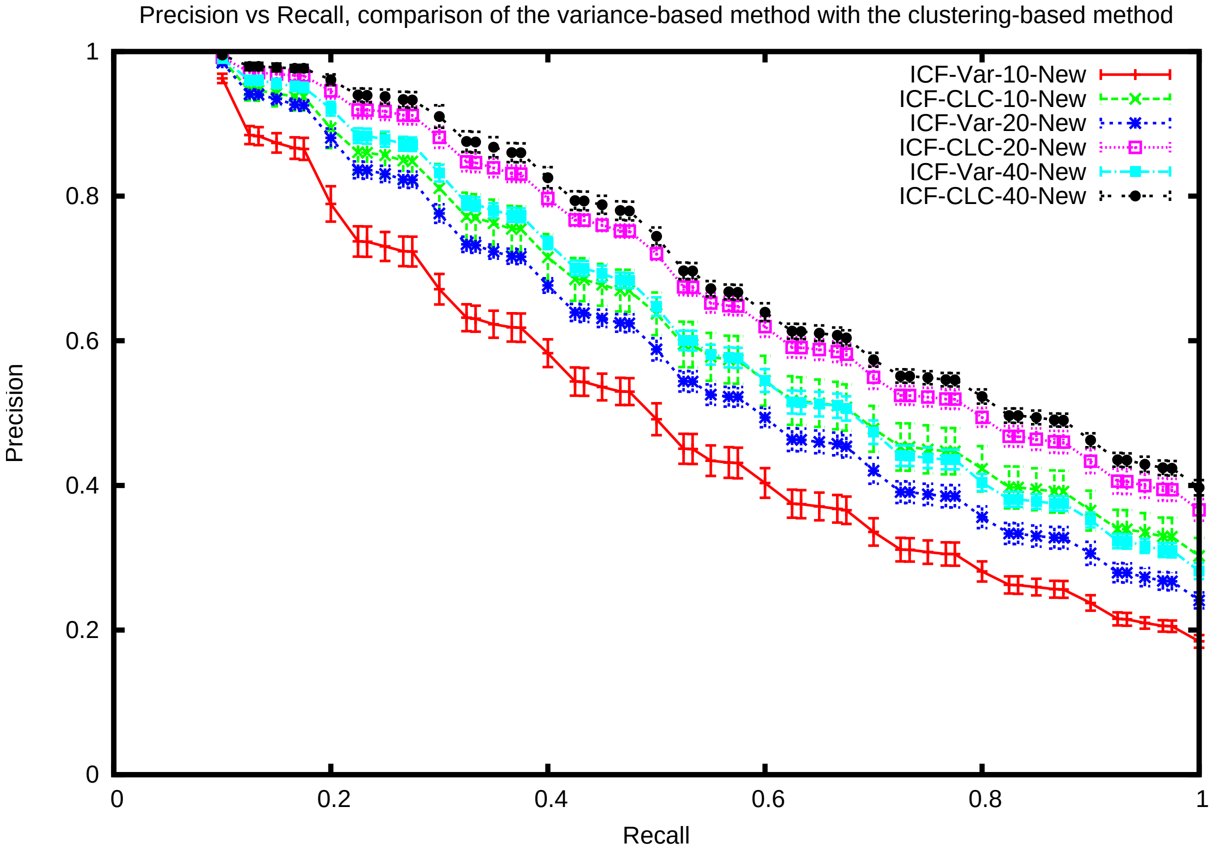
In this chapter we have presented two techniques, response scaling and locally normalised convolution, which improve the performance of global features extracted using filters. We have shown that by using response scaling we address the problem of filter responses differing by orders of magnitude, which had an adverse affect on the variance-based filter selection technique and also on feature comparison. We have shown that LNC allows us to identify textures which would otherwise be ignored. We have shown that LNC and response scaling both individually improve CBIR performance, but the most significant performance improvement comes when they are combined. These improvements are present when using all the ICF, and also when selecting a smaller sub-set of filters using the variance-based method. We have found that combining these two steps also improves the performance of the Gabor filters.

We have demonstrated a problem with variance-based filter selection: it is susceptible to choosing filters which are shifted/duplicate versions of each other, leading to redundant features. As a solution we have proposed a new method to select filters, which uses normalised cross-correlation as a method to measure filter similarity and then uses complete-link clustering to group similar filters together. Our heuristic to choose filters from each cluster gave filter sets that had less redundancy and we have demonstrated through experimentation that our method leads to filter selection which has better CBIR performance compared to the variance-based method.

With this chapter we conclude our development of ICF-based global features. We have shown in Chapter 3 that grid-based local features have poor performance on the class of images we are testing against. In the next chapter we present an adaptive local feature which uses salient points and the “bag of words” model. These perform much better than grid-based local features.

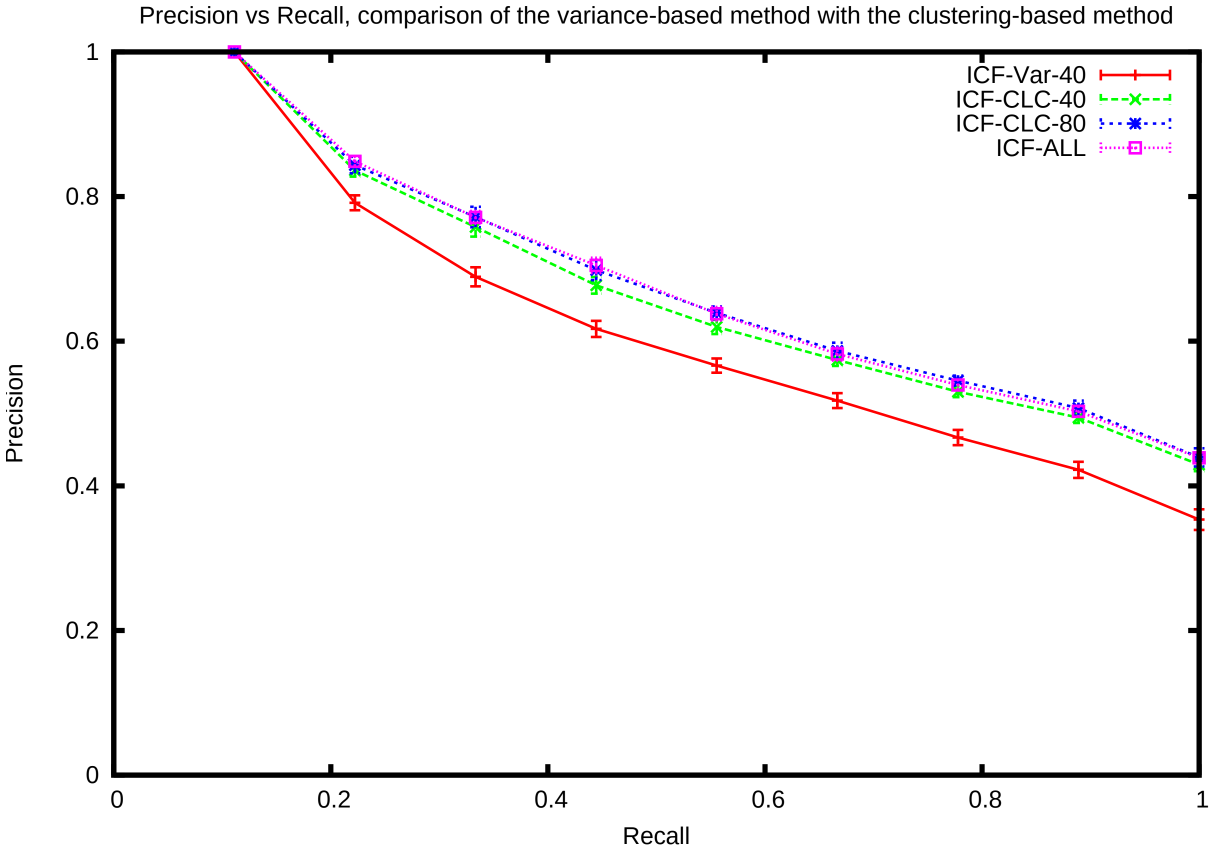


(a) Brodatz

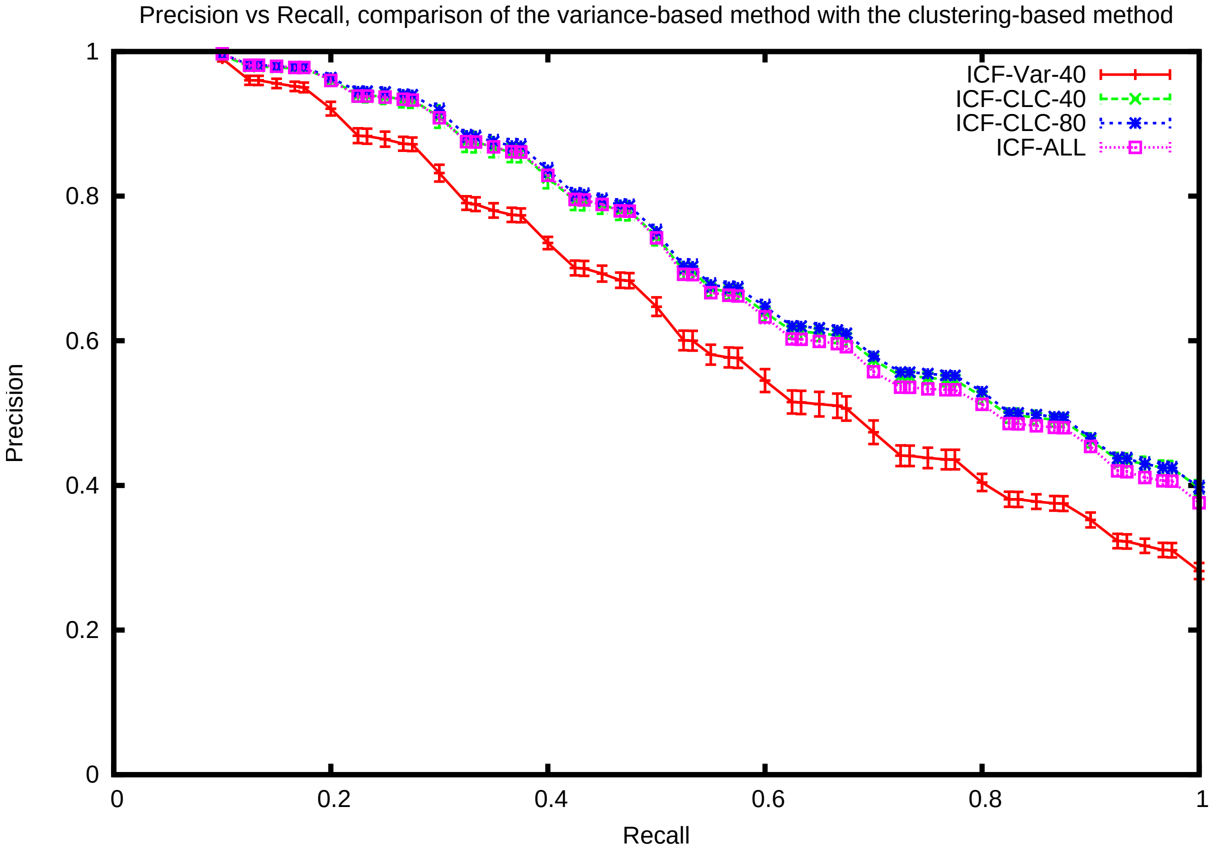


(b) VisTex

Figure 4.17: Results comparing the performance of the variance-based filter selection to the clustering-based filter selection.



(a) Brodatz



(b) VisTex

Figure 4.18: Results comparing the performance of the variance-based filter selection to the clustering-based filter selection.

Chapter 5

ICFSIFT:Local features that improve performance

5.1 Introduction

In this chapter we build on work described in Chapters 3 and 4. That work used global features, with some grid-based local features presented in Chapter 3. These grid-based features (average filter response energies, GLCM, etc.) were computed on image regions after the image was divided into a fixed set of rectangular subregions. We showed that these features performed poorly on the VisTex and Brodatz image collections, which have globally consistent texture. We also showed that one of the reasons for the poor performance of these features was the encoding of fixed locations in the feature vectors. In this chapter we propose the use of a different set of local texture features, which are extracted by identifying salient points in images. We show that these salient point-based local features perform better than global features, even for images with globally consistent texture. We present results using the popular Scale Invariant Feature Transform (SIFT) feature (Lowe, 1999) and then propose a new-ICF based feature—ICFSIFT—which utilises SIFT keypoints, but employs descriptors that are adapted to the image collection. Our proposed ICFSIFT features perform consistently better than the standard SIFT features and also the global ICF features. We further illustrate the utility of the ICFSIFT features by combining them with global features to achieve an even further improved performance. The ICFSIFT features lead to a large improvement in the performance of local features for this category of images, compared to the results presented earlier.

5.2 ICFSIFT

As with our previous ICF-based features, there are two distinct stages in learning and using the ICFSIFT features. In the first stage we learn the ICF from an image collection. The second stage can be further divided into two parts, (i) extracting the ICFSIFT features and, (ii) creating the image descriptor. All these steps involve extracting patches from a region around each keypoint. We describe patch extraction first, then filter learning, ICFSIFT feature extraction, and image descriptor creation.

5.2.1 Extracting keypoint patches

From an image I we first extract its keypoints. For each keypoint l_k, the following information is of interest:

- The keypoint scale (s_k).
- The keypoint orientation (θ_k).
- The keypoint location at the relevant scale ((x_k, y_k)).

We resize the image to the scale s_k to obtain the re-scaled image I^{s_k}. We want to extract 17×17 patches, so from I^{s_k} we extract a patch of size $17\sqrt{2}$ centered on (x_k, y_k). We rotate this patch by θ_k and from this rotated patch we extract a 17×17 patch p_k. Thus we get patches from the relevant scale for which the orientation has been normalised. It is important to note that our method extracts a patch from a scaled version of the image, not the DoG images. Figure 5.1 depicts this process.

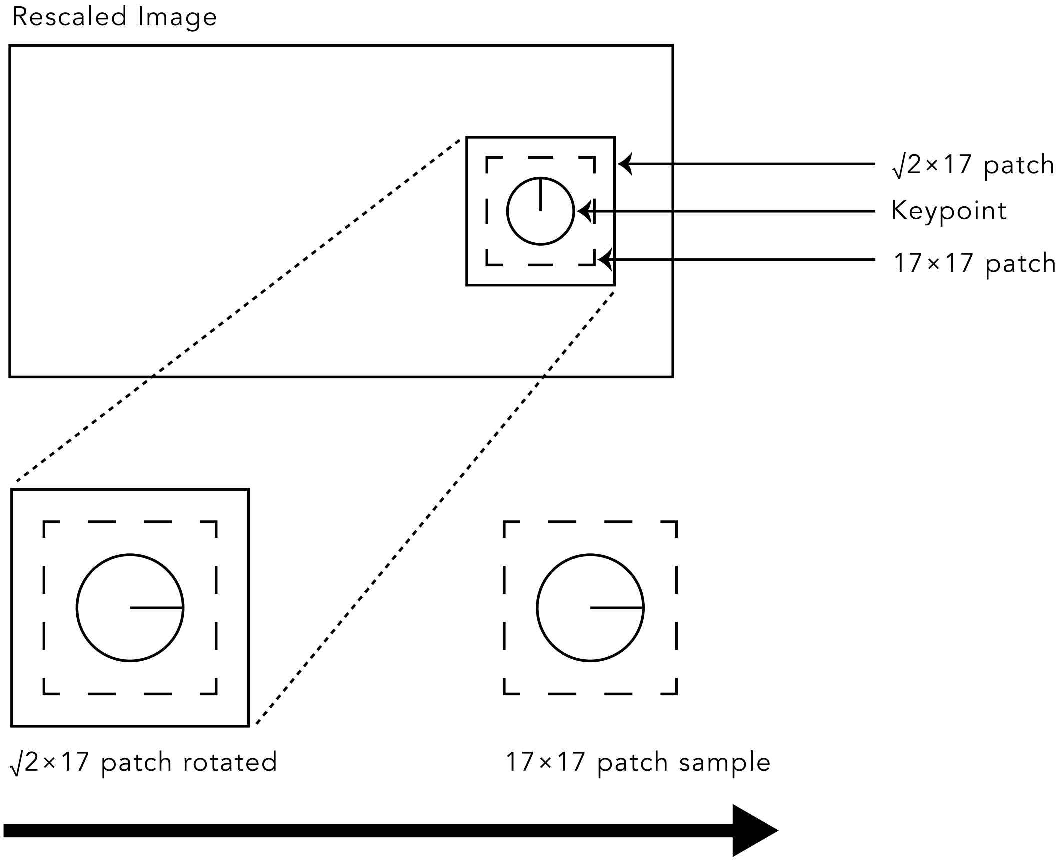


Figure 5.1: The ICFSIFT Patch Extraction Process

5.2.2 Learning the ICFSIFT filters

For each image collection we select a set of training images T. For each $I_t \in T$ we extract SIFT keypoint patches p_{kt}, as described above, and unstack them into vectors. These vectors are collected in a data matrix D. We execute ICA on D to get n filters $f_i, 1 \leq i \leq n$. These are the Independent Component Filters (ICF) extracted from SIFT keypoint patches. We call them ICFSIFT filters.

5.2.3 Extracting ICFSIFT keypoint descriptor

For an image I we have z SIFT keypoint patches $p_k, 1 \leq k \leq z$. For a keypoint patch p_k we calculate its descriptor $v_k(i) = max(C_{p_k,f_i})$ for every filter $f_i, 1 \leq i \leq n$ resulting in a n dimensional descriptor (here C_{p_k,f_i} is the normalised cross-correlation matrix of p_k and f_i as defined in equation 4.3).

Image descriptor and distance measurement

Unlike the features presented in Chapter 3 and 4, SIFT and ICFSIFT do not provide a single descriptor for an image, but rather provide multiple local descriptors per image, each one around an interest point. An obvious decision would be to avoid any extra processing and use a pairwise distance measure calculated over the local descriptors. This approach is complicated by the fact that each image can have a different number of keypoint descriptors. However we wanted to avoid using a specialised distance measure for these local features. The goal was to vary a minimum number of elements in the CBIRS process chain to make inferences about the features themselves. So we chose a method which would allow us to create an image descriptor which can be used sensibly with the distance measures commonly employed by the texture features we are comparing against, but at the same time provide independence from spatial location of the local descriptors.

We used the bag of words method (Yang et al., 2007) to create a single image feature vector from all descriptors extracted around keypoints. For an image collection we cluster the training image local descriptors using K-means clustering. When generating the feature vector for an image, each of its local descriptors is mapped to one of the clusters. We count how many descriptors are mapped to each cluster and from that generate a histogram which encodes the frequency of cluster mappings giving us a K-dimensional image descriptor. The distance between images is calculated as the Euclidean distance between their histograms for these experiments, but such a descriptor can also be sensibly used with other commonly used distance measures. This method is similar to the method of generating textons as described in §2.4.2, which motivated its use even further.

5.3 Experiments

We extracted 117 and 101 ICFSIFT filters for the VisTex and Brodatz collections respectively, using one training image per relevance class. The filters have some different characteristics compared to the ICF extracted in the previous chapters from randomly located patches, which had no rotation or scale correction. In general, they are much more regular in orientation and some show aspects of shapes. Figures 5.2 and 5.3 shows the two types of ICA based filters.

We then used these filters to extract the ICFSIFT descriptors for each image in the two collections using the method outlined in 5.2.3. We used $50, 100$ and 500 dimensional descriptors to conduct our experiments. The same method was employed for the SIFT features.

5.4 Performance Evaluation

As before as there is some randomness in the filter extraction through ICA, and also in the clustering to the image descriptors for the ICFSIFT features. We thus extracted 10 different ICFSIFT filter sets and used them in 10 iterations of feature extraction. The results for the ICFSIFT features using $K = 500$ had the best performance among the local features and we present these results using errorbars. The mean precision of the 10 repetitions is the location of the centre of the errorbar and the standard deviation of the precision values are reflected in the length.

One query image was chosen from each relevance class, giving, 124 and 111 query images for the VisTex and Brodatz collections respectively. For the VisTex database, we have relevance classes with different numbers of relevant images. In this case, where a recall value does not exist for a particular query image, the corresponding precision value is inferred through interpolation.

As before, we present results predominantly using Precision-Recall (PR) graphs but in this chapter we also use $P(n)$ values to compare the performance of different features for certain query images.

5.5 Results

Table 5.1 describes the labels used to present the results.

Label	Description
ICF-Global	Global features extracted using all collection-specific ICF
ICF-Local	ICF-based local feature as described in §3.4.2 using an image descriptor which encodes spatially local information
ICFSIFT-{K}bins	ICFSIFT Features extracted using ICFSIFT filters. Image descriptor created using the bag of words method for a descriptor of K dimensions.
SIFT-{K}bins	SIFT features. Image descriptor created using the bag of words method for a descriptor of K dimensions.

Table 5.1: Feature set labels used to present the results.

Figures 5.4(a) and 5.4(b) shows the results of comparing ICF-Global, ICFSIFT and SIFT features. We see that the performance of the ICFSIFT features are consistently better than the other two features. Figure 5.5 shows the average precision calculated

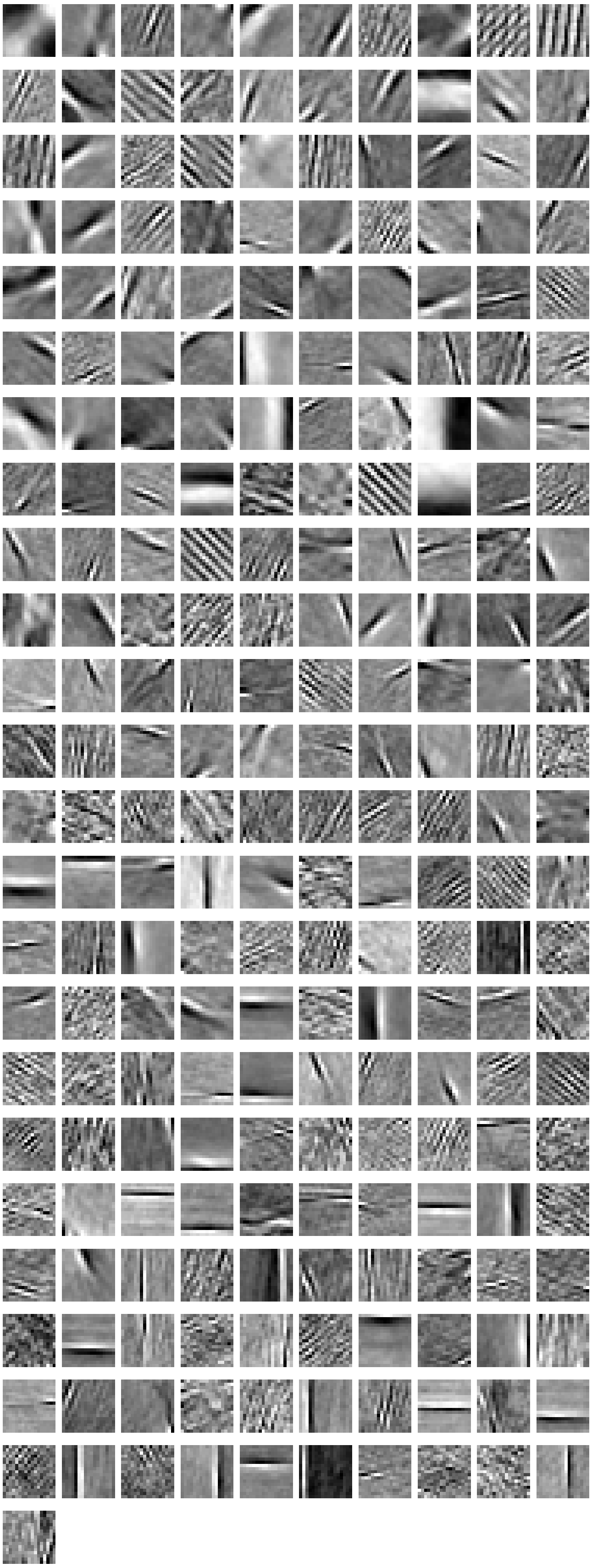


Figure 5.2: ICF extracted from the VisTex database

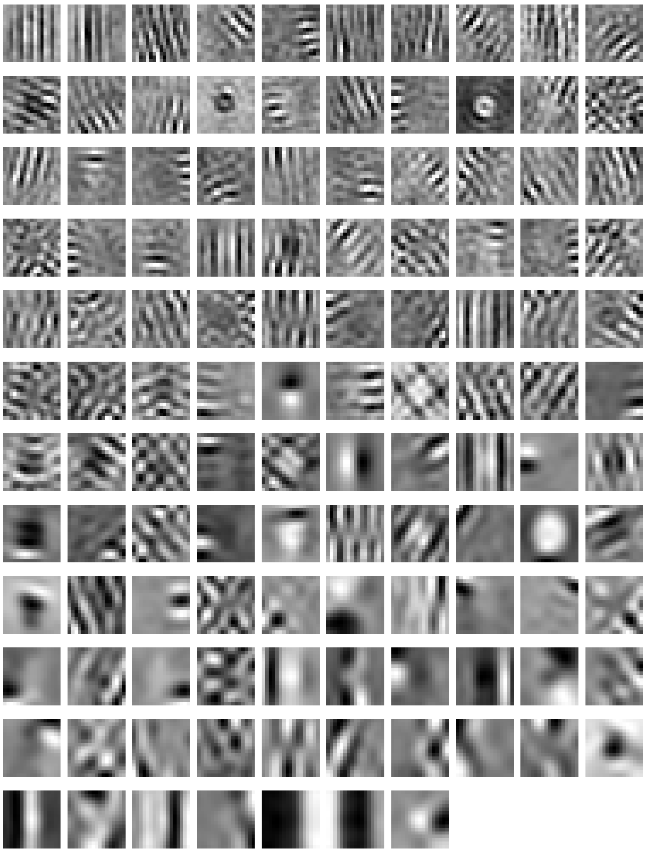
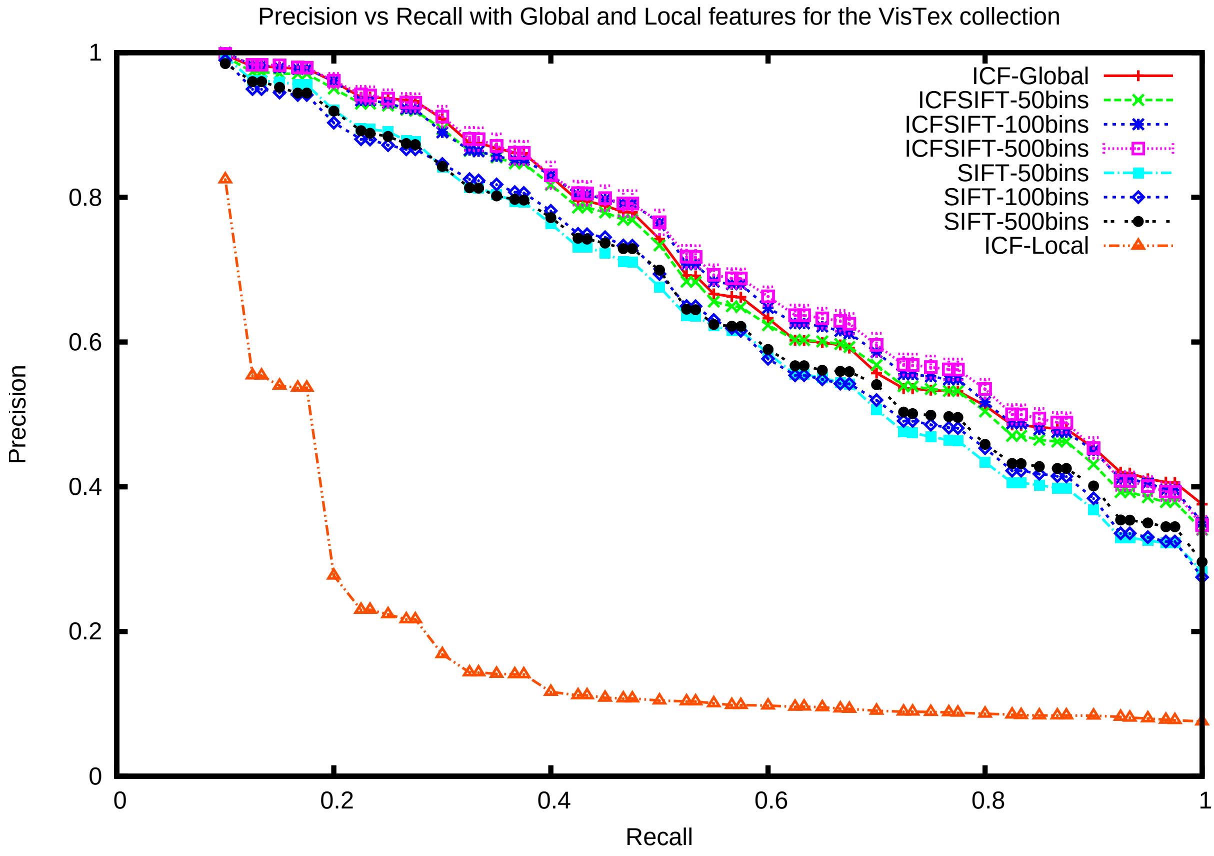


Figure 5.3: ICFSIFT filters extracted from the VisTex database

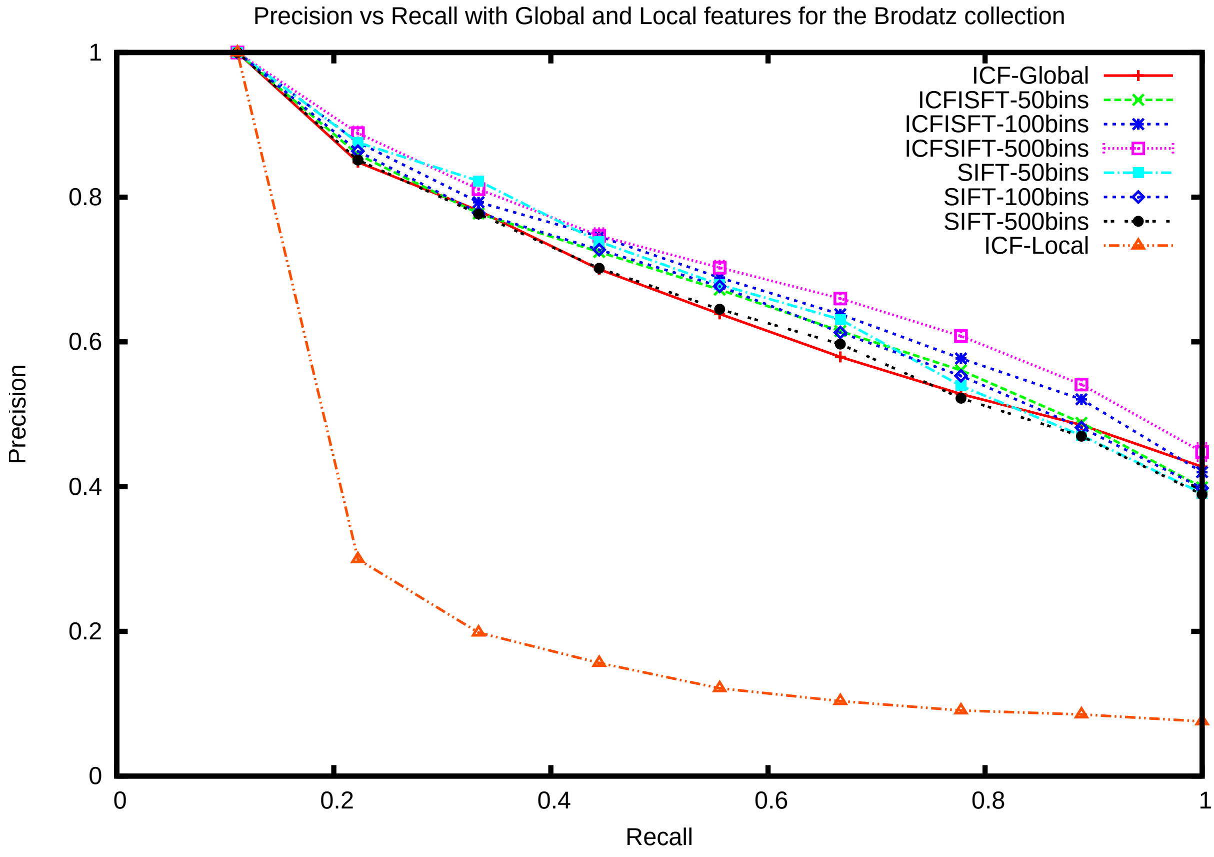
for different numbers of images retrieved for the VisTex collection. In terms of user experience, showing relevant pictures in the first few results is important. In this respect the best performing feature is ICFSIFT, closely followed by the ICF-Global feature and then the SIFT features.

The SIFT features perform better than ICF-Global for the Brodatz collection, but falls short for the VisTex collection. However the issue with the SIFT features in this scheme seems to be its lack of consistency in behaviour with varying of the descriptor size K. This is very apparent in the results of the Brodatz collection (Figure 5.4(b)) where the best performing SIFT features had $K = 50$. Increasing K actually decreased performance. For the VisTex collection SIFT features with $K = 100$ and $K = 500$ have comparable performance, however the worst is the features with $K = 50$. For the ICFSIFT features, increasing K from 50 to 500 improves performance, and after that increasing K did not yield any further benefit. However the behaviour is more consistent.

We analysed the first 10 result images returned from the queries for the VisTex collection. The VisTex collection was chosen specifically because it has a few classes where the images do not have globally consistent texture. Some of these are shown in Figure 5.6. The performance of ICF-Global for the image in Figure 5.6(a) is poor with a precision of 0.2 for the first 10 images retrieved. The global features performed better for the pumpkin query, Figure 5.6(b), with a precision of 0.8 for the first 10 images retrieved, but it does have much more regular texture compared to the image of the buildings.



(a)



(b)

Figure 5.4: Salient point-based local features for the VisTex and Brodatz database

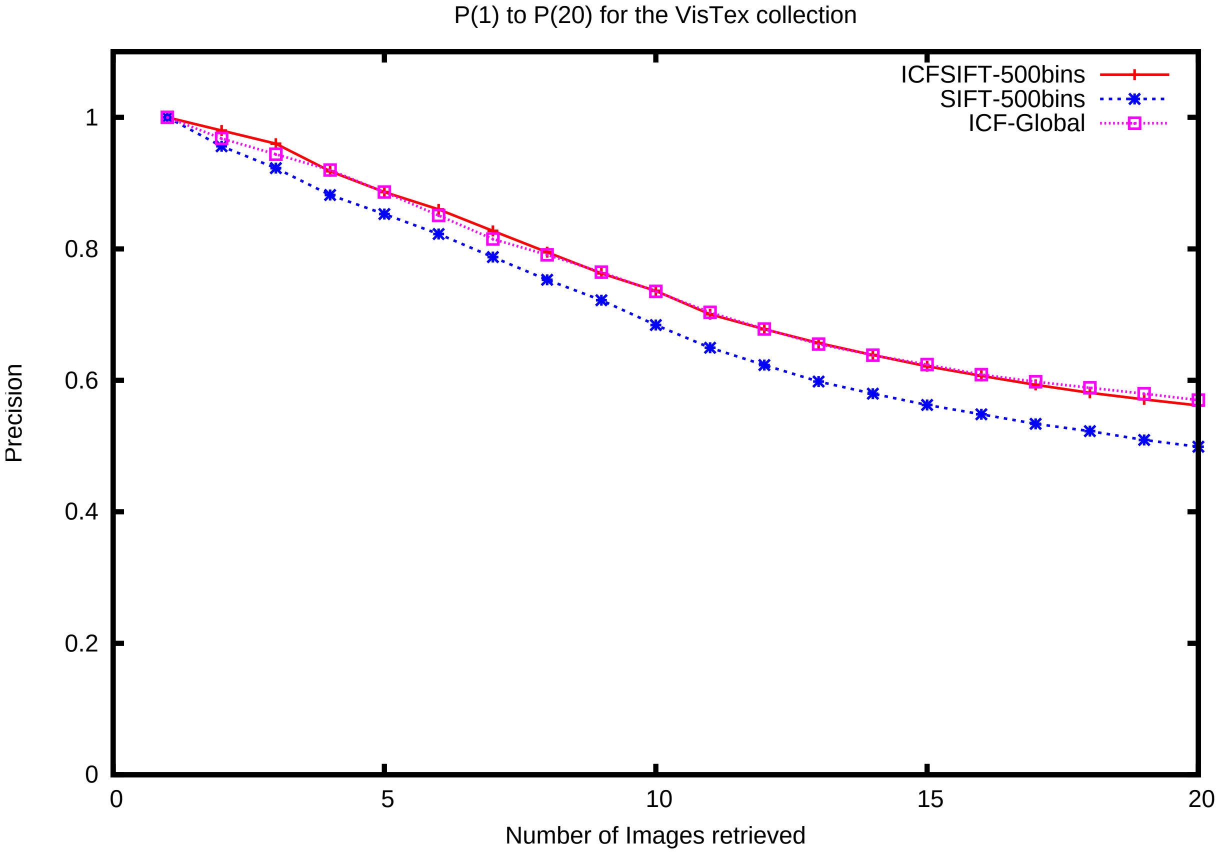
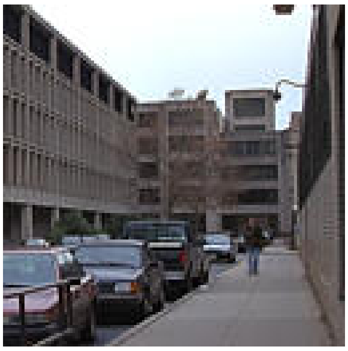
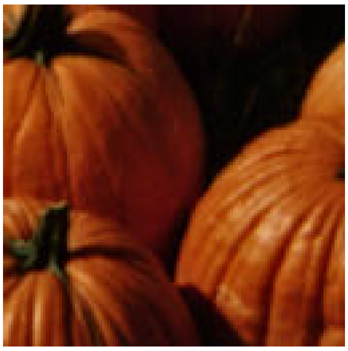


Figure 5.5: Precision vs number of images retrieved for the first 20 results on the VisTex collection.



(a) Building image



(b) Pumpkin image

Figure 5.6: Examples of VisTex images that do not have globally consistent texture

For each query image we calculated $P(10)$ for each of the three feature types. Table 5.2 summarises this data. Unsurprisingly the ICF-Global features have good $P(10)$ for a large number of images. It is interesting to see that SIFT features outperform the other two features in 21 images classes. The ICFSIFT features seem to have a better overlap with both the global features and the SIFT features. Given that it fits into the spectral class of texture features due to use of filters, and also incorporates SIFT's keypoint information, this is not surprising.

Feature	**Number of images with best $P(10)$**
Only ICF-Global	35
Only ICFSIFT-500bins	15
Only SIFT-500bins	21
ICF-Global and ICFSIFT-500bins	14
ICF-Global and SIFT-500bins	5
ICFSIFT-500bins and SIFT-500bins	18
All Three	16
Total query images	124

Table 5.2: Best P(10) results for the different feature sets

Figure 5.4(a) shows that overall the SIFT features have lower performance than the ICFSIFT features. Figure 5.7 shows a bar graph which compares $P(10)$ of the different feature types for every query image for which the SIFT features outperform the other two feature types. For 5 of the 21 image classes shown in Figure 5.7, the SIFT features have at least double the precision of the ICF-Global features. The ICFSIFT feature perform much better, and for 15 of the 21 classes it has a precision that is within 70% of the SIFT features. It never drops below 57%.

Figure 5.8 shows $P(10)$ for the queries in which the ICFSIFT features perform best. For 6 out of 15 queries the SIFT features have $P(10)$ values which are half or less than the ICFSIFT features.

It is interesting to examine the image classes for which the SIFT features performed best. SIFT features are well known for object recognition and have been shown to be effective for that task (Lowe, 2004). In the VisTex collection there are 10 relevance classes which are comprised of different views of buildings, some of which are shown in Figure 5.9. We had expected the texture-based features (ICF-Global and ICFSIFT) to not perform as well as SIFT in these classes. However of the 21 classes where the SIFT features have the best performance, only 2 have to do with the buildings, and in only one of them is the performance improvement remarkable.

Figure 5.10 shows a bar graph where the performance of the different feature sets are shown for the building queries. SIFT features have clearly superior performance for one query, in all other cases the performance of the SIFT feature are comparable to the ICFSIFT features, or slightly worse.

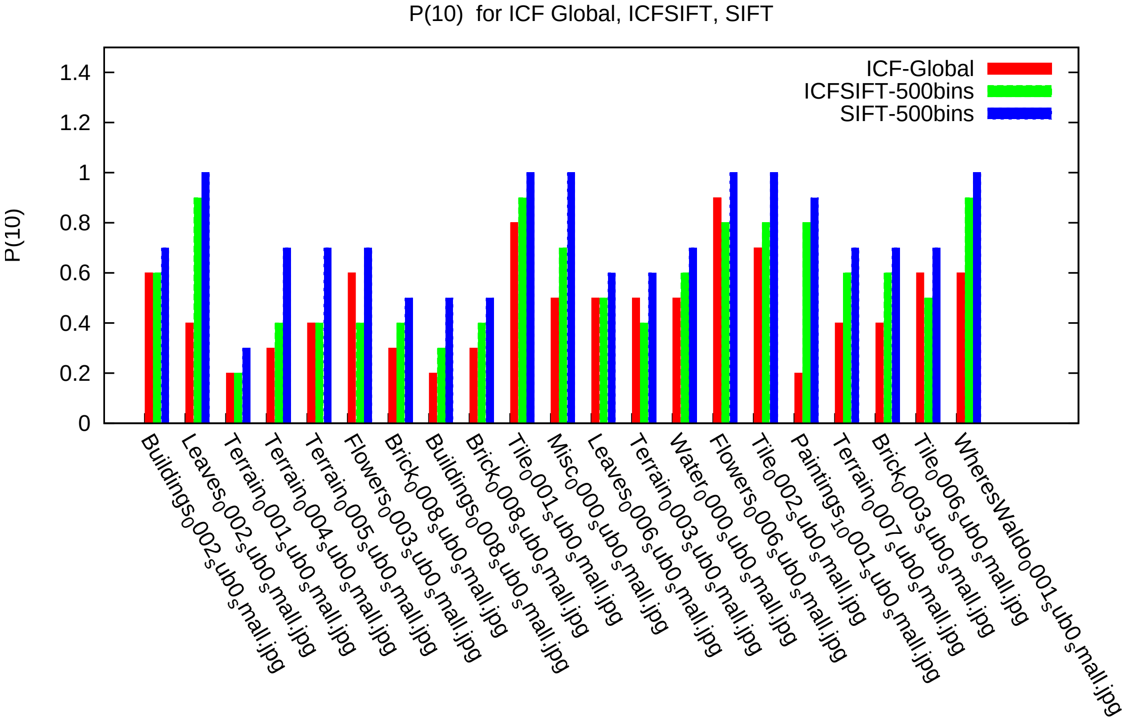


Figure 5.7: Comparing $P(10)$ for query images where the SIFT features performed best

5.5.1 Combining global and local features

In Chapter 3 we showed that that combining the global and grid-based local features decreased the performance of the ICF-based features compared to the performance of global features alone. The grid-based local features employed had a much lower performance compared to the global features. In this chapter, the proposed ICFSIFT features actually have improved performance so there is an incentive to test the performance of a combination of global and local features. To test this, we created a combined feature vector which was a concatenation of the global and local feature vectors extracted using the method outlined before. We then used this combined feature vector as the image feature and performed CBIR with the relevance classes used for the results shown above.

Figure 5.11(a) and 5.11(b) show the results of these experiments for the VisTex and the Brodatz database respectively. It can be seen that combining the global features with the grid-based local features decreases performance very slightly for the VisTex collection and quite markedly for the Brodatz collection. However, for both collections, combining the global features with the ICFSIFT features improves performance. This leads us to conclude that not only do the ICFSIFT features work well as a local feature for this class of images, but they also provide different information that can be usefully combined with the global ICF features.

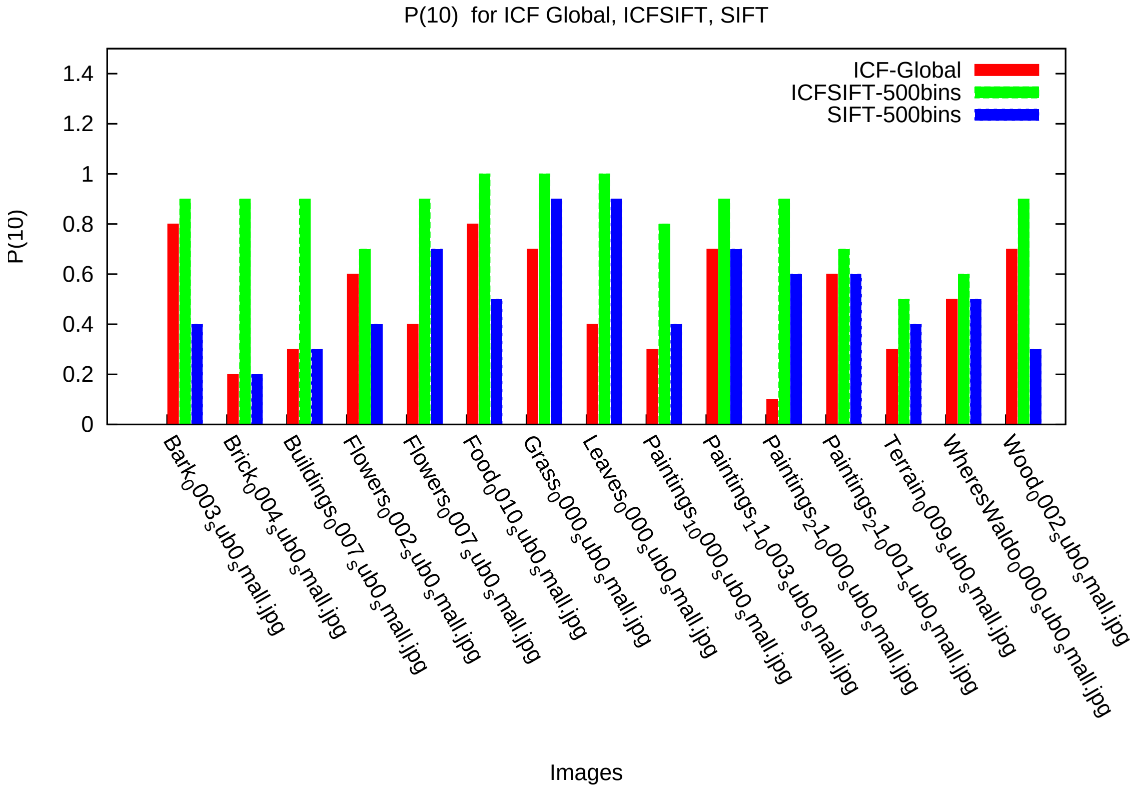
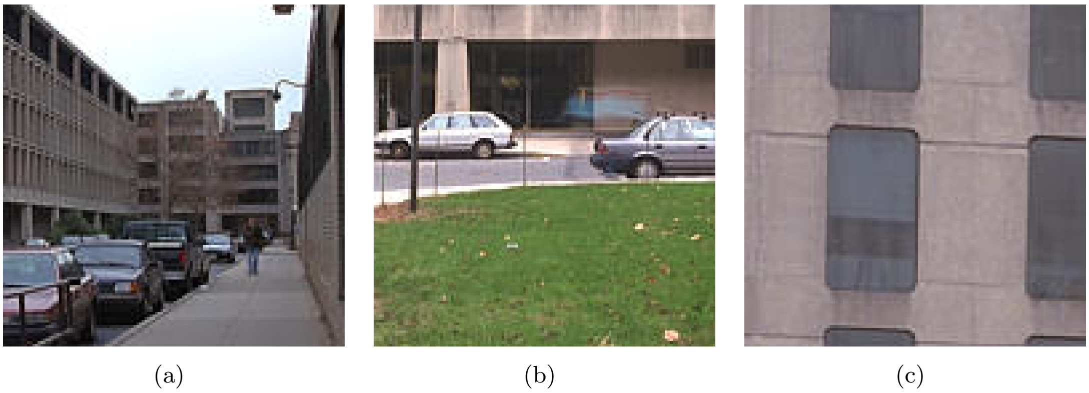


Figure 5.8: Comparing P(10) for query images where the ICFSIFT features performed best



(a) (b) (c)

Figure 5.9: 3 of the 10 query images which are different views of buildings.

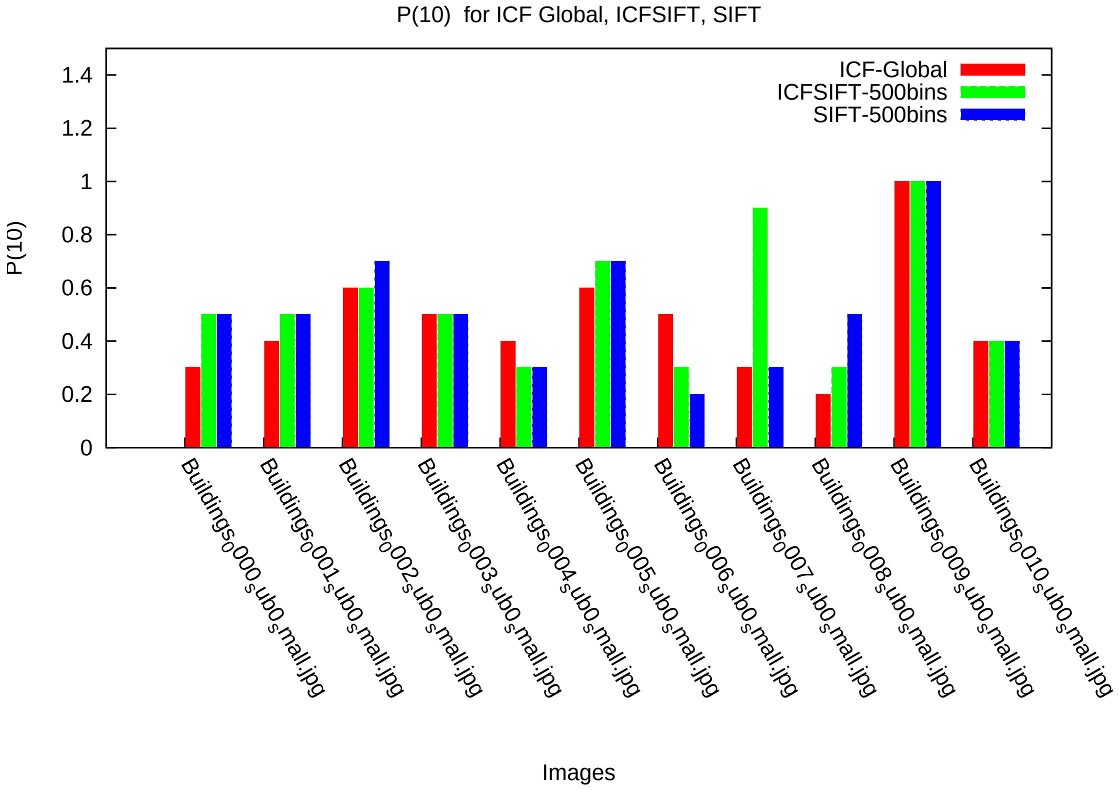


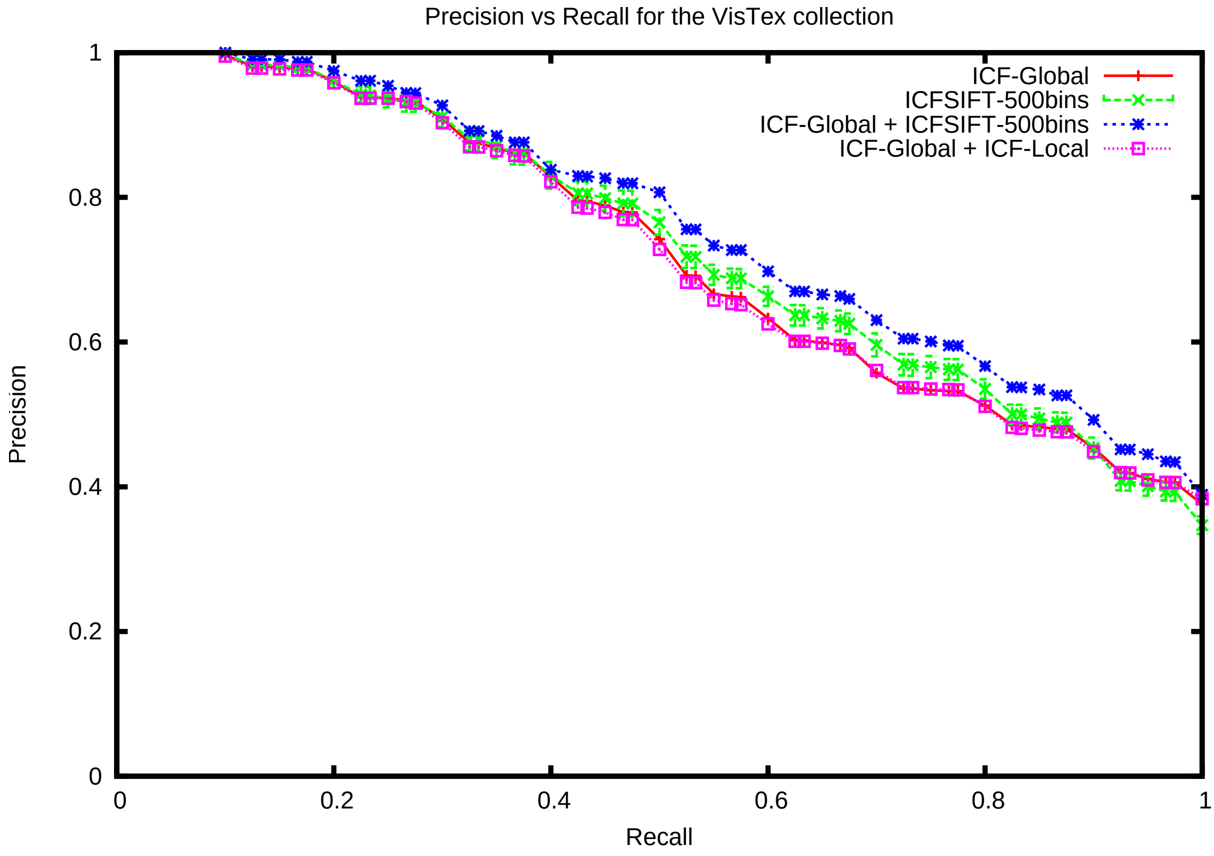
Figure 5.10: Comparing P(10) for query images of different views of buildings

Combining features using separate normalisation

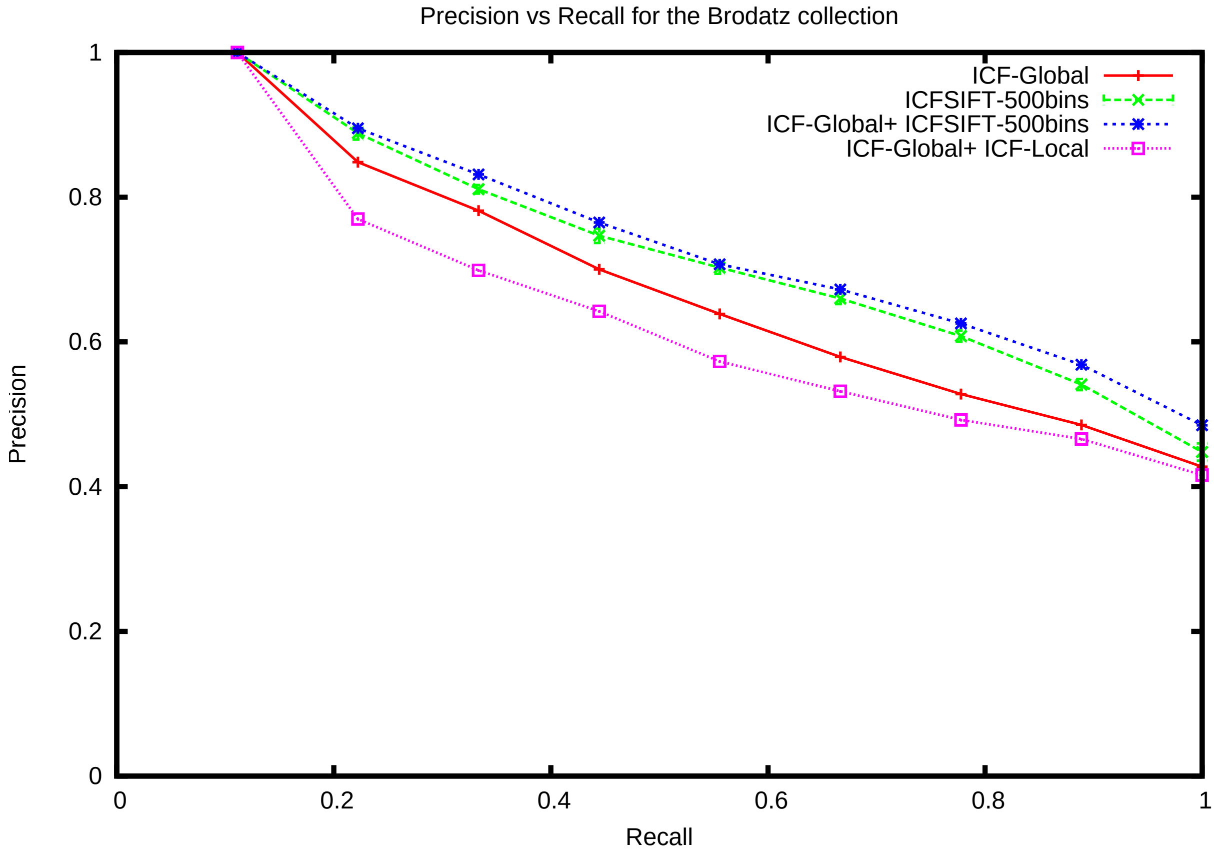
So far we have combined the global and local features naively by concatenating the feature vectors obtained from each individual feature into one feature vector. The ICF-Global and ICFSIFT features vectors were 200+ and 500 dimensions respectively, giving a feature vector over 700 dimensions. We use Euclidean distance to calculate distance measure. In this scheme it is clear that the ICFSIFT feature entries will have a greater influence on the distance calculation. To implement a more equal system we take ideas from the GIFT system which uses separate feature normalisation as a default setting. We implemented the normalisation scheme shown in Equation 5.1, where the distance between images I_p and I_q are calculated using k features $V_{p,1}..V_{p,k}$ and $V_{q,1}..V_{q,k}$. The distance is calculated as the sum of the distances of the individual normalised features. We have used Euclidean distance here, but this can be replaced with other distance measures like L_1 or χ^2.

$$distance(I_p, I_q) = \sum_{j=1}^{k} d\left(\frac{V_{p,j}}{sum(V_{p,k})}, \frac{V_{q,k}}{sum(V_{q,k})}\right) \tag{5.1}$$

Figure 5.12 shows that implementing separate normalisation has improved the performance of the combined features even further, signifying the utility of separate normalisation.



(a) VisTex

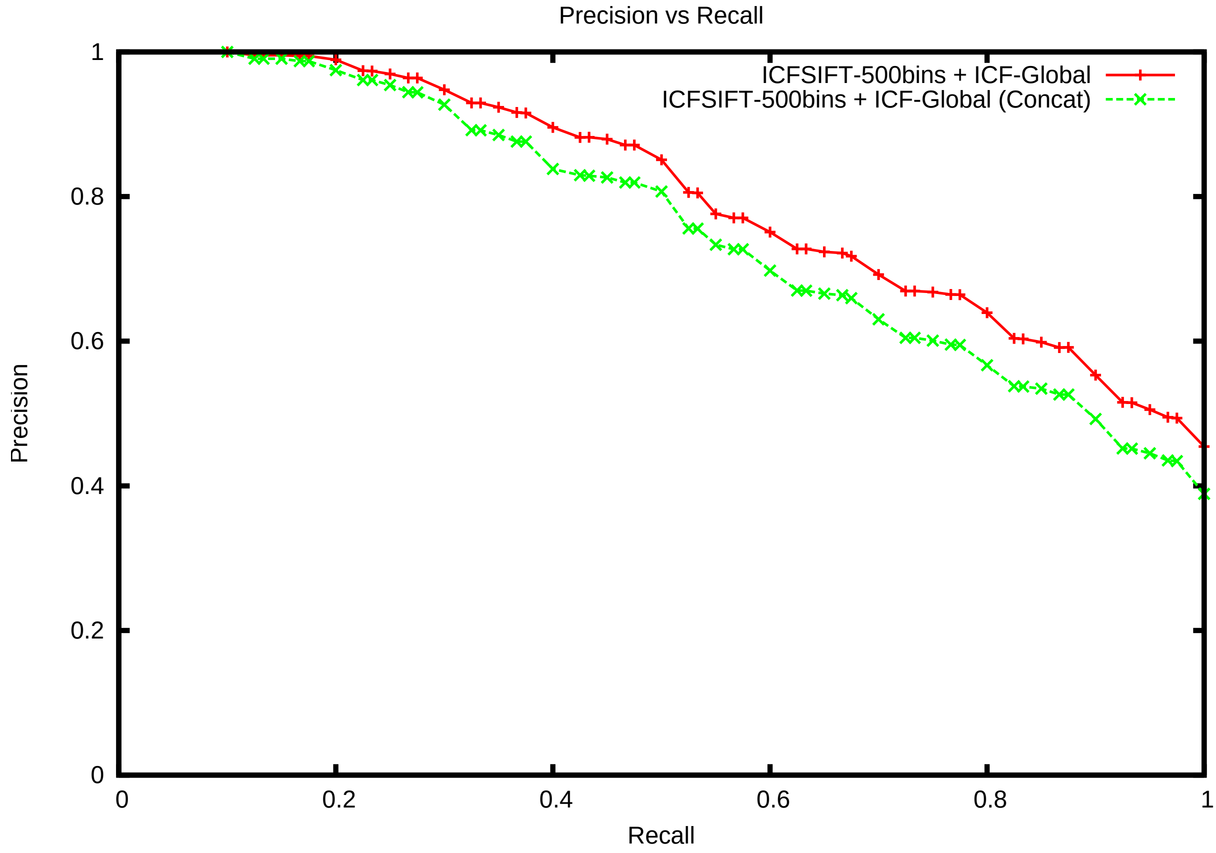


(b) Brodatz

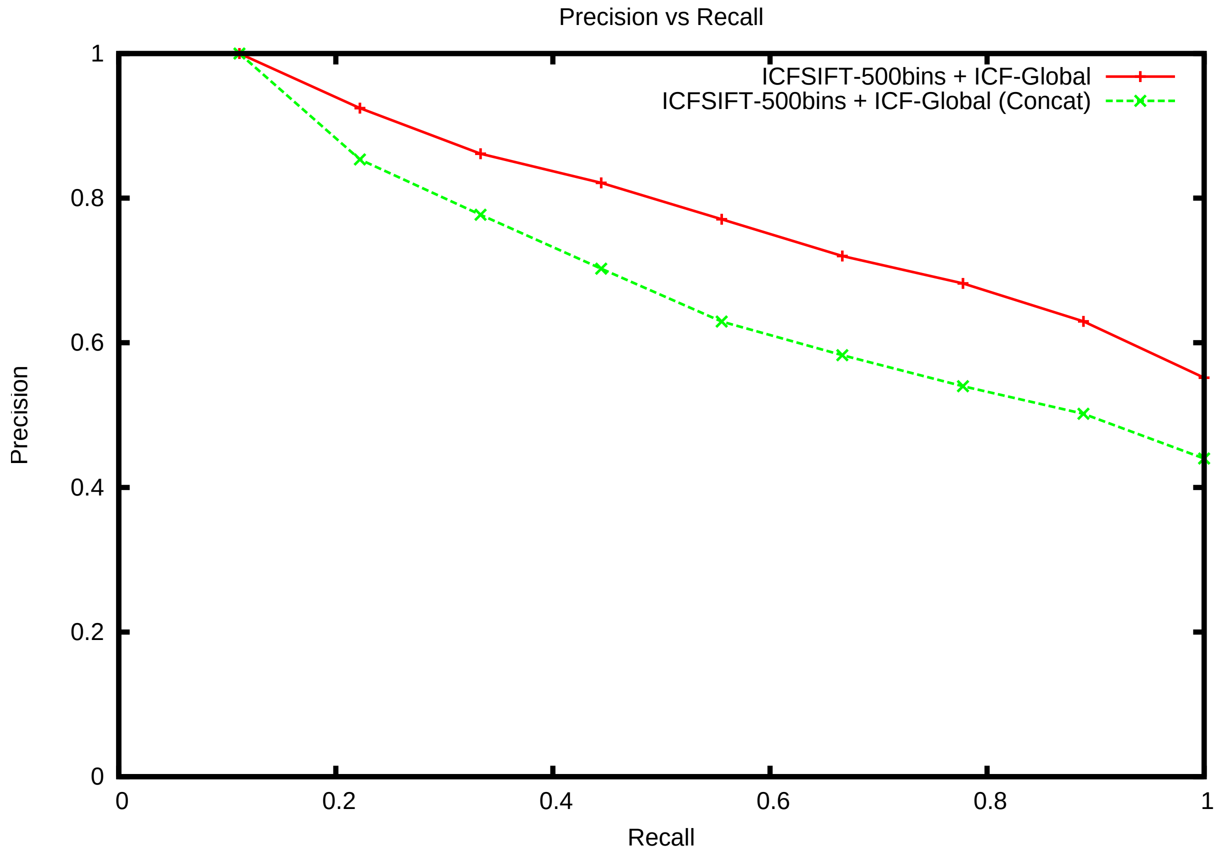
Figure 5.11: Comparing performance of a combined Global and Local features for the VisTex and Brodatz collections

5.6 Conclusion

In this chapter we have presented a new ICA-based adaptive local feature, ICFSIFT, which utilises SIFT keypoints. The new ICFSIFT feature combines keypoint detection, scale and orientation invariance of SIFT with the collection-specific adaptive properties of ICF features. We have tested this feature on the Brodatz and VisTex collections, comparing its performance against SIFT features and previously presented ICF-Global features. On both collections the ICFSIFT features performed best. We have also shown that combining these ICFSIFT features with the ICA-Global features improves CBIR performance for texture images, which is a marked improvement over the previous attempt using fixed grid-based local features. We have also shown the utility of performing separate normalisation when combining features, especially when they differ in size markedly.



(a) VisTex



(b) Brodatz

Figure 5.12: Results showing the performance of ICF-Global and ICFSIFT features combined using separate normalisation.

Chapter 6

Was it necessary to use keypoints in ICFSIFT?

In chapter 3 we presented results of local features which are extracted using a fixed grid-based method. The details of how such local features are extracted is given in §3.4.2. We concluded that the reason that these features did not perform well was that they encode spatially local information by assigning each grid block to a fixed place in the feature vector. In an effort to get around this spatial location dependence, in Chapter 5 we presented the ICFSIFT feature. It identified SIFT-like keypoints and extracted local features from a region around each keypoint. It then used the BOW method to create an image descriptor which does not encode the spatial location of the local feature vectors. The product of using the BOW method is a feature vector which is independent of the spatial locations of the keypoints.

The question we want to answer in this chapter is:

> Was the improved performance of ICFSIFT a by-product of using BOW, or does the keypoint-based processing and, scale and rotation invariance, of ICFSIFT give some extra benefit?

In other words, what makes us sure that using the keypoint-based processing gave us any advantage? It is entirely possible that the improved performance was due to the removal of spatial location dependence in the feature vector.

We explore this question by creating a local feature which subdivides an image into fixed sub-regions similar to the GIFT-like local features presented in chapter 3. However, instead of simple concatenation, this feature uses the BOW method to create the image descriptor. We call this the ICF-BOW feature and we show that it outperforms ICFSIFT. However we also demonstrate combining ICF-BOW and ICFSIFT features leads to significant improvement, leading us to believe that they both extract different but useful information. Thus we can reasonably conclude that while some of the performance improvements of the ICFSIFT features reported in Chapter 5 is due to the removal of fixed spatial dependencies, creating a keypoint-based descriptor gives us different but useful features.

6.1 Filter-based "Bag of words" features extracted from image regions

In this section we present two features which combine the 16 × 16 sub-division of images, and the BOW method, to give spatial location independent local features extracted from image grids instead of keypoints. We chose 16 × 16 because the grid-based local features presented in chapter 3, which were based on the GIFT texture features, use this subdivision, and uses the average response energy of each block as an entry into a histogram.

For each filter f_j, the LNC response energy for image I_i is calculated at each pixel. The energy image is then divided in a 16 × 16 grid resulting in 256 blocks. For image I_i, filter f_j and block b_k we extract the following information for comparison with each other:

- The average energy of the block AE_{ijk}.
- The maximum energy of the block MX_{ijk}.

Local feature vectors are created for each of the two feature types filter-wise for each block:

$$v_{ik}^{AE}(j) = AE_{ijk} \tag{6.1}$$

$$v_{ik}^{MX}(j) = MX_{ijk} \tag{6.2}$$

For each image we have 256 local feature vectors, each of N dimensions, where N is the total number of filters. We follow the BOW method described in §5.2.3 to create the image descriptor V. The results presented in this chapter uses $K = 500$, where K is the number of clusters used to find the visual words, which gives descriptors of 500 dimensions.

Apart from collection specific ICF, we used a bank of 24 Gabor filters at 3 scales and 8 orientations, and also Leung-Malik filters described in §2.4.2. The Leung-Malik filters in particular have been used to extract textons and the BOW method used here has similarities with the texton extraction method described in §2.4.2, motivating our inclusion of these filters in this evaluation.

6.2 Results and Discussion

We present the results in two stages. Initially we compare results of the two region-based features mentioned above in §6.1. Afterwards we show results from combining different ICF-based adaptive features to demonstrate the utility of ICFSIFT. All combination was done through separate normalisation (§5.5.1) The following are the labels used for the different feature types.

Label	Description
ICF-Global	Global features extracted using all collection-specific ICF
ICF-Local	ICF-based local feature as described in §3.4.2 using an image descriptor which encodes spatially local information
ICF-BOW-MX	ICF-based local features extracted as described in §6.1. The features extracted from each block is MX_{ijk}.
ICF-BOW-AE	ICF-based local features extracted as described in §6.1. The features extracted from each block is AE_{ijk}
ICF-BOW	Same as ICF-BOW-MX
Gabors-BOW	Local features extracted as described in §6.1 using a bank of Gabor filters at 3 scales and 8 orientations. The features extracted from each block is MX_{ijk}
LeungAndMalik-BOW	Local features extracted as described in §6.1 using The Leung and Malik filters as described in §2.4.2. The features extracted from each block is MX_{ijk}
ICFSIFT	ICFSIFT Features extracted using ICFSIFT filters. Image descriptor created using the BOW method with K=500 for a descriptor of 500 dimensions.

Table 6.1: Feature set labels used to present the results.

6.2.1 Comparison of region-based Bag of Words features

Figure 6.1 compares the performance of the two ICF-based BOW features proposed here. We have labelled the V^{AE} features as "ICF-BOW-AE" and V^{MX} features as "ICF-BOW-MX". Both these features are extracted from image sub-regions, similar to the ICF-Local features presented in Chapter 3. However, unlike the ICF-Local features, these two features create image descriptors which do not encode spatially local information. From Figure 6.1 we can see that this leads to a significant improvement in performance on both the Brodatz and VisTex collections. The ICF-Local features and the ICF-BOW-AE features presented here use LNC response energy and perform response scaling at a region level. For more details on LNC and response scaling please refer to chapter 4.

The two new features, ICF-BOW-AE and ICF-BOW-MX, have comparable performance. In the Brodatz database ICF-BOW-MX has a slightly better precision at recall one. The ICF-BOW-AE features have performance similar to ICFSIFT features for the Brodatz collection and better performance for the VisTex collection. ICF-BOW-MX features outperform the ICFSIFT features on both the image collections. We use only ICF-BOW-MX in all the results from now and we will refer to it simply as ICF-BOW.

To emphasise the effectiveness of using adaptive ICF, we show the results of BOW features extracted using a bank of 24 Gabor filters at 3 scales and 8 orientations (Gabor-BOW) and the Leung and Malik filters described in §2.4.2 (LMFilters-BOW).

These results are shown in Figure 6.2. Quite clearly the collection-adapted ICF-BOW features do significantly better than the BOW features using fixed pre-defined filter sets.

A comparison with the results presented in chapters 3 and 4 show that the ICF-BOW features outperform both ICF-Global and ICF-Local. This can be unintuitive, so we have explained the reason further in Appendix F.

6.2.2 Utility of the ICFSIFT feature

The fact that ICF-BOW features outperform the ICFSIFT features brings into question whether the latter feature was necessary at all. From the results it is clear that the spatial location independence obtained by BOW gives improved performance over fixed grid-based methods, even without identifying keypoints. To understand whether using the keypoint-based ICFSIFT gives any extra benefit, we adopt the same strategy that we used in chapter 5. We first combine ICF and ICF-BOW features. We then compare its results with a further combination with ICFSIFT features. If adding the ICFSIFT features to a combination of ICF-BOW and ICF features improves the retrieval performance than we can conclude that the ICFSIFT features gain some benefit from its keypoint-based processing.

Figure 6.2 shows the results of combining the ICF-based features in different combinations. In all cases we use separate normalisation to combine features (§5.5.1). The ICF-Global features are the features extracted using LNC and response scaling as described in chapter 4.

Combining ICF and ICF-BOW features gives a slight improvement over the ICF-BOW features for both image collections. As we have shown in chapter 5, combining the ICF and ICFSIFT features also gives an improved performance. However the most significant improvement comes when all three are used together. In particular we would like to draw attention to the difference in performance between using ICF-Global and ICF-BOW (the magenta line) and using all three features together (the red line with errorbars[1]). The increase in performance between them can be attributed to the inclusion of the ICFSIFT features.

Figure 6.4 and 6.5 shows some examples where adding the ICFSIFT feature with ICF+ICF-BOW improved the retrieval performance within the first 10 retrieved results. For each group of three images, the leftmost image is the query image. The middle image is one which was retrieved by both ICF+ICF-BOW and ICF+ICF-BOW+ICFSIFT. The rightmost image is one which was only retrieved after adding ICFSIFT with ICF+ICF-BOW. We can see from Figure 6.4(a) that the rotation invariance introduced by ICFSIFT helps to identify a relevant image, that was otherwise missed by ICF+ICF-BOW. In the case of Figure 6.4(b) and 6.5(b) the scale invariance of ICFSIFT helps to identify an otherwise missed relevant image. Figure 6.5(a) shows an example where ICFSIFT helped identify a relevant image which has some significant distortion when compared with the query image.

This allows us to conclude that ICFSIFT identifies information that is different and useful compared with ICF-BOW. As both features use the BOW method, we infer that the keypoint-based processing outlined in chapter 5 is useful and helps to find image content which the other two adaptive features could not identify.

The combination of ICFSIFT and ICF-BOW performs comparably with the use of all three features, with an improvement observed when ICF-Global features are included as

[1] The experiments using all three ICF-based features were repeated 10 times with 10 different sets of ICF and ICFSIFT filters, for each image collection. The mean precision of the 10 repetitions is the location of the centre of the errorbar and the standard deviation of the precision values are reflected in the length.

well. This again emphasizes the utility of the ICFSIFT features, as it is possible to get large improvements just by combining it with ICF-BOW.

6.3 Conclusion

In this chapter we explored the utility of ICFSIFT even further. We posed the question: Is the improved performance of ICFSIFT features a by-product of using BOW only, or does the keypoint-based processing give some extra benefit?

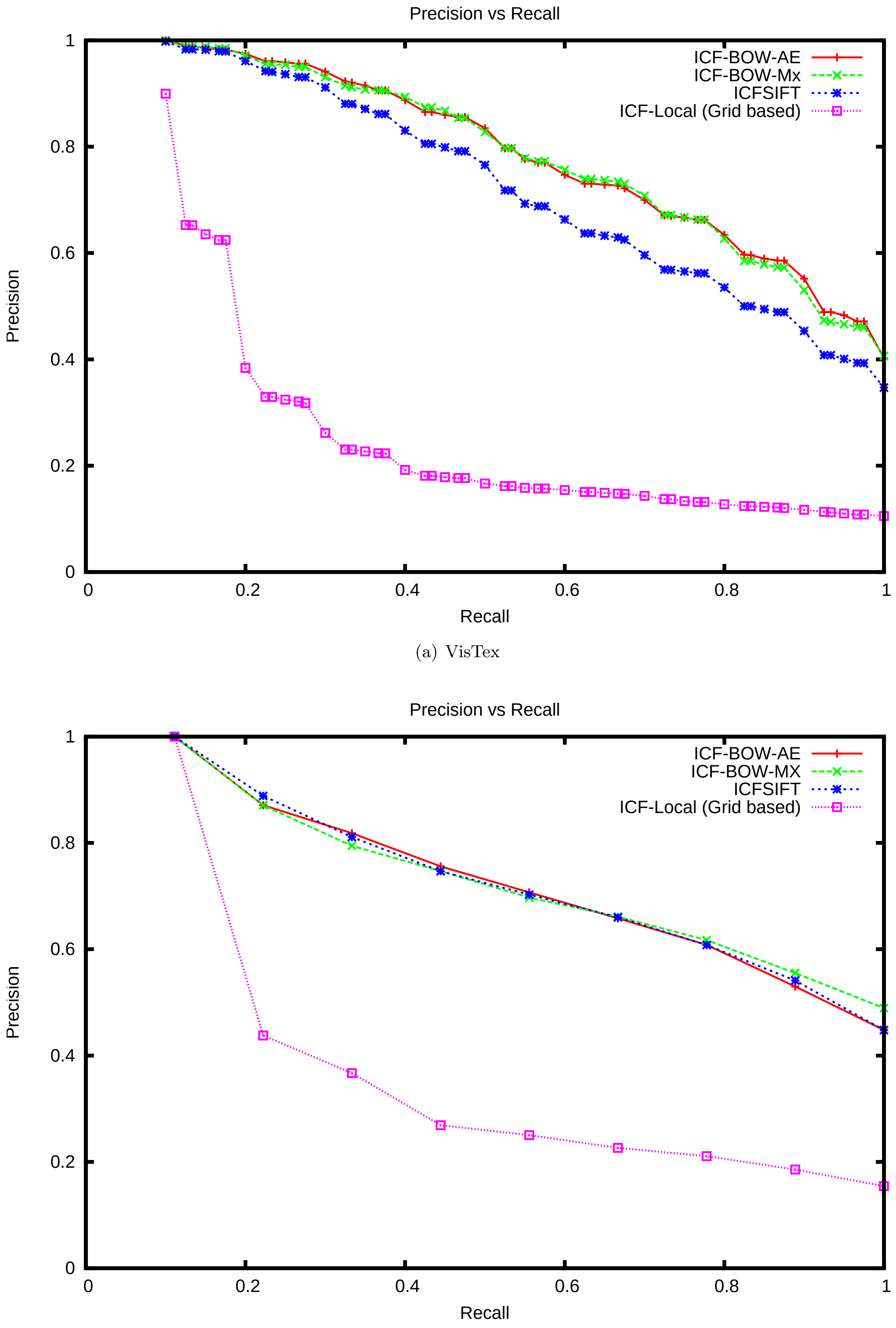
We attempted to answer this question by creating a feature that used fixed subdivisions of images, like GIFT's Gabor-based texture features, but instead of concatenating the features vectors from each image region, we used the BOW method to create a spatially local information independent image descriptor.

We have shown that features extracted using collection-specific ICF filters outperform features extracted by pre-defined banks of filters, re-enforcing our point about using collection-adapted texture features.

In our results the ICF-BOW features outperform the ICFSIFT features. However we have highlighted the utility of keypoint-based processing of ICFSIFT by demonstrating that ICFSIFT extracts useful and different features compared to ICF-BOW, as combining ICFSIFT features with a combination of ICF-BOW and ICF features gives the greatest improvement in performance. We have shown examples where the scale and rotation invariance of ICFSIFT features resulted in retrieving relevant images which the other two adaptive features failed to retrieve within the first 10 result images. In conclusion we can make the following claims.

- The BOW method is an effective way to achieve spatial location independence in an image descriptor. However in the comparison between SIFT and ICFSIFT in chapter 5 and also in the comparison between ICF-BOW, Gabor-BOW and LMFilters-BOW we see that using collection-adapted methods can give superior performance over using pre-defined/fixed methods.
- ICFSIFT features through their keypoint identification, and scale and rotation invariance, can provide different but useful features compared to ICF-BOW and ICF-Global, thus justifying the use of ICFSIFT.

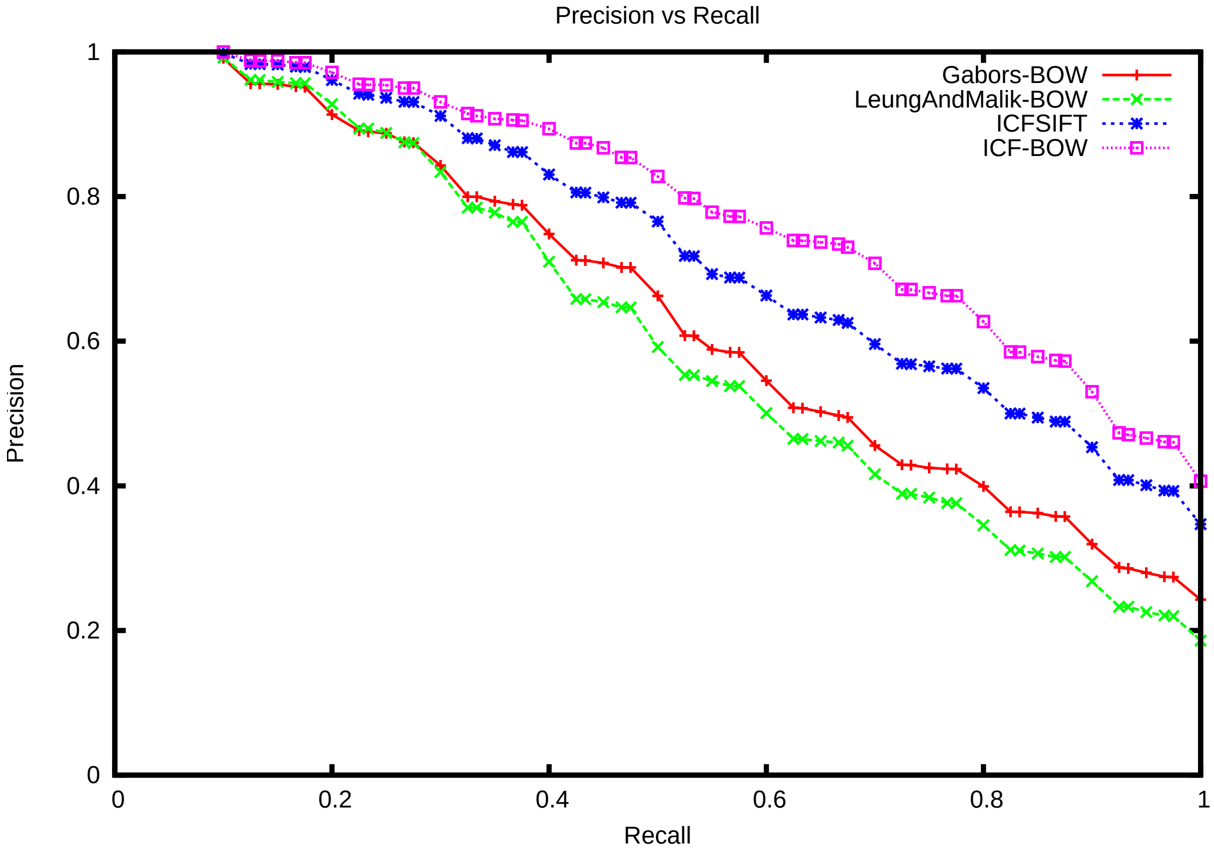
This marks the end of describing improvements to the adaptive features. In the next chapter we build on the findings of this and previous chapters, where we compare the combined performance of the three ICF-based adaptive features (ICF-Global, ICFSIFT, ICF-BOW) with the performance of LBP features which have been hand-tuned for a particular texture collection.



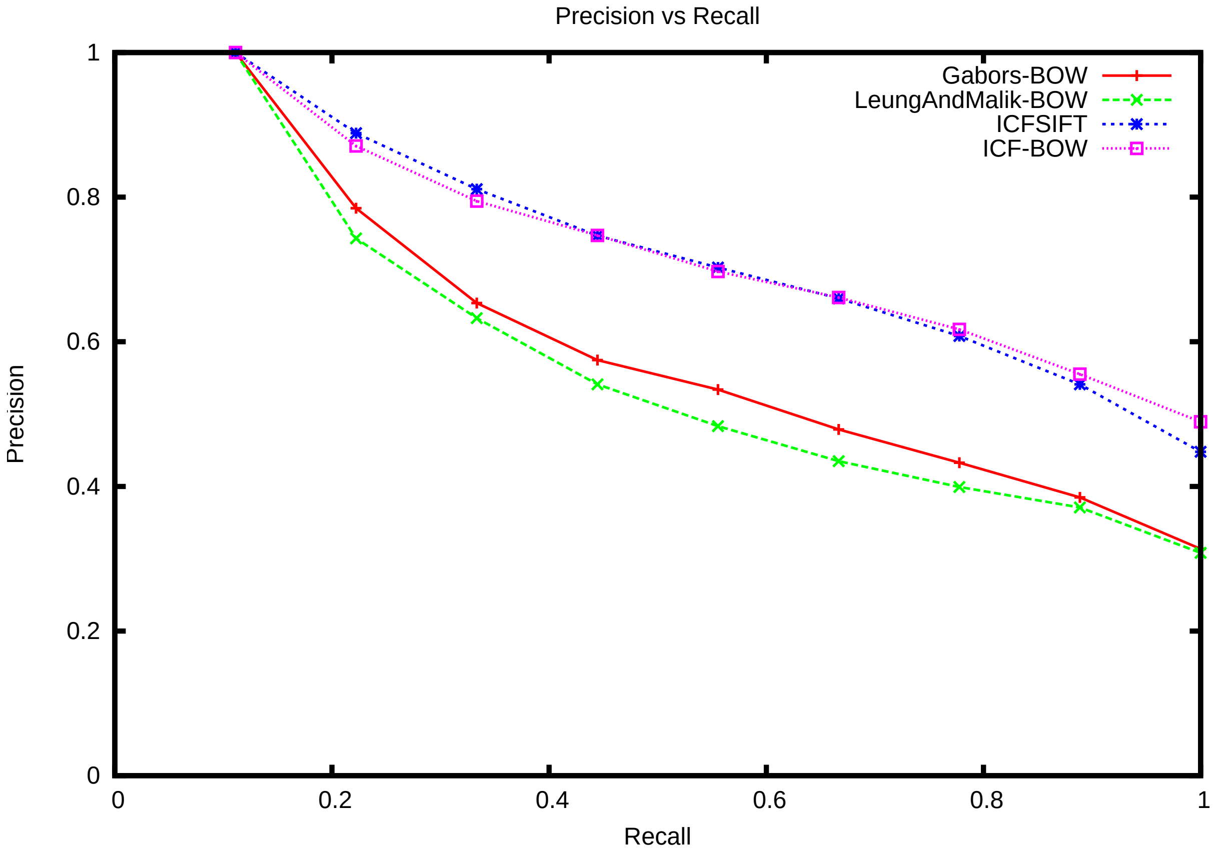
(a) VisTex

(b) Brodatz

Figure 6.1: Comparison of ICF-BOW-AE and ICF-BOW-MX features

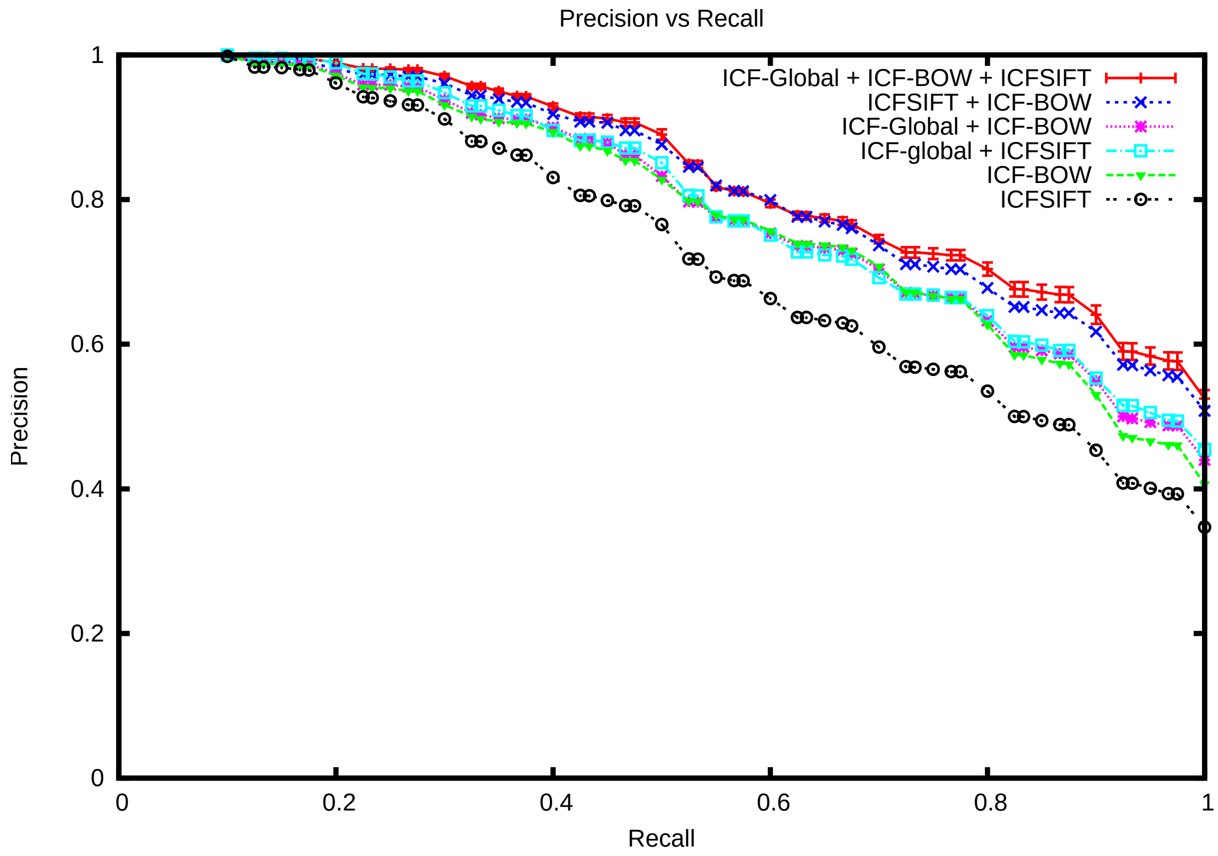


(a) VisTex

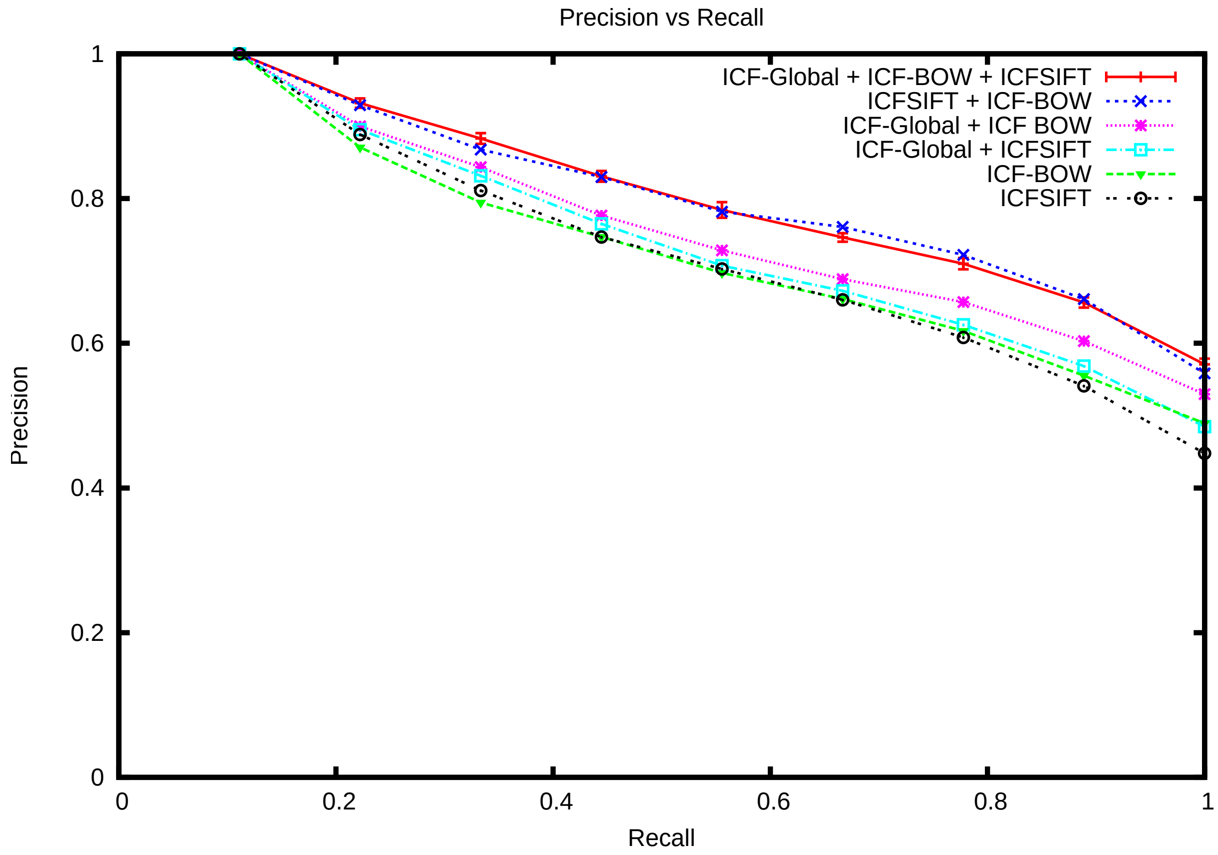


(b) Brodatz

Figure 6.2: Performance of ICF-BOW compared with other features features using the Bag of words model and LBP-Doshi.

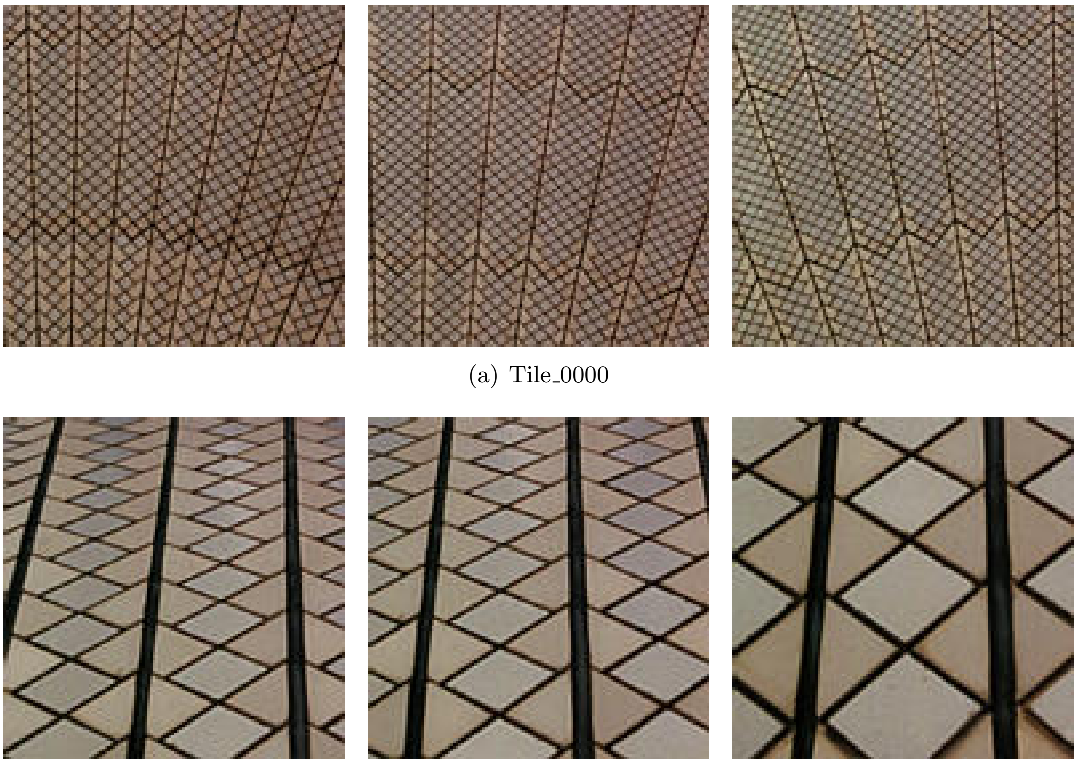


(a) VisTex



(b) Brodatz

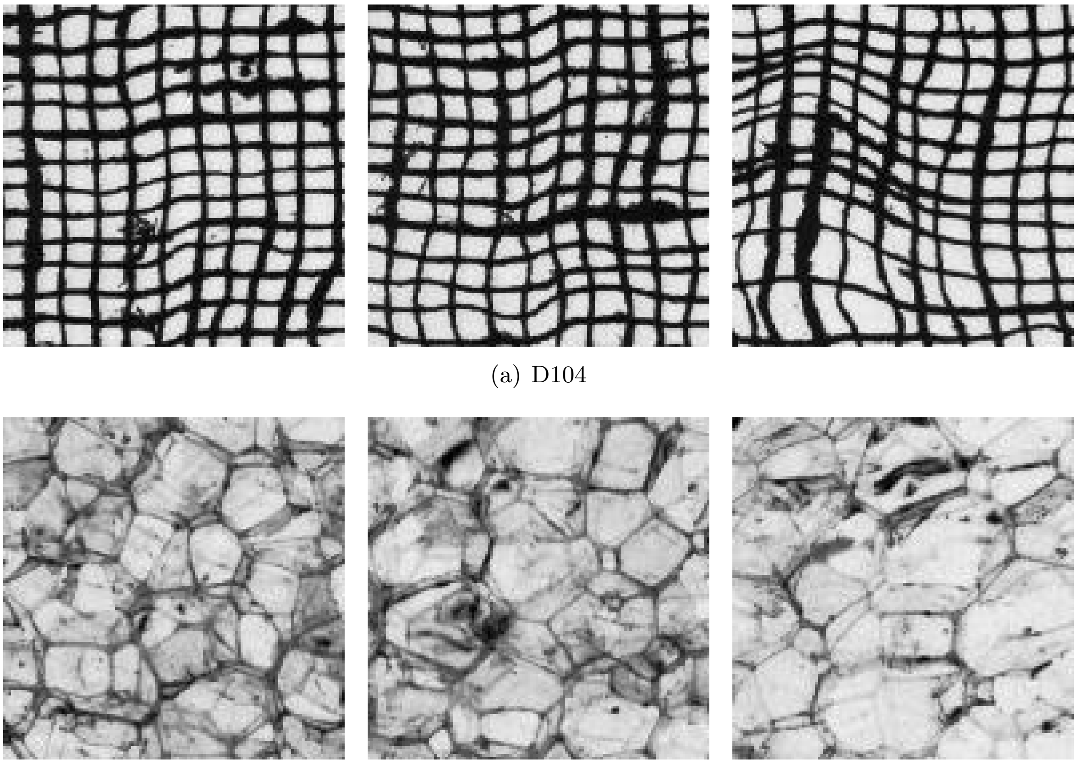
Figure 6.3: Performance of the combined adaptive features compared with LBP-Doshi



(a) Tile_0000

(b) Tile_0001

Figure 6.4: Examples relevance classes from the VisTex collection, for which adding the ICFSIFT feature improved performance. For each group of three images, the left most image is the query image. The middle image is one which was retrieved by both ICF+ICF-BOW and ICF+ICF-BOW+ICFSIFT. The right most image is one which was only retrieved by ICF+ICF-BOW+ICFSIFT.



(a) D104

(b) D112

Figure 6.5: Examples relevance classes from the Brodatz collection, for which adding the ICFSIFT feature improved performance. For each group of three images, the left most image is the query image. The middle image is one which was retrieved by both ICF+ICF-BOW and ICF+ICF-BOW+ICFSIFT. The right most image is one which was only retrieved by ICF+ICF-BOW+ICFSIFT.

Chapter 7

Comparing the LBP features

In chapter 3 we showed that collection-specific ICF-based global features perform better than global and grid-based features extracted using pre-defined fixed features. In chapter 4 we demonstrated how our proposed techniques improve the performance of ICF-based global features. In chapter 5 we presented ICFSIFT, a SIFT keypoint-based collection-specific local feature. In chapter 6 we presented ICF-BOW, a grid-based collection-specific local feature which uses the BOW method. We also showed in chapter 6 how the combination of ICF-based global features, ICFSIFT features and ICF-BOW features performs better than any of the adaptive features individually. In this chapter we will use the combined ICF-based feature in a comparison meant to mimic a plausible CBIR scenario.

When designing a generic CBIRS, the designer will usually try to select features which should work well for a broad range of image collections. We have already shown that collection-specific ICF-based features can perform better than these generically chosen features, such as the GIFT-like Gabor-based features, HOG, PHOG, SIFT, etc. In this chapter we want to compare the adaptive features with collection-specific human hand-tuned features. In particular we want to demonstrate the following:

> Features that are hand-tuned for one particular collection may not be suited for other collections, even if the collections apparently contain images with similar characteristics. As it is both difficult and time consuming to hand-tune features for every collection added to a CBIRS, automatically discovered collection-specific features can be a better option than just using the same feature for each collection.

We limit our discussion to texture features and texture collections containing images with globally consistent texture. The hand-tuned feature we used is a combination of LBP features which was found to work best for the Outex_TR_0000 image collection by Doshi and Schaefer (2012). We chose to find a hand-tuned feature which has been shown to work well on a database with images with globally consistent texture, as we were applying it to other databases with images with similar properties. We employed the same LBP combination on the Brodatz, VisTex and CureT collections. At the same time we extracted ICF-Global, ICFSIFT and ICF-BOW features from these collections. We compare the performance of the hand-tuned LBP features with performance of the combined ICF-based adaptive features here. We show that hand-tuned features can perform better than the adaptive features for its target image collection, but when presented with different collections the adaptive features perform as well as or better than the hand-tuned feature.

We briefly describe the work of Doshi and Schaefer (2012) below followed by our evaluation mechanism. Then we present individual sections for each image collection.

7.1 Doshi and Schaefer (2012)

Doshi and Schaefer (2012) tried 16 variants of LBP along with Gabor filters, Tamura features, and two different GLCM features to find the most effective texture feature for the Outex_TR_00000 image collection. In reality the numbers are even more extensive, as they tried different combinations of the LBP variants for a total of 38 LBP features. These were tested using three different distance measures: the Bhattacharya, chi-squared and L_1 distances. The non-LBP features were only tested using the chi-squared distance. There was in total 118 different attempts to find the best performing texture feature.

The Outex_TR_00000 has 319 image classes with 20 images in each class. Doshi et al. used every image as a query for the whole database. For each query the percentage of relevant images retrieved in the top 20 retrieved images is used as a metric of evaluation.

The best performing non-LBP feature was the Gabor-filter based features which had an accuracy of 44.43%. All the other non-LBP features had accuracy below 30%. The best performing texture feature for this collection was a concatenation of $LBP_{1,8}$, $LBP_{3,8}$ and $LBP_{5,8}$, which gave an accuracy of 63.74% when using the L_1 distance. The L_1 distance measure coupled with the fact that all three individual features have a 256 dimensional feature vector means that Doshi and Schaefer (2012) did not encounter the problem with concatenation that we described in §5.5.1.

It is worth noting that the best performing feature was a combination of multiple LBP features, which do not use the uniform patterns and are not rotation invariant. We compare CBIR performance of the combined adaptive features (ICF-Global+ICFSIFT+ICF-BOW) with these combined LBP features. We will refer to the combined LBP features as LBP-Doshi features from now.

7.2 Evaluation method

We performed our evaluation on the following four image collections:

- Outex_TR_0000
- Brodatz
- VisTex
- CureT

All four collections largely have images with globally consistent texture. We have already discussed the Brodatz and VisTex collections in §3.1. We will discuss the other two further when we discuss the results. Below we describe the features used to do the evaluations, followed by the metrics used.

7.2.1 Texture features and distance measure

ICF-based collection-adapted texture features

We extracted ICF and ICFSIFT filters from each collection. We then extracted ICF-Global, ICFSIFT (using 500 bins) and ICF-BOW (using 500 bins) features using the collection-specific filters. The results of the adaptive features presented are obtained by combining these three features using separate normalisation as described in §5.5.1.

LBP-Doshi features

As discussed earlier, the LBP-Doshi feature is a combination of $LBP_{1,8}$, $LBP_{3,8}$ and $LBP_{5,8}$. We extracted each LBP feature separately and concatenated them together to form a feature vector of 768 dimensions. This was done to remain faithful to the implementation of Doshi and Schaefer (2012).

Distance measure

Doshi and Schaefer (2012) found that the LBP-Doshi features performed best when used with the L_1 distance measure, as defined in equation 7.1, where V_p and V_q are two feature vectors of n dimensions, and j is the index into the vectors.

$$d_1(V_p, V_q) = \sum_{j=1}^{n} |V_p(j) - V_q(j)| \tag{7.1}$$

The results present in this chapter were obtained by using this distance measure.

7.2.2 Performance measurement

As before, for each image collection we choose one image per relevance class as the query image. We perform queries and calculate the average precision for each unique recall value. These are used to generate the PR graphs.

Doshi and Schaefer (2012) use a measure of the average percentage of relevant images retrieved in the top 20 retrieved images. In this case, all the images in the database are used as query images. This is no different from calculating average P(20) of all the query images and expressing it as a percentage, or the average retrieval rate (ARR) used by Bai, Zou, Kpalma and Ronsin (2012), Bombrun et al. (2011) and Stitou et al. (2009) to report results for a subset of the VisTex collection. The recall rate (RR) for a query image I_q calculated for the first N images retrieved is calculated as

$$RR_N(I_q) = \frac{number\ of\ relevant\ images\ retrieved\ by\ N}{N} * 100 \tag{7.2}$$

and ARR for a set of query images Q is then simply

$$ARR_N = \frac{\sum_{q \epsilon Q} RR_N(I_q)}{size(Q)} \tag{7.3}$$

We will report ARR_N results, using all the images in the database as queries, where N is the number of images per relevance class in the collection.

As before to account for the randomness in the ICF-based feature extraction process, the experiments were repeated ten times when using these features. The results are presented using errorbars in the PR graphs. The mean precision of the 10 repetitions is the location of the centre and the standard deviation of the precision values are reflected in the length. When presenting ARR_N values, we present the mean and the standard deviation. Such details are not shown for the LBP-Doshi features as there is no randomness in its feature extraction process.

7.3 Results

We present results for one database at a time.

7.3.1 Outex_TR_0000

The Outex framework (Ojala, Maenpaa, Pietikainen, Viertola, Kyllonen and Huovinen, 2002) mainly concentrates on image classification and segmentation. However it provides the Outex_TR_0000[1] collection as a benchmark for retrieval evaluation. As mentioned before this collection has 319 relevance classes, each with 20 images for a total of 6380, 128×128 images. Most of the images have globally consistent texture. Some examples are shown in Figure 7.1. The images in this collection have a wide variety of surface textures varying in illumination direction, rotation and resolution.

Figure 7.2 shows the comparative performance of the ICF-based adaptive features and the LBP-Doshi features. Quite clearly the LBP-Doshi features outperform the adaptive features. This is not surprising, as we mentioned that this combination of LBP features was found after extensive experimentation for this particular collection.

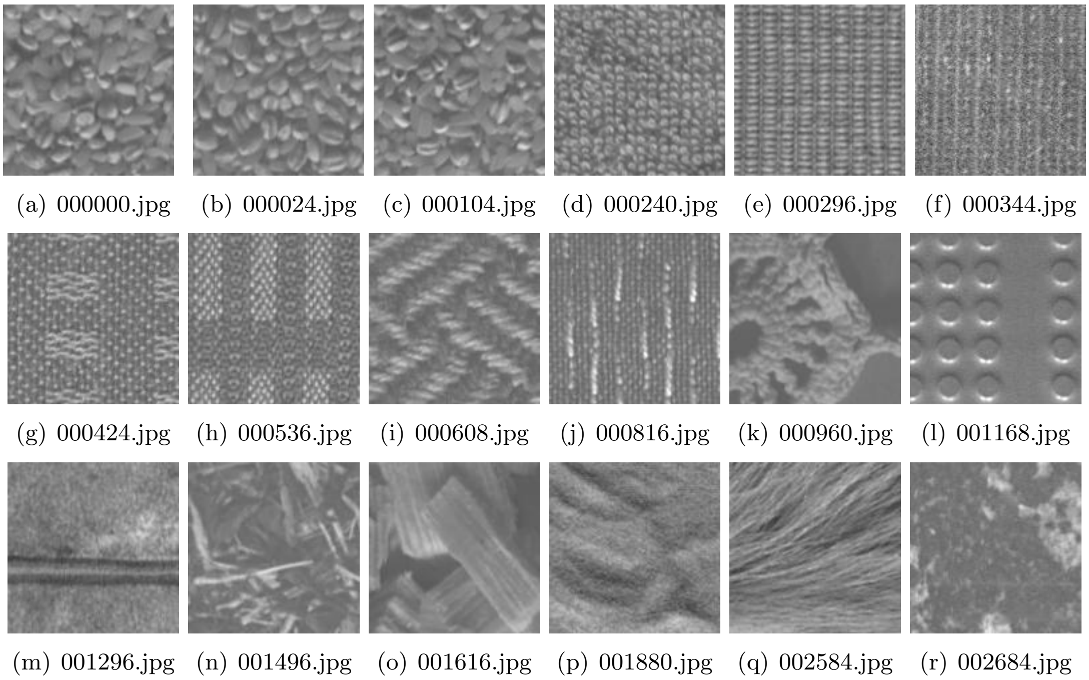
The ICF-based adaptive features achieve an average ARR_{20} of 55.13%±0.06%. This is lower than the 63.74% achieved by LBP-Doshi. However, these ICF-based results are the best non-LBP results with the next best one being the Gabor-based features at 44.43%. In fact, the ICF-based features outperform 53 other LBP-based features.

7.3.2 Brodatz

Figure 7.3 shows that the LBP-Doshi and the ICF-based adaptive features have comparable performance for the Brodatz collection.

The ICF-based adaptive features achieve an ARR_9 of 77.60% ± 0.20%. The LBP features achieve 77.67%. Both these values are lower than the 84% reported by Xu et al.

[1] http://www.outex.oulu.fi/index.php?page=retrieval



(a) 000000.jpg (b) 000024.jpg (c) 000104.jpg (d) 000240.jpg (e) 000296.jpg (f) 000344.jpg

(g) 000424.jpg (h) 000536.jpg (i) 000608.jpg (j) 000816.jpg (k) 000960.jpg (l) 001168.jpg

(m) 001296.jpg (n) 001496.jpg (o) 001616.jpg (p) 001880.jpg (q) 002584.jpg (r) 002684.jpg

Figure 7.1: Example images from the Outex_TR_0000 collection.

(2000), which used multi-resolution simultaneous autoregressive (MRSAR) model. MRSAR has been criticised for its limited ability to measure perceptual similarity (Lazebnik et al., 2005). The next best results as far as we could find is 76.26% by Lazebnik et al. (2005) which is comparable with the results of the ICF-based features and the LBP-Doshi features.

7.3.3 VisTex

Figure 7.4 shows that the ICF-based adaptive features outperform the LBP-Doshi features for the VisTex collection. The VisTex collection has a wider variety of illumination intensities and viewing angles compared to the Outex_TR_0000 and the Brodatz collections and for this collection the ICF-based adaptive features perform slightly better than the LBP-Doshi features.

The VisTex collection we used has a different number of images per relevance class. Table 7.1 reports results for different values of N. Again both feature types perform very well, with a slight lead for the adaptive features.

	ICF-Global+ICFSIFT+ICF-BOW	LBP-Doshi
ARR_{10}	$89.87\% \pm 0.20\%$	90.39%
ARR_{20}	$59.14\% \pm 0.18\%$	58.23%
ARR_{30}	$42.84\% \pm 0.16\%$	41.99%
ARR_{40}	$33.86\% \pm 0.14\%$	32.74%

Table 7.1: ARR results for the VisTex collection.

When varying N for each query image such that N is the number of relevant images in the associated relevance class, the ICF-based features achieve an ARR of $78.09\% \pm 0.23\%$

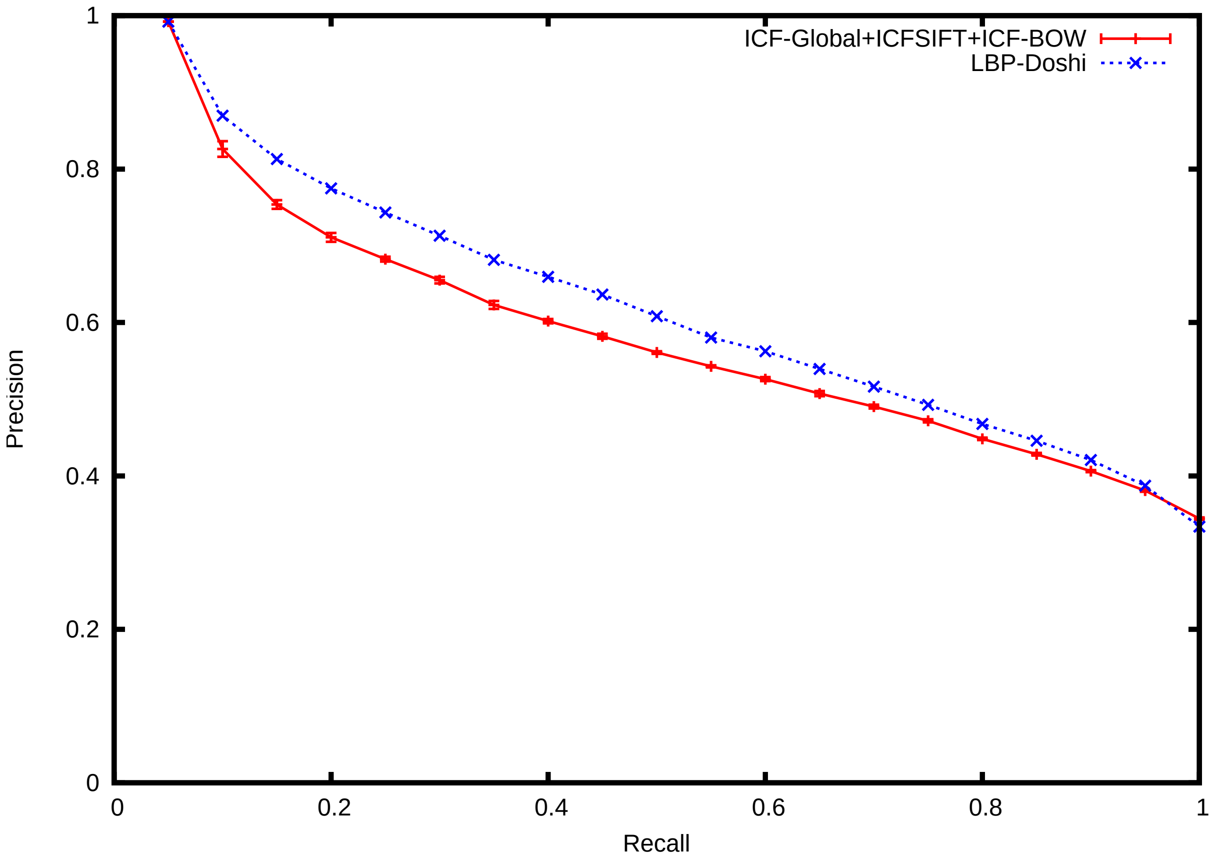


Figure 7.2: Performance of LBP-Doshi and ICF-based adaptive features on the OUTex_TR_0000 database.

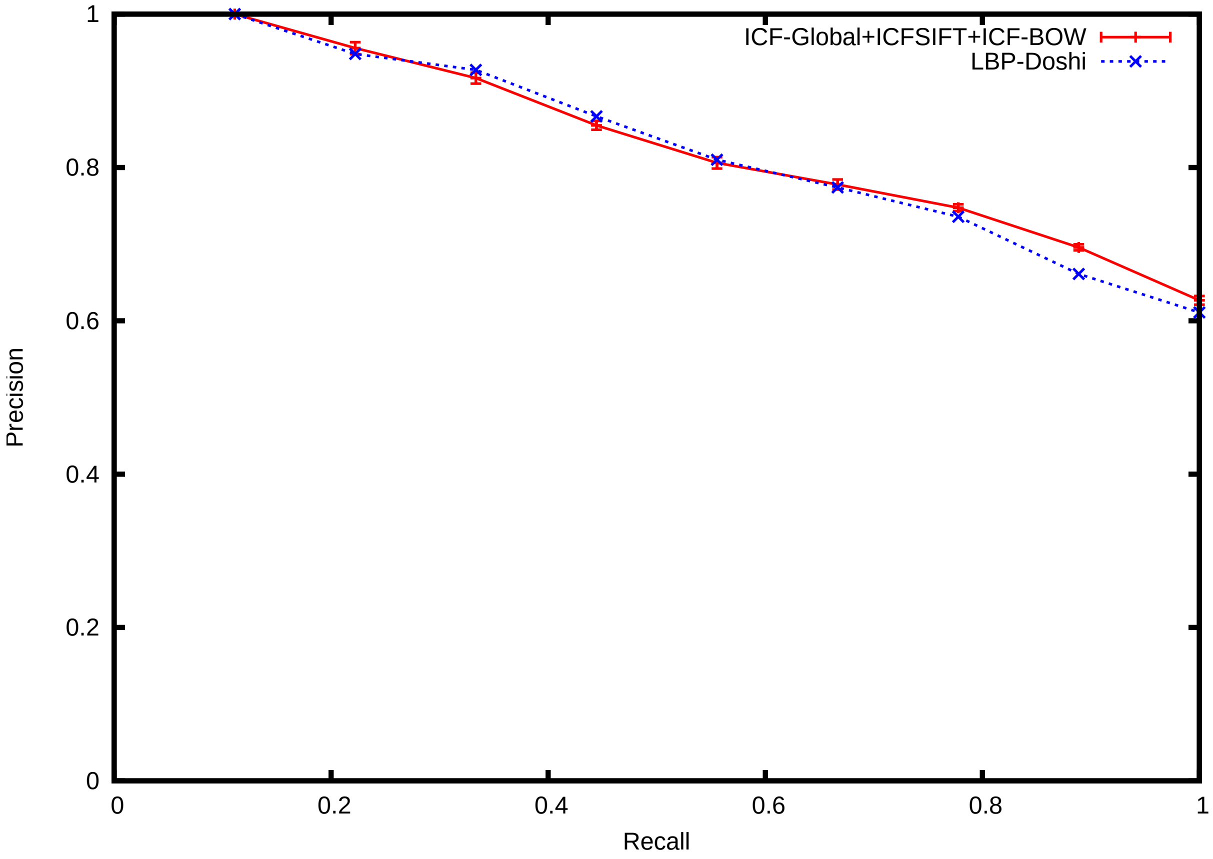


Figure 7.3: Performance of LBP-Doshi and ICF-based adaptive features on the Brodatz database.

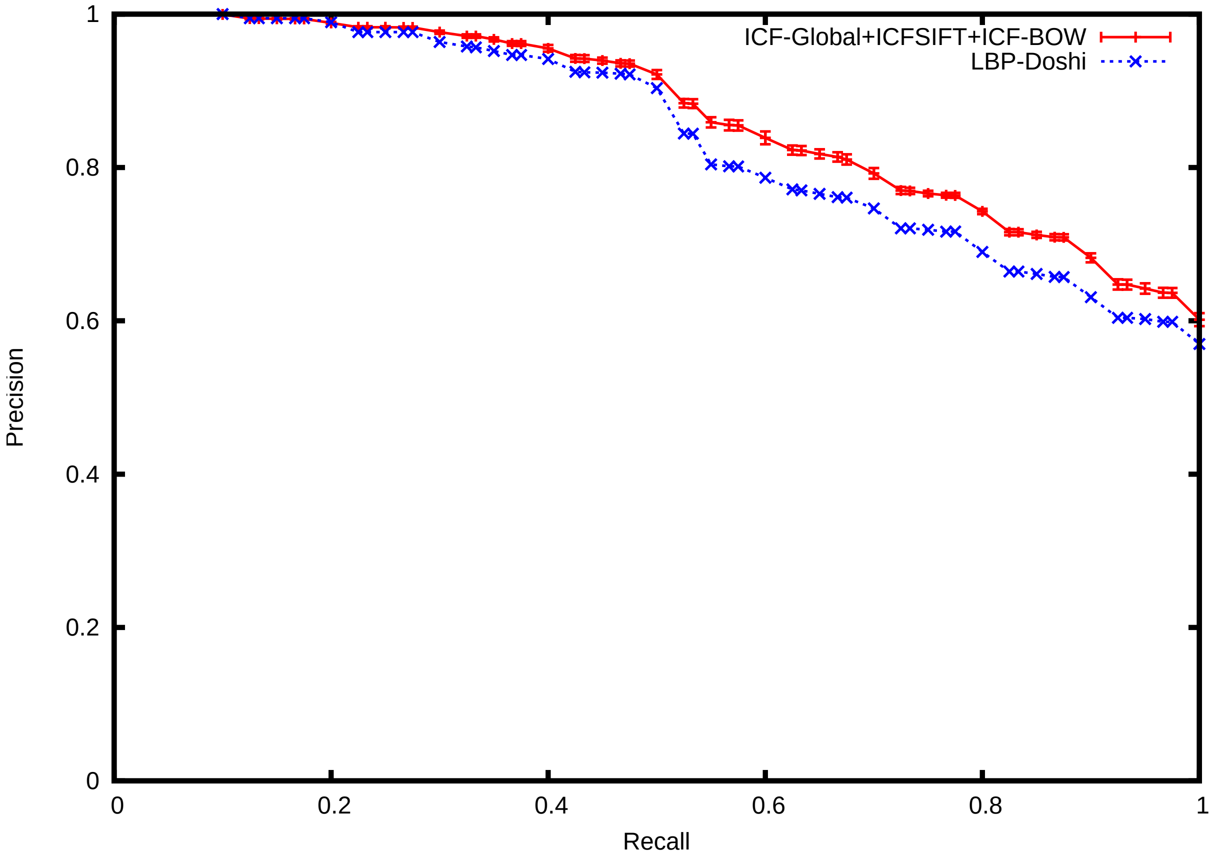


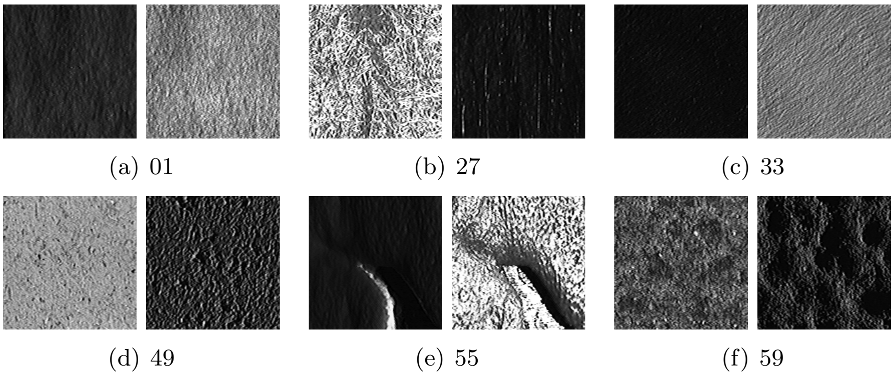
Figure 7.4: Performance of LBP-Doshi and ICF-based adaptive features on the VisTex database.

and the LB-Doshi features achieve an ARR of 77.10% percent. However the adaptive features reach a recall of one at a slightly higher precision compared to the LBP-Doshi features.

It is difficult to compare our VisTex performance with other work as we use a different relevance judgement. However in Appendix D we compare the performance of the the three ICF-based features with LBP-Doshi as well as other methods for a subset of 40 homogeneous VisTex textures. We call this the VisTex40 collection. We show that the ICF-based features are the best performing texture-only features with an ARR_{16} of 88.95%. The LBP-Doshi features also perform well on the VisTex40 collection with an ARR_{16} of 86.29%.

7.3.4 CUReT

The Columbia-Ultrecht Reflectance and Texture (CUReT) collection is a very challenging database and has been used widely in texton-based image classification (Varma and Zisserman, 2005; Zhang, Zhao and Liang, 2012). It has a total of 61 image classes with many real world textures. The images are obtained from a wide range of view and illumination directions. Images in the collection contain specularities, shadows and surface normal variation (Varma and Zisserman, 2005). They do not have significant scale variation, so non scale-invariant features usually perform well on this collection compared to scale invariant features (Hossain and Serikawa, 2013). Some example images are shown in Figure 7.5, where each pair of images belong to the same relevance class. Immediately it is apparent that variations in illumination intensity and directions within the same class poses a challenge for judging similarity.



(a) 01 (b) 27 (c) 33

(d) 49 (e) 55 (f) 59

Figure 7.5: Example images from the CureT collection.

The version in study has 92, 200 × 200 images per texture category. This version has been prepared by Varma and Zisserman (2009) and is often used by other studies (Hossain and Serikawa, 2013). Figure 7.6 shows that the ICF-based adaptive features perform much better than the LBP-Doshi features on this collection. LBP features are designed specifically to be invariant to illumination intensity, but even so on this collection the LBP-Doshi features do not perform as well as the adaptive features.

The ICF-based features achieve an ARR_{92} of 47.02% ± 0.028% and the LBP-Doshi features achieve 44.91%.

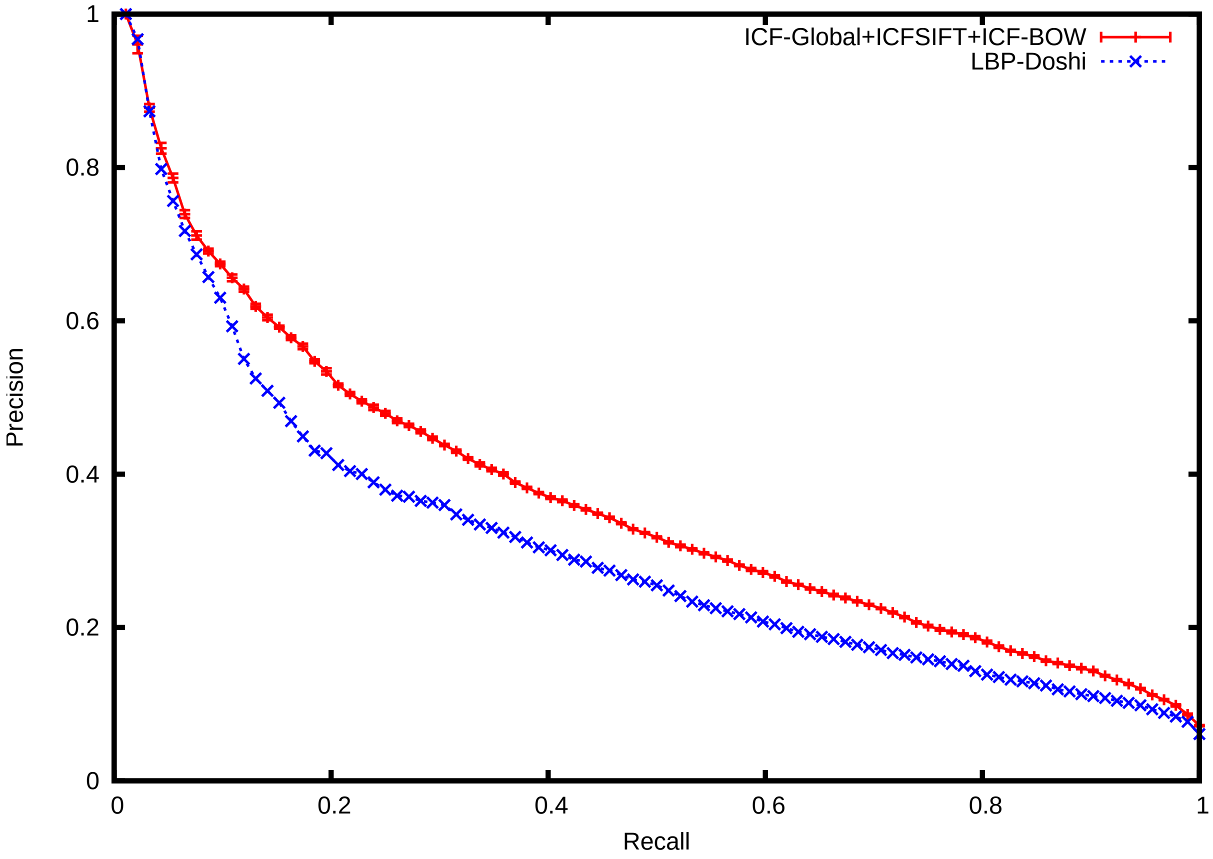


Figure 7.6: Performance of LBP-Doshi and ICF-based adaptive features on the CureT database.

7.4 Conclusion

In this chapter we have demonstrated that while hand-tuned features can in fact outperform automatic collection-specific features for the target collection, in cases when hand-tuning is not performed, using the automatic collection-specific ICF-based texture features can be a better alternative to using fixed pre-defined texture features, even if the fixed pre-defined features were hand-tuned for some other collection.

We used the LBP-Doshi feature, which was hand-tuned for the Outex_TR_0000 collection. We applied it to that collection and three other texture collections. For two out of the four texture collections, the adaptive features performed better than the LBP-Doshi features. For one the performance was comparable. The LBP-Doshi features performed better than the adaptive features for the Outex_TR_0000 collection, for which it was hand-tuned. However even for Outex_TR_0000, the adaptive features were the best non-LBP feature and in fact had better performance than 53 of the LBP features tried by Doshi and Schaefer (2012) . These results demonstrate the viability and utility of using automated collection-adapted texture features in CBIR systems.

Chapter 8

Future work

In this thesis we have presented the use of ICF-based collection-specific features that are automatically found from image collections. Our results are very promising and we would like to explore these ideas further. In the next sections we present some possible avenues of future work, and in chapter 9 we conclude this thesis.

8.1 Experimenting on larger image collections

So far we have used standard texture collections used for research. The largest of these had 6380 images in 319 different texture classes. In terms of practical applications these are small collections, so we need to study how the techniques developed in this thesis perform on much larger collections.

8.2 Experimenting on other class of images

In this thesis we conducted experiments on texture collections. The images in general have had globally consistent texture. Most real world cases will have images with much more diverse texture in them. We need to study whether adaptive features are effective in those cases and what other changes are necessary for these features to work in more general problem domains.

8.3 More work on filter selection

Our proposed clustering-based filter selection uses cross-correlation followed by complete-link clustering. While this choice of clustering algorithm suited our purpose well, there is a scope of exploring the performance of other methods to group filters. Given the presence of training data it might be even possible to select filters on completely different criteria, i.e. inter-class and intra-class response statistics, etc.

8.4 More work on ICFSIFT

We have utilised SIFT keypoint information in our proposed ICFSIFT. However this feature needs more study to have a better understanding of its scale and rotation invariance properties. Also, we have found it has performed better than standard SIFT for texture images, but we have not explored the application of ICFSIFT in object recognition or interest point matching. Work on this area will give a more robust understanding of the possible applications of ICFSIFT outside of texture collections.

With the current implementation, ICFSIFT extracts the local descriptors sequentially. With sufficiently large number of keypoints, this can be a slow process. However this calculation can easily be translated into a parallel version which can make the application of this feature suitable for systems with time constraints.

Also, so far we have used the the DoG-based keypoint detection mechanism proposed by Lowe (2004). There is scope to try other keypoint detectors that may give better texture information.

8.5 Per relevance class filters

So far we have worked with ICF extracted for entire collections. In cases where the relevance classes are known it is possible to extract ICF for each individual class and apply them to the problem of image similarity/distance judgement. The problem of filter selection will be more challenging in this case.

8.6 Texton-based dictionaries

The proposed ICFSIFT and ICF-BOW features employ ideas very similar to texton-based dictionaries, although the process has some differences. Instead of using pre-defined fixed filters, we would like to employ collection-specific ICF to create texton-based dictionaries. This work is currently in progress.

8.7 Extension into image classification and segmentation

So far we have only tested the utility of collection-specific features using CBIR. The results are positive and encourage the use of such features to other image processing applications, most immediately image classification and segmentation.

Chapter 9

Conclusion

The work reported in this thesis was built on the hypothesis that: "for many image processing applications adapted collection-specific features would be more effective at identifying useful and pertinent image content compared to pre-defined and fixed features, for the target image collection". There can be two ways to find such collection-specific features, either by human experimentation, or by automatic means. Most human attempts at hand-tuning features for particular collections involve trying different configurations of a set of features and choosing the combination that is deemed to perform best. This is almost always time consuming and in reality hard to do for every collection. In this thesis we concentrated on using automated means of finding collection-specific texture features. Instead of doing an automated imitation of a human's effort, we concentrated our work on using collection-specific independent component filters (ICF) found by ICA. We used standard texture collections and extracted ICA-based collection-specific filters for each collection. To compare the performance we used content-based image retrieval (CBIR) as the image processing application. In the context of CBIR we compared the performance of features extracted using these filters with other well known features: banks of Gabor filters, GLCM, HOG, PHOG, SIFT, LBP.

In our initial chapters, we showed that global features extracted using collection-specific ICF can indeed perform better than a pre-defined bank of Gabor filters. In fact we showed that ICF filters can outperform very large banks of Gabor filters. At the same time we showed that features which are extracted by sub-dividing an image into grids can be ineffective for images with globally consistent texture, if they encode spatially local information in their feature vectors.

For practical purposes we evaluated a previously published variance-based filter selection method used to choose a smaller subset of all the extracted ICF. We demonstrated the shortcomings of this technique and to address some of the problems with this method we developed the first contribution we reported: the use of response scaling. Response scaling was performed to ensure that filter response energies were scaled on a per-filter basis to prevent the response energies of different filters differing by orders of magnitudes. We demonstrated that the response-scaled energies were more effective when used as a global feature. The technique also improved the filter selection of the variance-based method, ensuring that it was selecting filters which had high variance due to the variation of its responses in the image classes, and not due to the energies varying by orders of magnitudes.

Our next contribution was the use of locally normalised convolution (LNC) to find the filter response energies. We demonstrated that LNC can be an effective method to deal with local intensity differences within an image which can influence standard convolution in a detrimental way. We have shown that using LNC improves the performance of filter-based features, with the ICF-based adaptive features having the greater improvement. We then demonstrated that combining the use of LNC and response scaling gave the best results.

The set of ICF extracted by ICA for a particular collection often contains shifted/near-duplicate filters. Even with the above mentioned improvements, the variance-based filter selection method is susceptible to selecting a subset of filters with these duplicate versions in them, leading to redundant features. To address this issue we proposed a filter selection technique that does not depend on filter responses but utilises the structure of the filter. Our technique used normalised cross-correlation as a similarity measure and complete-link clustering as a grouping technique to group similar filters together, along with a simple heuristic to choose a filter from each group. We demonstrated that our clustering-based filter selection method consistently outperforms the previously published variance-based filter selection method.

Earlier we had theorised that some grid-based features do not perform well on images with globally consistent texture because they encode spatially local information in their feature vector. So, we developed an adaptive local feature which utilises SIFT keypoints, ICFSIFT. We used the bag-of-words (BOW) method to create a spatial location independent image descriptor. We demonstrated that ICFSIFT consistently outperforms its pre-defined counterpart SIFT. It also outperformed the ICF-based adaptive global features. To clarify the utility of employing a keypoint-based technique we used ICF response energies (using LNC and response scaling) in a grid-based method to extract local features, but created the image descriptor using BOW. This ICF-BOW performed better than the ICFSIFT features. Their combination, however, resulted in significant increase in performance. We have shown that this improvement can be attributed to the scale and rotation invariance properties of ICFSIFT.

This thesis did not just document a comparison of features, but documented techniques which have resulted in a significant improvement in CBIR performance using just automated collection-specific texture features. Figure 9.1 shows the performance of ICF-based features at the start of the thesis compared to the performance achieved by its end.

While hand-tuned texture features can work well for the collection for which they were designed, the adaptive features proposed in this thesis perform better in the general case. This was shown using four standard texture collections, thus demonstrating the utility of the automatically learnt collection-specific features presented in this thesis.

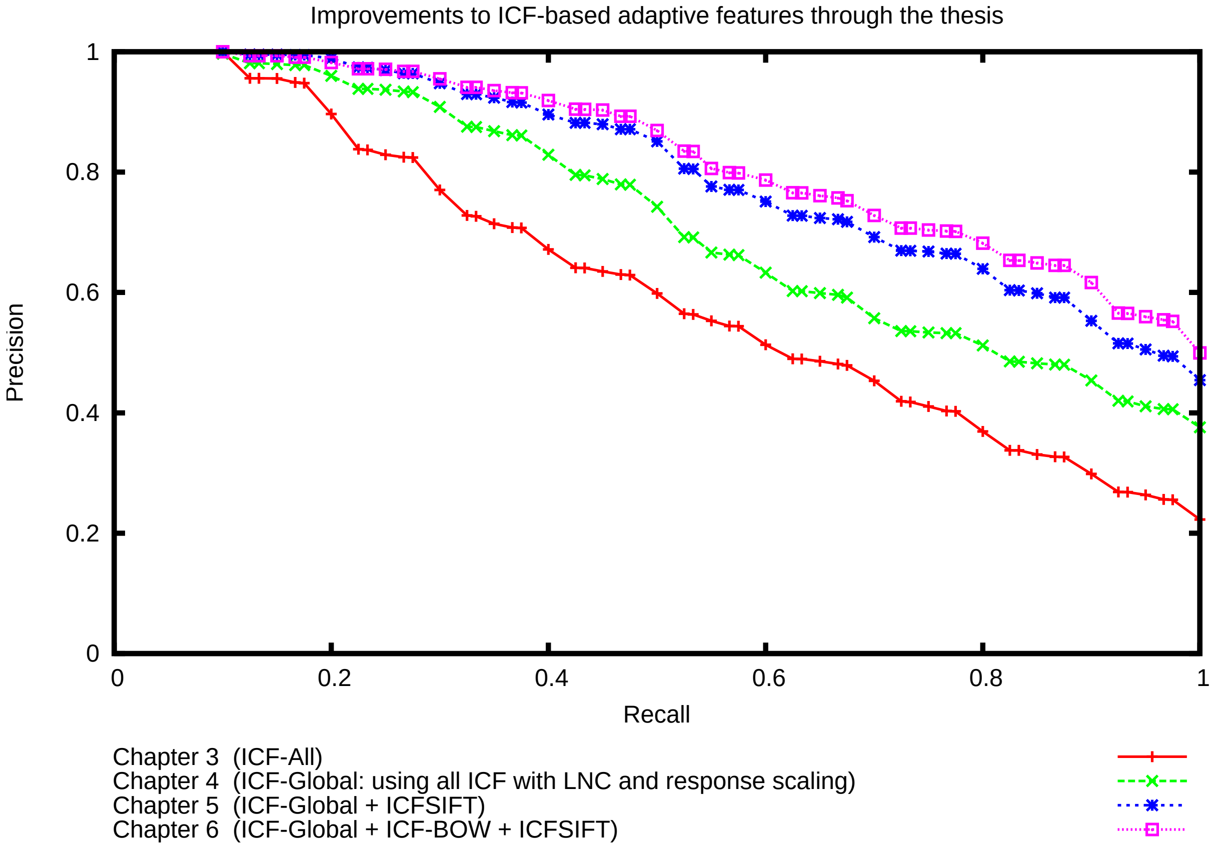


Figure 9.1: Performance gains, on the VisTex collection, of the ICF-based features through the thesis.

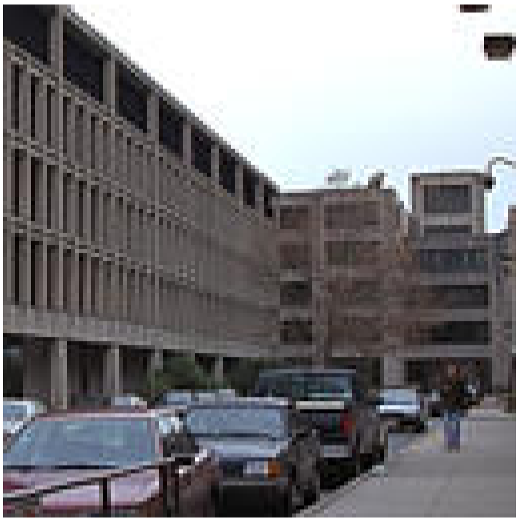
Appendix A

Similarity between HOG and GIFT like Gabor-based features

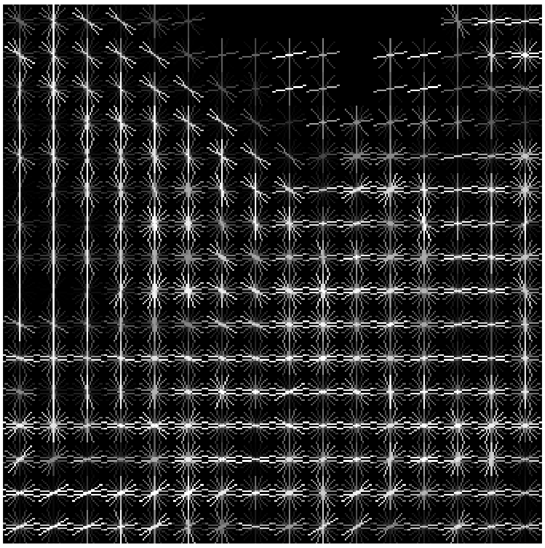
A.1 Similarity between HOG and GIFT features

The HOG descriptor calculates gradients within a cell after dividing the region using a grid-like method. The texture features used in the GNU Image Finding Tool (Müller, 2001; Squire et al., 1999) also divides an image into subregions using a grid-like method. Instead of calculating the gradients within each region, the GIFT texture features employ a bank of 12 circularly symmetric Gabor filters. The response energy of each filter is quantized into histogram bins. These histograms are then used to create entries into an inverted file index. These filters identify images features that are very similar to those extracted by HOG. The original implementation of the GIFT texture features used Gabor filters with 3 magnitudes and 4 orientations (Squire et al., 1999). To evaluate whether the HOG and GIFT features do indeed extract similar features, we replaced those filters with 9 filters of the same magnitude and orientations ranging from $0° - 180°$.

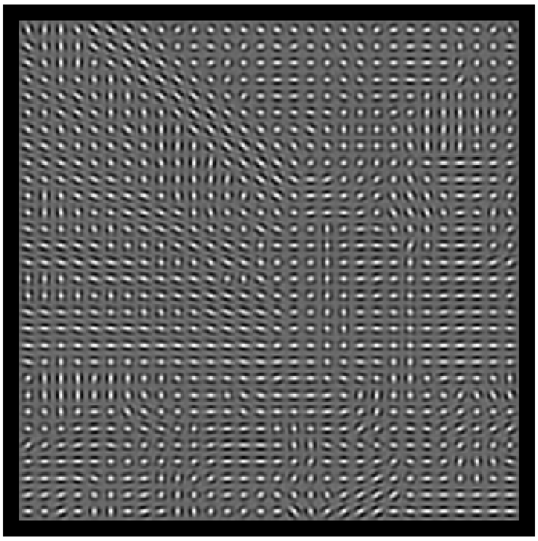
Figure A.1(a) shows an image of a building and some cars. There are strong edge orientations in this image making it a suitable test case. Figure A.1(b) shows the gradient image generated from the HOG features. Figure A.1(c) shows a similar image generated from the histogram entries of the 9 filters. Although there are some differences, by and large the GIFT features detect gradients similar to those that the HOG features detect.



(a) Original Image



(b) HOG gradients



(c) GIFT gradients

Appendix B

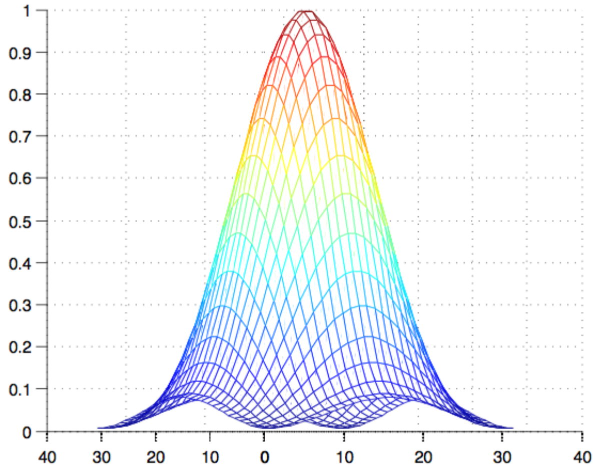
The use of Hamming windows

We have conducted some experiments to test ICF which do not have abrupt edges. To extract such filters we modified the process mentioned in §3.2 slightly. On each of the $10,000$ patches extracted from ICF extraction, we applied a two dimensional Hamming window as per the work of Le Borgne and Guérin-Dugué (2001). The Hamming window in one dimension is defined as in Equation B.1, where n is the size of the window.

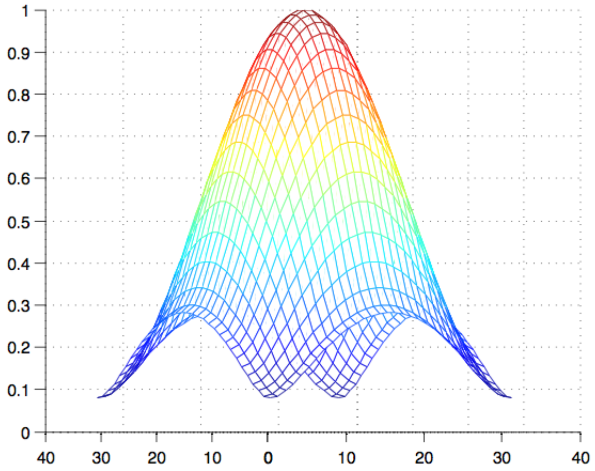
$$h(n) = 0.54 - 0.46 \cos\left(2\pi \frac{n}{N}\right) \quad 0 \leq n \leq N \tag{B.1}$$

We obtained the two dimensional window by $h_2 = h * h'$. From Figure B.1(d) shows the two dimensional Hamming window. This had a very sharp peak and discarded a large area of each patch, leading to filters with a very small useful region. We performed an element-wise square root of this two dimensional window to obtain the version shown in Figure B.1(e) which has a slightly wider peak and allows for more useful information to be retained. Figure B shows an example set of 175 such ICF.

However from Figure B.3 we see that in the context of CBIR using our setup there is no advantage in applying using such smoothing techniques. In-fact in both cases there is a slight drop in performance from using these smoothed filters.



(d) Two dimensional Hamming window



(e) Square rooted Two dimensional Hamming window

Figure B.1: Figure B.1(d) shows a two dimensional Hamming window. It has a sharp peak and discarded a large area of each patch. Figure B.1(e) shows the version used for experiments which is obtained by doing an element-wise square root of the two dimensional Hamming window giving a much flatter peak

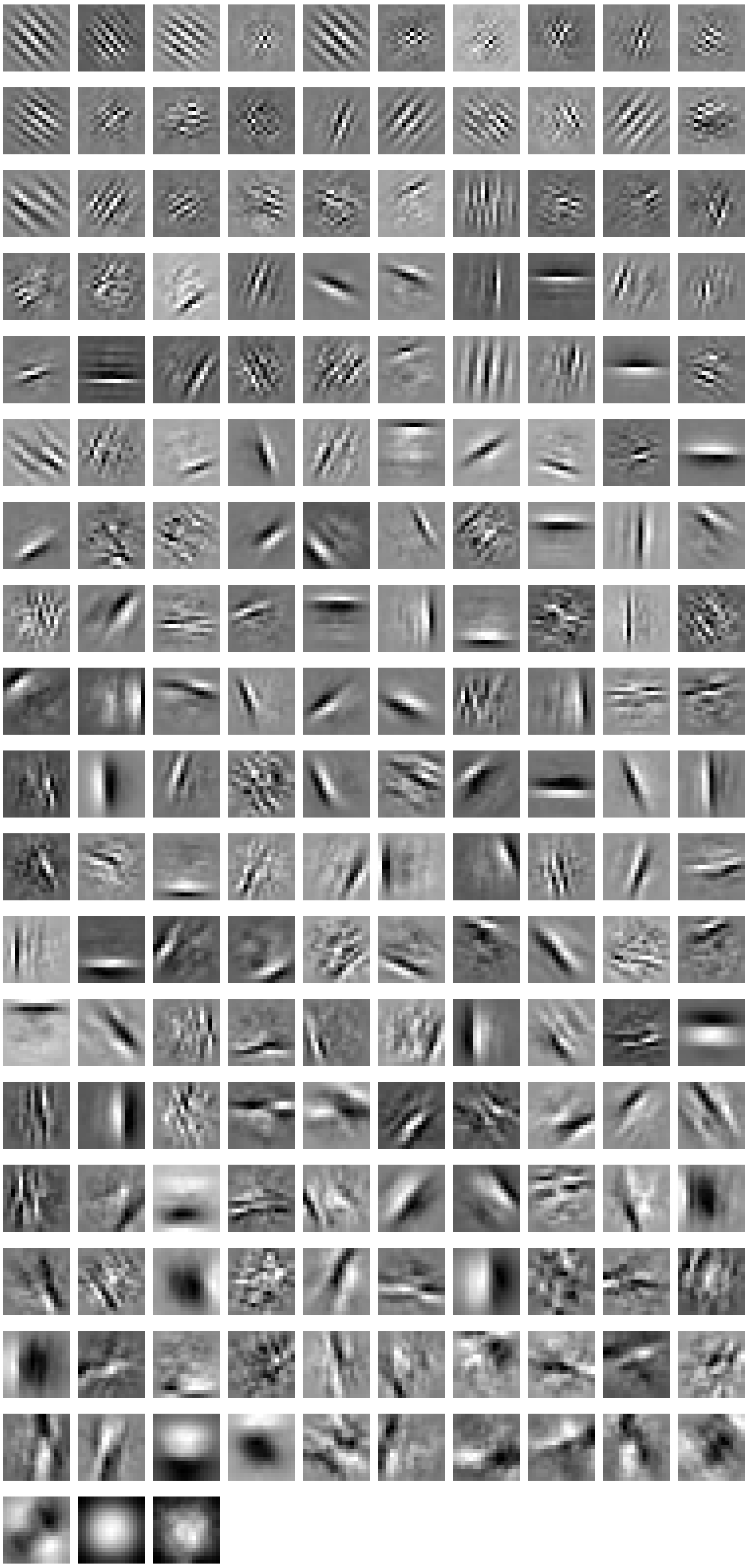
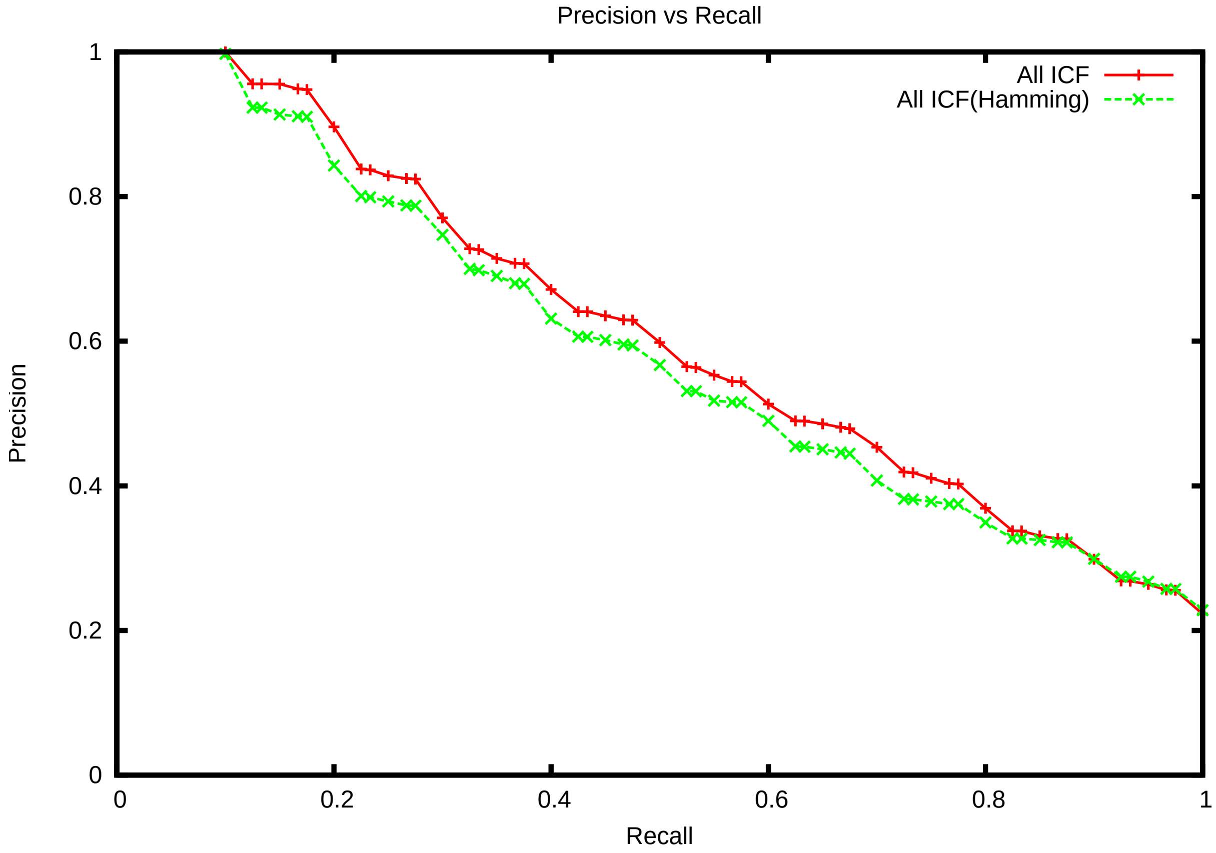
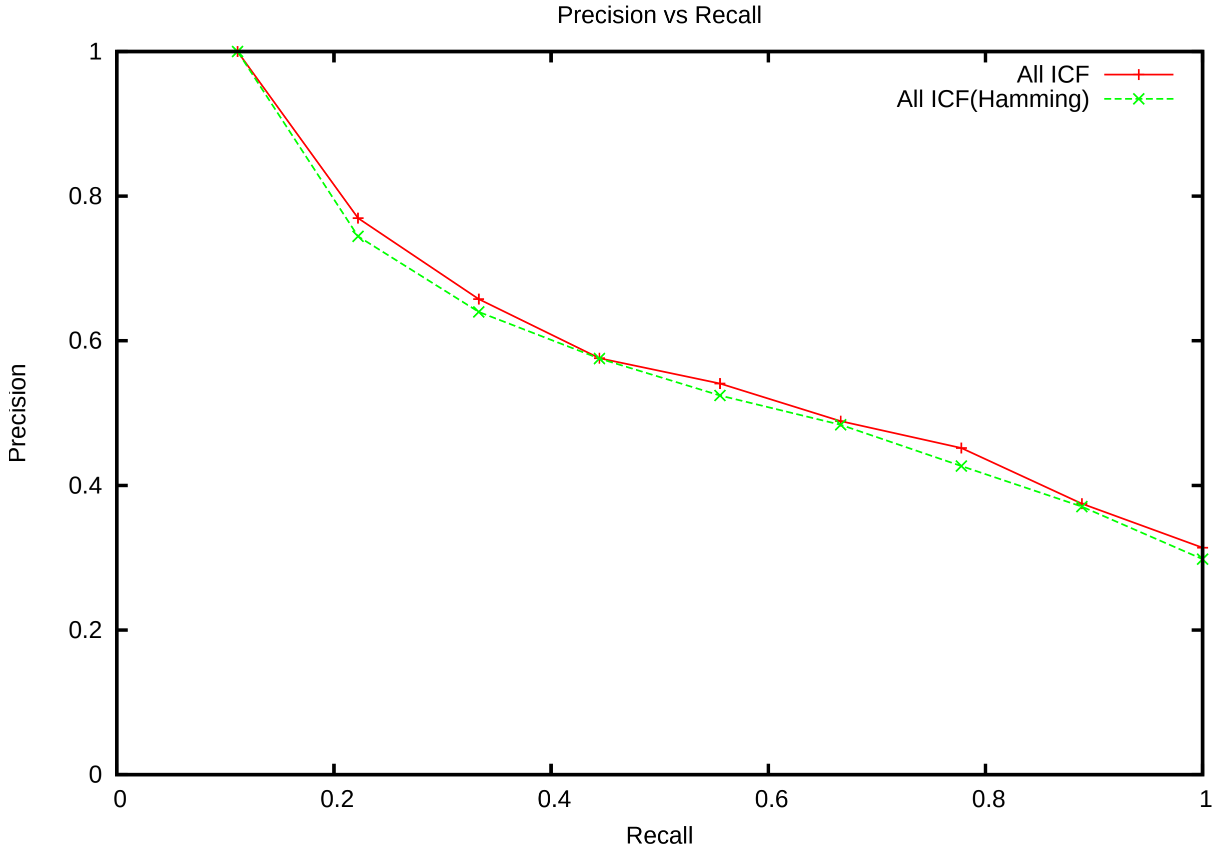


Figure B.2: Some filters extracted from patches which had a Hamming window applied to them. The smooth edges are noticeable in the filters.



(a) VisTex



(b) Brodatz

Figure B.3: Comparing the results of ICF extracted with and without Hamming windows

Appendix C

Principal component analysis

The goal of principal component analysis (PCA) and other such techniques is to find a representation of a multivariate data which gives a good representation with reduced redundancy. PCA uses the measure of correlation to determine redundancy and can be thought of as finding a transformation of the data into a new set of basis vectors which maximises the variance thereby making the transformed data uncorrelated.

The PCA model can be described in terms of a random vector v of size n_c which has zero mean. The assumption is that values are mutually correlation. If the values are already independent than PCA cannot find anything useful. We want to transform v to a new vector y with n_{uc} elements, where $n_{uc} \leq n_c$, so that the variables in y are uncorrelated. So,

$$y_1 = \sum_{k=1}^{n_c} w_{k1} v_k = w_1^T x, \tag{C.1}$$

where $v_1, v_2 ... v_{n_c}$ are the elements of the vector v and $w_1, w_2 .. w_{n_c}$ are elements of a unit norm vector w. y_1 would be considered to be a the first principal component if its variance is maximally large. Each successive component $y_2, y_3 ...$ also has the highest variance with the constraint that it is uncorrelated (or orthogonal) with the previous components. In practice the solution to the PCA problem is found in terms of the unit-norm eigenvectors $e_1, e_2 .. e_{n_c}$ of the covariance matrix, which is also the correlation matrix for zero mean random variables.

$$C_v = E\{vv^T\} \tag{C.2}$$

The eigenvectors are arranged in the order of their sorted eigenvalues $d_1, d_2 ... d_{n_c}$ such that $d_1 \geq d_2 \geq ... \geq d_{n_c}$. So, the first principal component is $y_1 = e_1^T x$ and the j^{th} principal component is $y_j = e_j^T x$.

The eigenvectors $e_1, e_2 .. e_{n_c}$ of the covariance matrix actually form the basis vectors of the transformed space in which the data is uncorrelated. The variance of a principal component y_j are just the eigenvalues of the covariance matrix because:

$$E\{y_j^2\} = E\{e_j^T v v^T e_j\} = e_j^T C_v e_j = d_j. \tag{C.3}$$

Small eigenvalues indicate that the corresponding principal component captures only a small part of the variance of the data. So, when trying to reduce dimensionality, the user can choose a smaller set of eigenvectors $e_1, e_2..e_s$ where $s < n_c$ so that their cumulative eigenvalues satisfies the user requirement of how much variance needs to be retained.

Whitening is an operation that transforms a zero-mean random vector v to z

$$z = Gv \tag{C.4}$$

where the elements of z are uncorrelated and has unit variance. The PCA transformation makes data uncorrelated, so all that is required is a scaling operation. The following linear transformation does both:

$$G = D^{-\frac{1}{2}}E^T. \tag{C.5}$$

The matrix E has the unit-norm eigenvectors of the covariance matrix C_v and D is the diagonal matrix of the eigenvalues.

Appendix D

VisTex40 results

When discussing the image collections used in this thesis we discussed the problems with the VisTex collection and how a smaller subset of 40 VisTex textures have been used to report results. Figure D.1 shows these 40 textures which are deemed to be homogeneous and have been used by some recent studies to report their results.

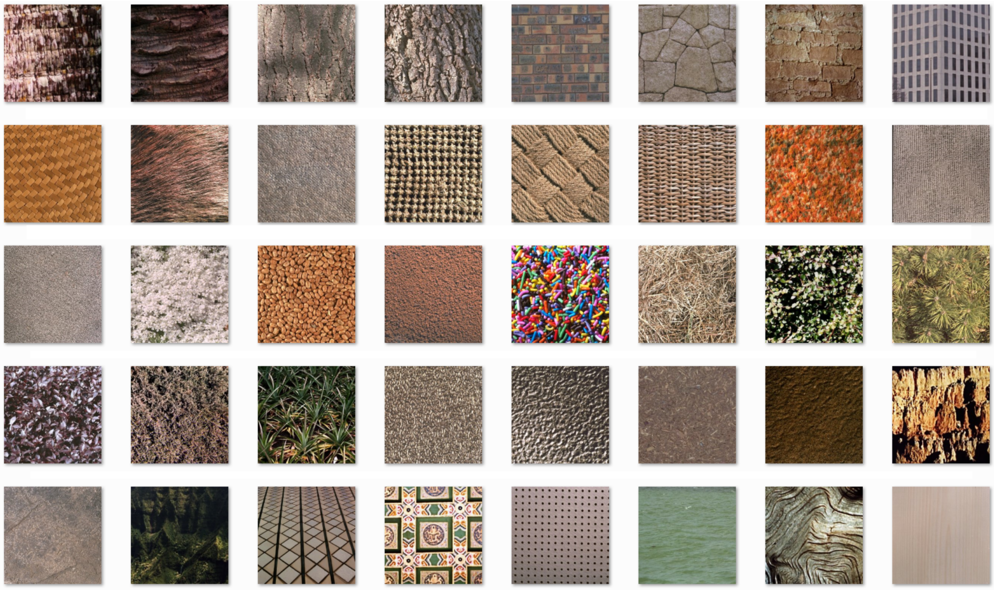


Figure D.1: 40 VisTex homogeneous textures used in multiple recente studies

.

Here we compare our results against the results presented by the four studies shown in Table D.1. All these studies use average recall rate (ARR) to report their results, which is defined in §7.2.2.

The studies divide each of the original 40 VisTex textures in 16 non-overlapping sub-images. So, there are 40 relevance classes, and each relevance class has 16 images. Every sub-image is used as a query, so $size(Q) = 640$ and they report results for $n = 16$.

The best results are reported by Bai, Zou, Kpalma and Ronsin (2012) and Bai, Kpalma and Ronsin (2012) with ARR close of 90.16% and 91.68%. Both these studies however extract features from each colour channel to achieve such high performance. The best results obtained just by using texture features is reported by Bombrun et al. (2011) at 79.38% and Stitou et al. (2009) at 83.45%.

Study	Colour and Texture	Just Texture
Bai, Zou, Kpalma and Ronsin (2012)	91.68%	Did not report
Bai, Kpalma and Ronsin (2012)	90.16%	Did not report
Bombrun et al. (2011)	88.23%	79.38%
Stitou et al. (2009)	Not done	**83.45**

Table D.1: State of the art results for the VisTex 40 collection

Our basic global ICF feature has and ARR_{16} of 77%. When we use the combined ICF-based features as done in chapter 5 and 7 and use the L_1 distance measure, we get an ARR of 89.95% which is the best performing texture only results. All results which are better than this use a combination of colour and texture. The LBP-Doshi features as described in chapter 7 also do very well with 86.29%, but the adaptive features perform better.

Features	Euclidean	CHI2	L_1
ICF-Global	77.30%	77.30%	76.87%
ICF-Global+ICFSIFT+ICF-BOW	83.67%	88.00%	**88.95**%
LBP-Doshi	83.33%	86.33%	86.29%

Table D.2: VisTex 40 results using features presented in this thesis.

Appendix E

Sparse filters vs CLC filters

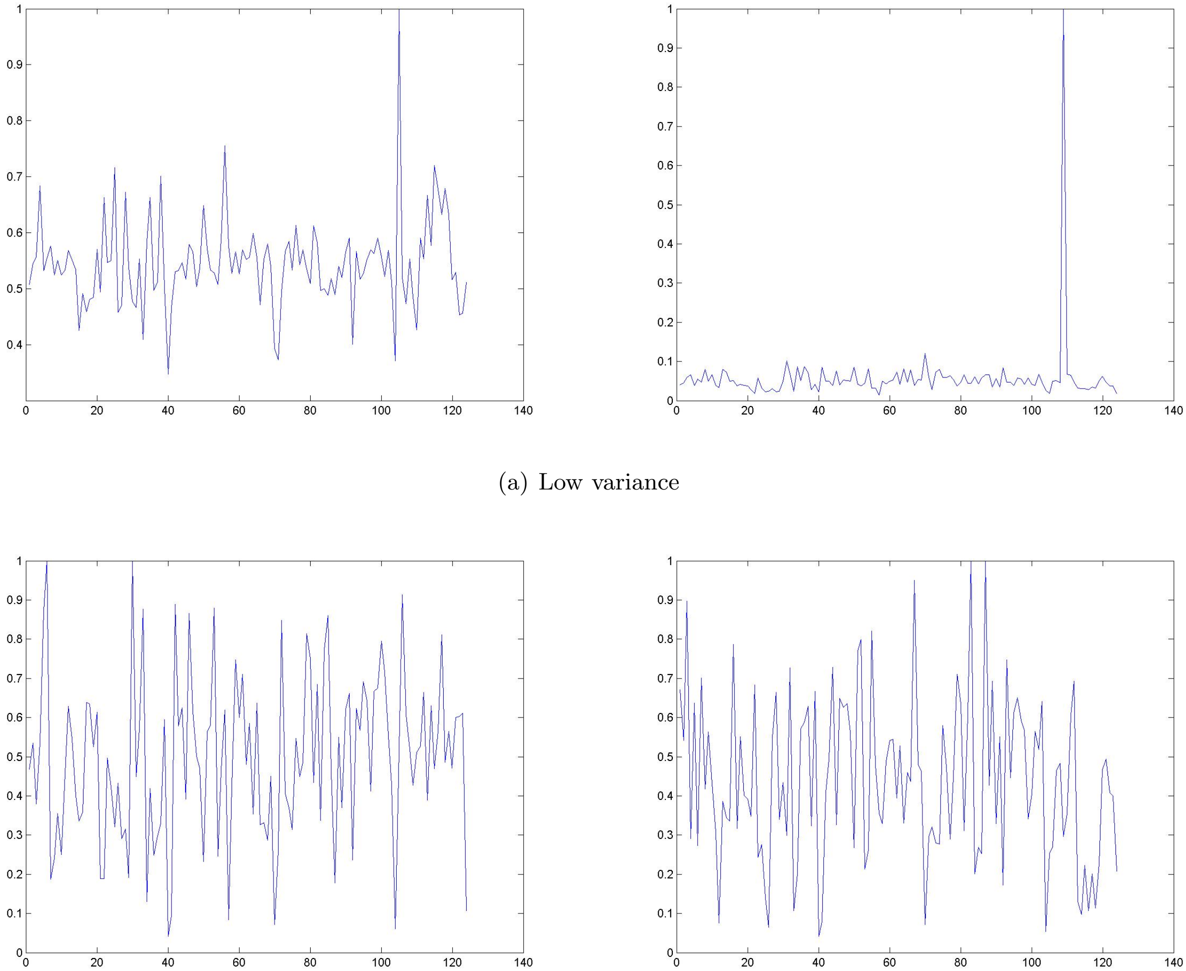
In Chapter 4 we mentioned that after using response scaling, the filter with the lowest variance exhibited greater sparseness compared to the filter with the highest variance. Figure E.1 shows the the response energy of four Vistex ICF, two with the lowest variance and two with the highest variance, calculated across 124 images, one coming from each relevance class. These energy values are obtained after using LNC and response scaling. We see that the low variance ICFs here also exhibit greater sparseness compared to the high variance ICFs.

As sparseness is a property of the simple cells of the human visual cortex and is a desirable property in feature extractors, it is reasonable to assume that these filters with low-variance in their response energies would perform well for CBIR. Figure E.2 shows 231 ICF extracted from the VisTex collection sorted in ascending order of the variance of their response energies across the 124 images. As discussed in Chapter 4, the low variance filters seem to be those with smaller scales. Figure E.3 shows 40 filters chosen using the clustering-based method. It also has quite a few filters with smaller scales, but also includes some filters which are clearly edge filters and have larger scales (and high variance).

We have proposed a clustering-based filter selection method in this thesis. In this appendix we compare the performance of 10, 20 and 40 filters with the lowest variance with an equal number of filters selected using the clustering-based method. The results are labelled as ICF-LowVar-N and ICF-CLC-N for N filters selected using the low-variance criteria and the clustering-based method respectively.

From Figure E.4 we can see that the filters chosen using the clustering-based method has better performance compared to the low-variance filter selection, when 10 and 20 filters are selected. When 40 filters are selected, the performance gap is very small, but the clustering-based filter selection still has better performance.

The VisTex collection has 124 query images, one from each relevance class. We calculated P(10) for each query image for ICF-CLC-40 and ICF-LowVar-40. Figure E.5 shows the 24 query images for which the ICF-LowVar-40 performed better than ICF-CLC-40. Figure E.6 shows the 37 query images for which the ICF-CLC-40 performed better than the ICF-LowVar-40.



(a) Low variance

(b) High variance

Figure E.1: Filter response energies of four filters. Two with the lowest variance and two with the highest.

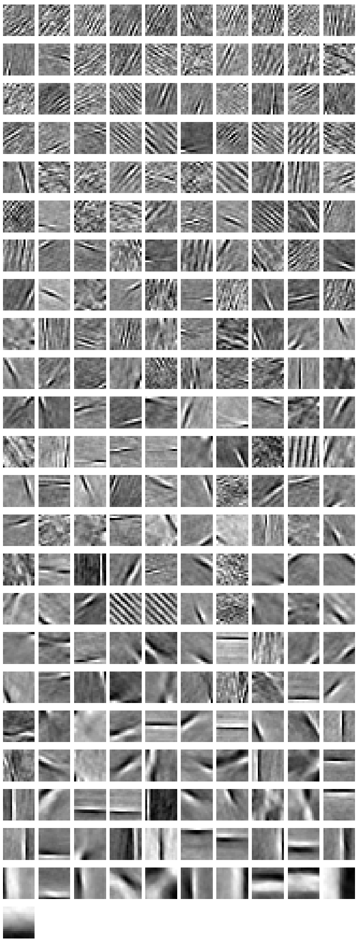


Figure E.2: ICF extracted for the VisTex collection sorted from lowest variance to highest

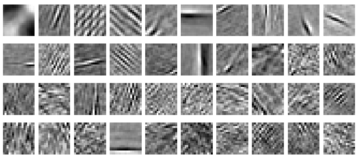
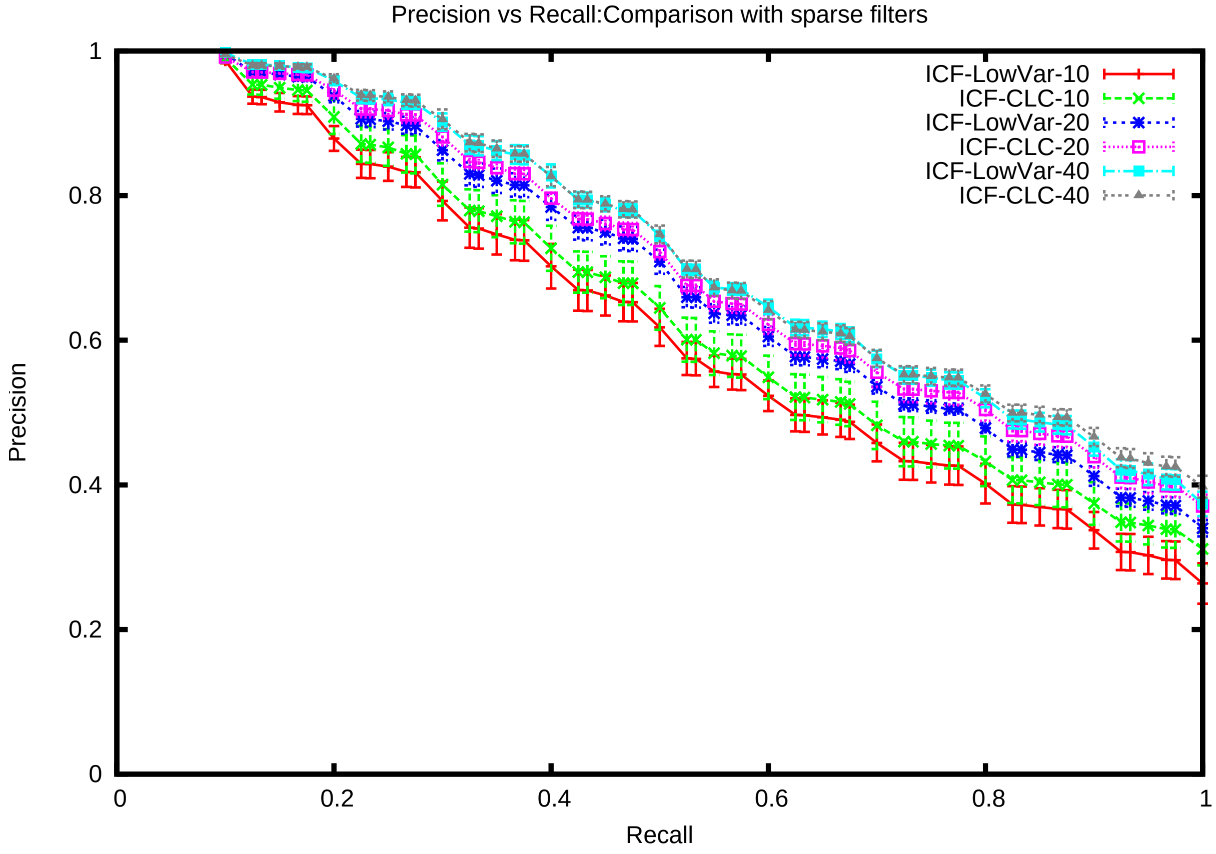
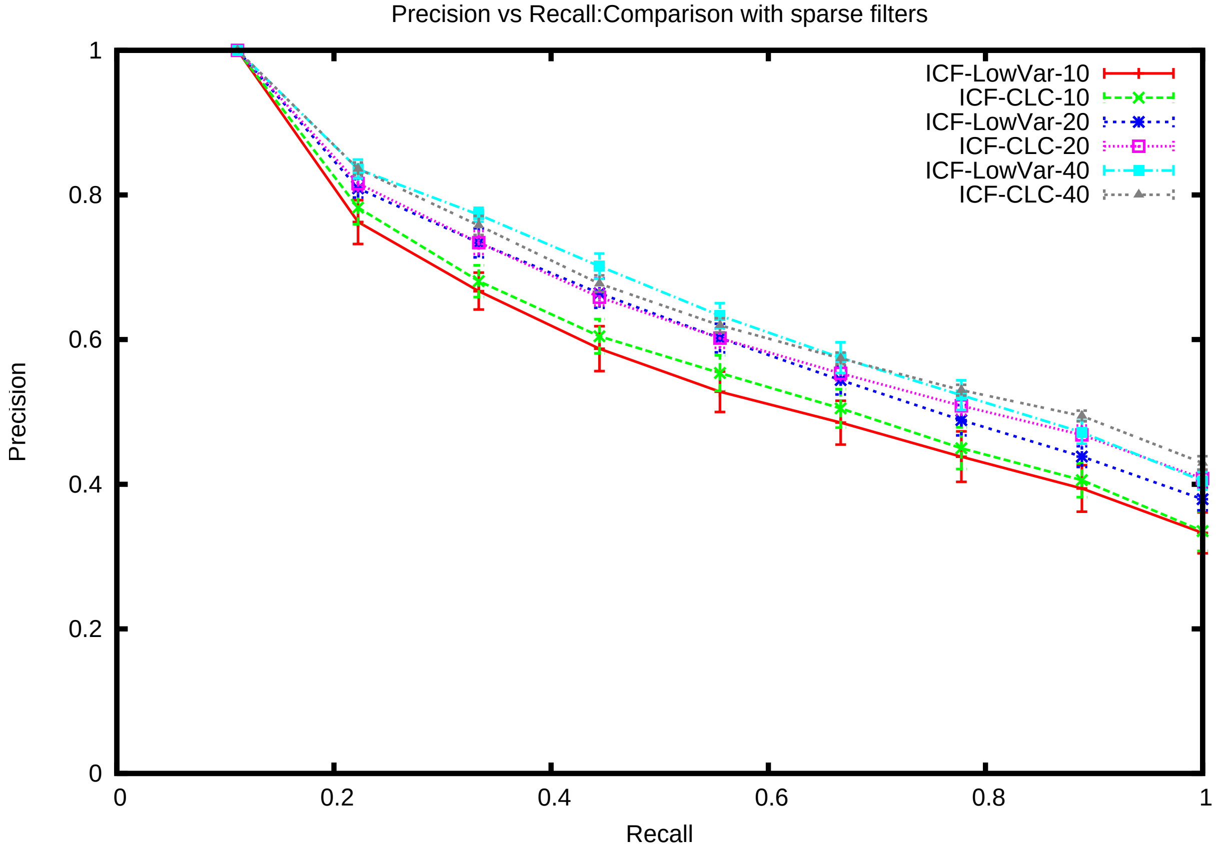


Figure E.3: 40 filters chosen using the clustering-based method

[htb!]



(a) VisTex



(b) Brodatz

Figure E.4: Performance of ICF-CLC-N and ICF-LowVar-N filters where N is 10,20 and 40

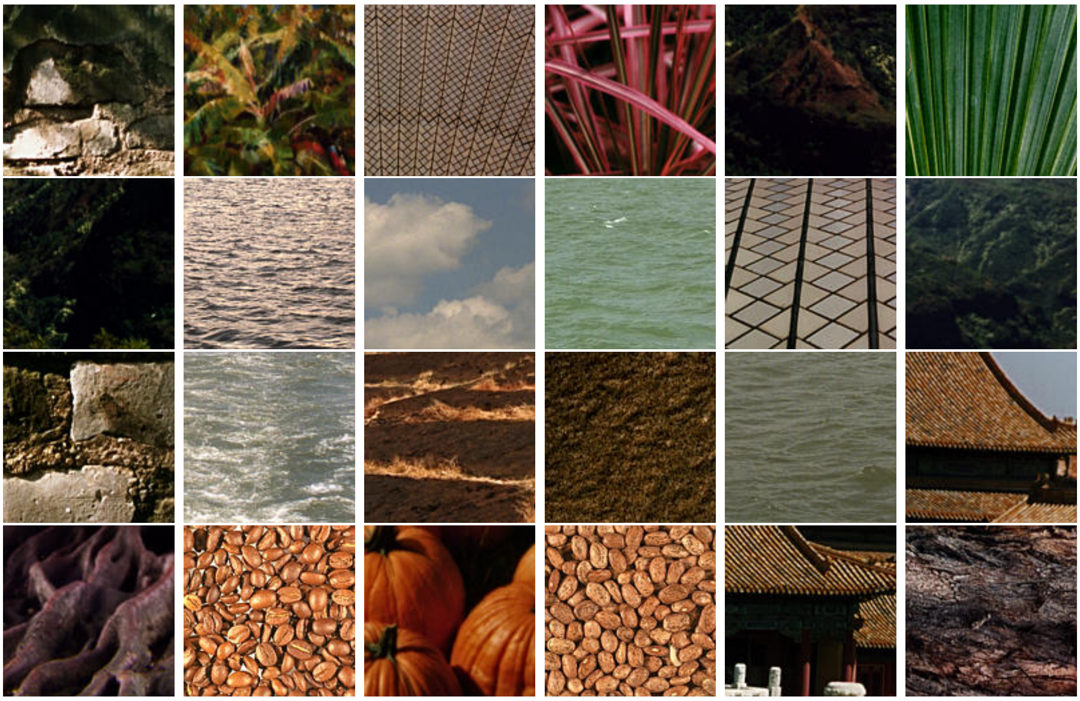


Figure E.5: The ICF-LowVar-40 filters perform better on these images compared to the ICF-CLC-40

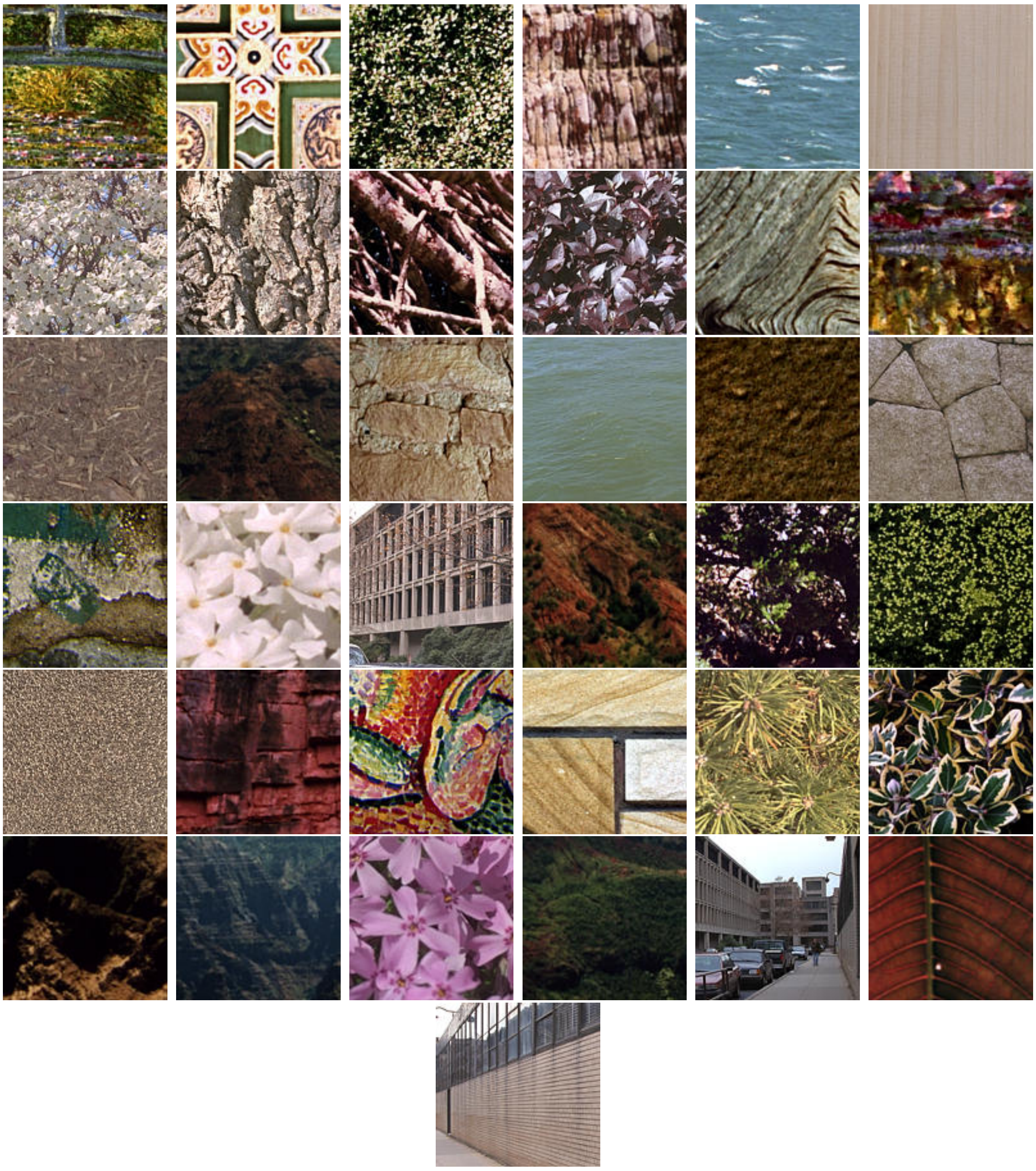


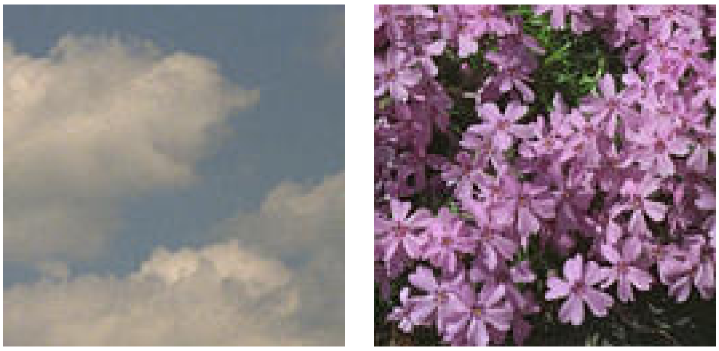
Figure E.6: The ICF-CLC-40 filters perform better on these images compared to the ICF-LowVar-40 filters

Appendix F

Analysis of the improved performance of ICF-BOW

In Chapter 6 we presented the ICF-BOW feature. This feature is extracted by sub-dividing an image into a 16 × 16 grid, similar to the ICF-Local feature presented in Chapter 3. However comparing the results we see that the ICF-BOW outperforms not only ICF-Local, but also ICF-Global. The difference between ICF-Local and ICF-BOW is the use of the Bag-of-Words (BOW) technique to create the image feature vector, which achieves independence from the spatial location of each grid block, whereas the ICF-Local feature mapped grid blocks to particular locations of the feature vector, thereby enforcing a spatial location dependence.

The question then is, why is spatial location dependence or independence important when dealing with images with globally consistent texture? We discussed in Chapter 2 that, even images which have globally consistent texture visually, are actually made up of multiple textures. We show two query images from the VisTex collection in Figure F.1. Figure F.1(a) has two obvious textures: the cloud and the flat background of the sky. Figure F.1(b) is maybe less obvious, but it too has at-least two textures: the actual flower petals and the leaves.



(a) Clouds_0000_sub0 (b) Flowers_0002_sub0

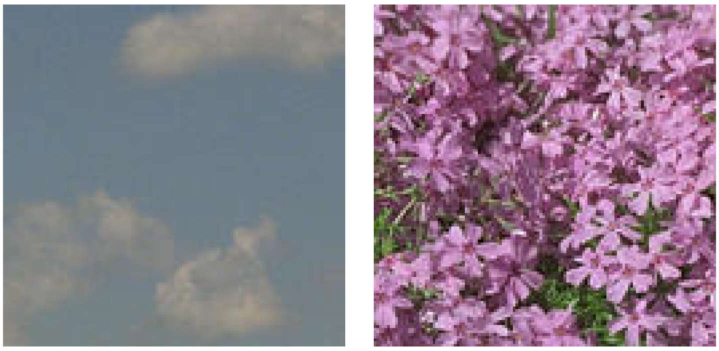
Figure F.1: Two query images from the VisTex collection

Now, ICF-Global is calculated by using the average filter energy, calculated on the whole image. This means, for two images to be judged to be similar, the textures present in each image would have to be similar, and the ratio of the individual textures present in each image would have to be similar, irrespective of their spatial location. If the ratios are quite different, the global energies would be quite different, even if the actual textures are similar.

ICF-Local encodes spatial location information in the feature vector. So, for two images to be judged to be similar, they would need to have similar textures present in similar grid locations, thus enforcing a much stricter criteria for similarity. This is reflected in its poor performance, as images with globally consistent visual texture tend to have the individual textures spread out in irregular patterns.

ICF-BOW, utilises a grid similar to ICF-Local, however as it does not encode spatially local information in the feature vector, it is much better suited to these class of images. In fact, ICF-BOW can be thought of as a feature that encodes the frequency of different textures that are present in an image.

Figures F.2(a) and F.2(b) are two images which are in the relevance classes of Figures F.1(a) and F.1(b) respectively. However they are not included in the first 10 results when using ICF-Global and ICF-Local. The ratio of the individual textures are different enough of ICF-Global to exclude them. The location of the individual textures are also different enough for ICF-Local to exclude them. However, these two images are included in the top 10 results when using ICF-BOW.



(a) Clouds_0001_sub6

(b) Flowers_0002_sub6

Figure F.2: Two images not included in the top 10 results by ICF-Global and ICF-Local, but found by ICF-BOW

Vita

Publications arising from this thesis include:

Nabeel Mohammed and David McG. Squire, ICFSIFT: Improving collection-specific CBIR with ICF-based local features, In Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA 2013), Hobart, Australia, November 26-28 2013.

Nabeel Mohammed and David McG. Squire, Improved Texture Features for CBIR using Response Scaling and Locally Normalised Convolution, In Proceedings of the 11th International Workshop on Content-Based Multimedia Indexing, Veszprm, Hungary, June 17-19 2013.

Nabeel Mohammed and David McG. Squire, Efficient and Accurate Independent Component Filter-based Features for Texure Similarity, In Proceedings of the 20th IEEE International Conference on Image Processing (ICIP), Melbourne, Australia, September 15-18 2013.

Nabeel Mohammed and David McG. Squire, An improved method for choosing effective Independent Component Filters for CBIR, In Proceedings of the 26th International Conference on Image and Vision Computing New Zealand, Auckland, New Zealand, pp. 517-522, November 29-December 1 2011.

Nabeel Mohammed and David McG. Squire, Effectiveness of ICF features for collection-specific CBIR, In Proceedings of the 9th International Workshop on Adaptive Multimedia Retrieval, co-located with the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011), Barcelona, Spain, No. 7836 in Lecture Notes in Computer Science, pp. 83-95, Springer-Verlag, July 18-19 2011.

Permanent Address: Clayton School of Information Technology
Monash University
Australia

This thesis was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this thesis were written by Glenn Maughan and modified by Dean Thompson and David Squire of Monash University.

References

Ahonen, T. and Pietikäinen, M. (2009). Image description using joint distribution of filter bank responses, *Pattern Recognition Letters* **30**(4): 368–376.

Amari, S.-i., Cichocki, A., Yang, H. H. et al. (1996). A new learning algorithm for blind signal separation, *Advances in neural information processing systems* pp. 757–763.

Arbter, K. (1989). Affine-invariant Fourier descriptors, *From Pixels to Features* pp. 153–164.

Arfanakis, K., Cordes, D., Haughton, V. M., Carew, J. D. and Meyerand, M. E. (2002). Independent component analysis applied to diffusion tensor MRI, *Magnetic resonance in medicine* **47**(2): 354–363.

Asada, H. and Brady, M. (1986). The curvature primal sketch, *IEEE Transactions on-Pattern Analysis and Machine Intelligence* **PAMI-8**(1): 2–14.

Bai, B., Kantor, P., Shokoufandeh, A. and Silver, D. (2007). fMRI brain image retrieval based on ICA components, *Proceedings of the Eighth Mexican International Conference on Current Trends in Computer Science, 2007. ENC 2007.*, pp. 10 –17.

Bai, C., Kpalma, K. and Ronsin, J. (2012). Color textured image retrieval by combining texture and color features, *Proceedings of the 20th European Signal Processing Conference (EUSIPCO), 2012*, IEEE, pp. 170–174.

Bai, C., Zou, W., Kpalma, K. and Ronsin, J. (2012). Efficient colour texture image retrieval by combination of colour and texture features in wavelet domain, *Electronics letters* **48**(23): 1463–1465.

Barlow, H. B. (1989). Unsupervised learning, *Neural computation* **1**(3): 295–311.

Bell, A. J. and Sejnowski, T. J. (1997). The “independent components” of natural scenes are edge filters, *Vision Research* **37**(23): 3327–3338.
URL: *http://dx.doi.org/10.1016/S0042-6989(97)00121-1*

Belongie, S., Carson, C., Greenspan, H. and Malik, J. (1998). Color-and texture-based image segmentation using EM and its application to content-based image retrieval, *Proceedings of the Sixth International Conference on Computer Vision, 1998.*, IEEE, pp. 675–682.

Belongie, S., Malik, J. and Puzicha, J. (2001). Matching shapes, *Proceedings of the Eighth IEEE International Conference on Computer Vision, 2001.*, Vol. 1, IEEE, pp. 454–461.

Berk, T., Kaufman, A. and Brownston, L. (1982). A human factors study of color notation systems for computer graphics, *Communications of the ACM* **25**(8): 547–550.

Berman, A. and Shapiro, L. (1999). Efficient content-based retrieval: experimental results, *Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries, 1999. (CBAIVL '99)*, pp. 55–61.

Bombrun, L., Berthoumieu, Y., Lasmar, N.-E. and Verdoolaege, G. (2011). Multivariate texture retrieval using the geodesic distance between elliptically distributed random variables, *Proceedings of the 18th IEEE International Conference on Image Processing, 2011*, IEEE, pp. 3637–3640.

Boquete, L., Ortega, S., Miguel-Jiménez, J. M., Rodríguez-Ascariz, J. M. and Blanco, R. (2012). Automated detection of breast cancer in thermal infrared images, based on independent component analysis, *Journal of medical systems* **36**(1): 103–111.

Bosch, A., Zisserman, A. and Munoz, X. (2007). Representing shape with a spatial pyramid kernel, *Proceedings of the 6th ACM international conference on Image and video retrieval*, ACM, pp. 401–408.

Bosch, A., Zisserman, A. and Muoz, X. (2007). Image classification using random forests and ferns, *Proceedings of the IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007.*, IEEE, pp. 1–8.

Brill, E. L. (1968). Character recognition via Fourier descriptors, *WESCON, session* **25**: 1–10.

Brodatz, P. (1966). *Textures: a photographic album for artists and designers*, Vol. 66, Dover New York.

Calhoun, V. D., Liu, J. and Adalı, T. (2009). A review of group ica for fmri data and ica for joint inference of imaging, genetic, and erp data, *Neuroimage* **45**(1): S163–S172.

Campbell, F. W. and Robson, J. G. (1968). Application of Fourier analysis to the visibility of gratings., *J Physiol* **197**(3): 551–566.
URL: *http://view.ncbi.nlm.nih.gov/pubmed/5666169*

Carson, C., Thomas, M., Belongie, S., Hellerstein, J. M. and Malik, J. (1999). Blobworld: A system for region-based image indexing and retrieval, *Visual Information and Information Systems*, Springer, pp. 509–517.

Chen, G., Bui, T. D. and Krzyżak, A. (2009). Invariant pattern recognition using radon, dual-tree complex wavelet and Fourier transforms, *Pattern Recognition* **42**(9): 2013–2019.

Chen, Y.-W., Zeng, X.-Y. and Lu, H. (2002). Edge detection and texture segmentation based on independent component analysis, *Proceedings of the 16th International Conference on Pattern Recognition, 2002.*, Vol. 3, IEEE, pp. 351–354.

Comon, P. (1994). Independent component analysis, a new concept?, *Signal processing* **36**(3): 287–314.

Cox, I. J., Miller, M. L., Omohundro, S. M. and Yianilos, P. N. (1996). Target testing and the `PicHunter` Bayesian multimedia retrieval system, *Advances in Digital Libraries (ADL'96)*, Library of Congress, Washington, D. C., pp. 66–75.
URL: *ftp://ftp.nj.nec.com/pub/ingemar/papers/adl96.ps*

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2005.*, Vol. 1, IEEE, pp. 886–893.

Daoudi, M. and Matusiak, S. (2000). Visual image retrieval by multiscale description of user sketches, *Journal of Visual Languages & Computing* **11**(3): 287–301.

Datta, R., Joshi, D., Li, J. and Wang, J. Z. (2008). Image retrieval: Ideas, influences, and trends of the new age, *ACM Computing Surveys (CSUR)* **40**(2): 5.

Daugman, J. (1980). Two-dimensional spectral analysis of cortical receptive field profiles, *Vision Research* **20**(10): 847–856.
URL: *http://dx.doi.org/10.1016/0042-6989(80)90065-6*

Deserno, T. M., Güld, M. O., Plodowski, B., Spitzer, K., Wein, B. B., Schubert, H., Ney, H. and Seidl, T. (2008). Extended query refinement for medical image retrieval, *Journal of Digital Imaging* **21**(3): 280–289.

Doshi, N. P. and Schaefer, G. (2012). A comprehensive benchmark of local binary pattern algorithms for texture retrieval, *Proceedings of the 21st International Conference on Pattern Recognition (ICPR), 2012.*, IEEE, pp. 2760–2763.

Duan, L., Ma, J., Miao, J. and Qiao, Y. (2009). A texture images segmentation method based on ica filters, *Proceedings of the Fifth International Conference on Natural Computation, 2009.*, Vol. 6, IEEE, pp. 484–487.

El-ghazal, A., Basir, O. and Belkasim, S. (2009). Farthest point distance: A new shape signature for Fourier descriptors, *Signal Processing: Image Communication* **24**(7): 572–586.

Faloutsos, C., Barber, R., Flickner, M., Hafner, J., Niblack, W., Petkovic, D. and Equitz, W. (1994). Efficient and effective querying by image content, *Journal of intelligent information systems* **3**(3-4): 231–262.

Fan, J., Gao, Y., Luo, H. and Xu, G. (2004). Automatic image annotation by using concept-sensitive salient objects for image content representation, *Proceedings of the*

27th annual international ACM SIGIR conference on Research and development in information retrieval, ACM, pp. 361–368.

Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Gorkani, M., Hafner, J., Lee, D., Petkovic, D. et al. (1995). Query by image and video content: The QBIC system, *IEEE Computer* **28**(9): 23–32.

Formisano, E., Esposito, F., Kriegeskorte, N., Tedeschi, G., Di Salle, F. and Goebel, R. (2002). Spatial independent component analysis of functional magnetic resonance imaging time-series: characterization of the cortical components, *Neurocomputing* **49**(1): 241–254.

Francos, J. M., Meiri, A. Z. and Porat, B. (1993). A unified texture model based on a 2-d wold-like decomposition, *IEEE Transactions on Signal Processing* **41**(8): 2665–2678.

Fu, K. S. and Albus, J. E. (1982). *Syntactic pattern recognition and applications*, Vol. 4, Prentice-Hall Englewood Cliffs, NJ.

Gonzalez, R. C. and Woods, R. E. (2007). *Digital Image Processing (3rd Edition)*, Prentice Hall.

Goshtasby, A. (1985). Description and discrimination of planar shapes using shape matrices, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **PAMI-7**(6): 738–743.

Haddad, S. (2005). Texture based image segmentation using textons, *Sixteenth Annual Symposium of the Pattern Recognition Association of South Africa*, Citeseer, p. 79.

Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J. and Stahel, W. A. (2011). *Robust statistics: the approach based on influence functions*, Vol. 114, Wiley. com.

Haralick, R. M. (1979). Statistical and structural approaches to texture, *Proceedings of the IEEE* **67**(5): 786–804.

Haralick, R. M., Shanmugam, K. and Dinstein, I. (1973). Textural features for image classification, *IEEE Transactions on Systems, Man and Cybernetics* **3**(6): 610–621.

Hawkins, J. K. (1970). Textural properties for pattern recognition, *Picture processing and psychopictorics* pp. 347–370.

He, Q. (1997). An evaluation on MARS-an image indexing and retrieval system, *Technical report, Graduate School of Library and Information Science, University of Illinois at Urbana-Champaign, Champaign, Ip* **61820**.

Hervé, N. and Boujemaa, N. (2007). Image annotation: which approach for realistic databases?, *Proceedings of the 6th ACM international conference on Image and video retrieval*, ACM, pp. 170–177.

Ho, J.-M., Lin, S.-Y., Fann, C.-W., Wang, Y.-C. and Chang, R.-I. (2012). A novel content based image retrieval system using K-means with feature extraction, *Proceedings of the 2012 International Conference on Systems and Informatics*, IEEE, pp. 785–790.

Hossain, S. and Serikawa, S. (2013). Texture databases–a comprehensive survey, *Pattern Recognition Letters* .

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components, *The Journal of educational psychology* pp. 498–520.

Howarth, P. and Rüger, S. (2004). Evaluation of texture features for content-based image retrieval, *Proceedings of the Third International Conference on Image and Video Retrieval (CIVR 2004)*, number 3115 in *Lecture Notes in Computer Science*, Springer-Verlag, Dublin, Ireland, pp. 326–334.
URL: *http://www.springerlink.com/link.asp?id=ywqv0229t56fkwfa*

Howarth, P., Yavlinsky, A., Heesch, D. and Roger, S. (2005). Medical image retrieval using texture, locality and colour, *in* C. Peters, P. Clough, J. Gonzalo, G. Jones, M. Kluck and B. Magnini (eds), *Multilingual Information Access for Text, Speech and Images*, Vol. 3491 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 918–918. 10.1007/1151964572.
URL: *http://dx.doi.org/10.1007/1151964572*

Hu, X., Li, K., Han, J., Hua, X., Guo, L. and Liu, T. (2012). Bridging the semantic gap via functional brain imaging, *IEEE Transactions on Multimedia* **14**(2): 314–325.

Hubel, D. H. and Wiesel, T. N. (1968). Receptive fields and functional architecture of monkey striate cortex, *The Journal of physiology* **195**(1): 215–243.

Hvyarinen, A. (1998). New approximations of differential entropy for independent component analysis and projection pursuit, *Advances in Neural Information Processing Systems* **10**: 273–279.

Hyvarinen, A. (1999). Fast and robust fixed-point algorithms for independent component analysis, *IEEE Transactions on Neural Networks* **10**(3): 626–634.

Hyvärinen, A. and Hoyer, P. (2000). Emergence of topography and complex cell properties from natural images using extensions of ICA, *Advances in neural information processing systems* **12**: 827–833.

Hyvarinen, A., Hoyer, P. and Inki, M. (2000a). Topographic ICA as a model of V1 receptive fields, *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, 2000.*, Vol. 4, IEEE, pp. 83–88.

Hyvärinen, A. and Hoyer, P. O. (2001). Topographic independent component analysis as a model of v1 organization and receptive fields, *Neurocomputing* **38**: 1307–1315.

Hyvärinen, A., Hoyer, P. O. and Inki, M. (2000b). Topographic ICA as a model of natural image statistics, *Biologically Motivated Computer Vision*, Springer, pp. 535–544.

Hyvärinen, A., Hoyer, P. O. and Inki, M. (2001). Topographic independent component analysis, *Neural computation* **13**(7): 1527–1558.

Hyvarinen, A., Karhunen, J. and Oja, E. (2001). *Independent Component Analysis*, Wiley-Interscience.

Hyvärinen, A. and Oja, E. (1997). A fast fixed-point algorithm for independent component analysis, *Neural computation* **9**(7): 1483–1492.

Jain, A. K., Murty, M. N. and Flynn, P. J. (1999). Data clustering: a review, *ACM Computer Survey* **31**(3): 264–323.
URL: *http://dx.doi.org/10.1145/331499.331504*

Joblove, G. H. and Greenberg, D. (1978). Color spaces for computer graphics, *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '78, ACM, New York, NY, USA, pp. 20–25.

Kaplan, S. T. and Ulrych, T. J. (2003). Blind deconvolution and ICA with a banded mixing matrix, *Online Proceedings of the 4th International Symposium on Independent Component Analysis and Blind Signal Separation*, pp. 591–596.

Kaur, P. and Mann, K. S. (2012). A novel algorithm for region based image retrieval framework, *International Journal of Innovations in Engineering and Technology (IJIET)* .

Ke, Y. and Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors, *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004.*, Vol. 2, IEEE, pp. II–506.

Kekre, D. H., Mishra, M. D. and Kariwala, M. A. (2011). Survey of CBIR Techniques and Semantics, *International journal of Engineering science and Technology (IJEST)* **3**(5).

Kelly, P. M., Cannon, M. and Hush, D. R. (1995). Query by image example: the CANDID approach, *in* W. Niblack and R. C. Jain (eds), *Storage and Retrieval for Image and Video Databases III*, Vol. 2420 of *SPIE Proceedings*, pp. 238–248.

Kent, A., Berry, M. M., Luehrs, F. U. and Perry, J. W. (1955). Machine literature searching VIII. operational criteria for designing information retrieval systems, *American documentation* **6**(2): 93–101.

Khaparde, A., B.L.Deekshatulu, M.Madhavilatha, Farheen, Z. and Sandhya Kumari, V. (2008). Content based image retrieval using independent component analysis, *IJCSNS International Journal of Computer Science and Network Security* **8**(4): 327–332.

Khaparde, A., Jain, N., Mantha, S. and Chowdary, N. S. (2011). Searching query by color content of an image using independent component analysis, *International journal of Enterprise and business Systems* **1**(2).

Kosch, H. and Maier, P. (2010). Content-based image retrieval systems–reviewing and benchmarking., *JDIM* **8**(1): 54–64.

La Cascia, M. and Ardizzone, E. (1996). JACOB: just a content-based query system for video databases, *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, 1996.*, IEEE Computer Society, Washington, DC, USA, pp. 1216–1219.

Lazebnik, S., Schmid, C. and Ponce, J. (2005). A sparse texture representation using local affine regions, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(8): 1265–1278.

Lazebnik, S., Schmid, C. and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006*, Vol. 2, IEEE, pp. 2169–2178.

Le Borgne, H. and Guérin-Dugué, A. (2001). Sparse-dispersed coding and images discrimination with independent component analysis, *Proceedings of the Third International Conference on Independent Component Analysis and Signal Separation (ICA'2001).*

Le Borgne, H., Guérin-Dugué, A. and Antoniadis, A. (2004). Representation of images for classification with independent features, *Pattern Recognition Letters* **25**(2): 141–154. **URL:** *http://dx.doi.org/10.1016/j.patrec.2003.09.011*

Lee, K.-L. and Chen, L.-H. (2005). An efficient computation method for the texture browsing descriptor of MPEG-7, *Image and Vision Computing* **23**(5): 479–489.

Lehmann, T. M., Gold, M., Thies, C., Fischer, B., Spitzer, K., Keysers, D., Ney, H., Kohnen, M., Schubert, H. and Wein, B. B. (2004). Content-based image retrieval in medical applications, *Methods of Information in Medicine* **43**(4): 354–361.

Leung, T. and Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons, *International Journal of Computer Vision* **43**(1): 29–44.

Lewis, J. (1995). Fast template matching, *Vision Interface*, Vol. 95, pp. 15–19.

Li, Z., Wu, S., Wang, X., Ye, H., Wang, M. and Ye, J. (2009). Dimensional reduction based on independent component analysis for content based image retrieval, *Proceedings of the International Joint Conference on Artificial Intelligence, 2009.*, IEEE, pp. 741–745.

Liao, S., Law, M. W. and Chung, A. C. (2009). Dominant local binary patterns for texture classification, *IEEE Transactions on Image Processing* **18**(5): 1107–1118.

Liao, S. X. and Pawlak, M. (1996). On image analysis by moments, *IEEE Transactions on Pattern analysis and machine intelligence.* **18**(3): 254–266.

Liu, F. and Picard, R. W. (1996). Periodicity, directionality, and randomness: Wold features for image modeling and retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* **18**(7): 722–733.

Long, L. R., Antani, S., Deserno, T. M. and Thoma, G. R. (2009). Content-based image retrieval in medicine: retrospective assessment, state of the art, and future directions, *International journal of healthcare information systems and informatics: official publication of the Information Resources Management Association* **4**(1): 1.

Lowe, D. G. (1999). Object recognition from local scale-invariant features, *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2*, ICCV '99, IEEE Computer Society, Washington, DC, USA, pp. 1150–.
URL: *http://dl.acm.org/citation.cfm?id=850924.851523*

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision* **60**(2): 91–110.
URL: *http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94*

Lu, G. and Sajjanhar, A. (1999). Region-based shape representation and similarity measure suitable for content-based image retrieval, *Multimedia Systems* **7**(2): 165–174.

Lu, Z.-M., Li, S.-Z. and Burkhardt, H. (2006). A content-based image retrieval scheme in JPEG compressed domain, *International Journal of Innovative Computing, Information and Control* **2**(4): 831–839.

Lux, M. (2009). Caliph & Emir: MPEG-7 photo annotation and retrieval, *Proceedings of the 17th ACM international conference on Multimedia*, ACM, pp. 925–926.

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

Marques, O. and Furht, B. (2002). *Content-Based Image and Video Retrieval*, Kluwer Academic Publishers, Norwell, MA, USA.

Materka, A., Strzelecki, M. et al. (1998). Texture analysis methods–a review, *Technical university of lodz, institute of electronics, COST B11 report, Brussels* pp. 9–11.

Mikolajczyk, K. and Schmid, C. (2004). Scale & affine invariant interest point detectors, *International journal of computer vision* **60**(1): 63–86.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(10): 1615–1630.

Mohammed, N. and Squire, D. M. (2011a). Effectiveness of ICF features for collection-specific CBIR, *Proceedings of the 9th International Workshop on Adaptive Multimedia Retrieval, co-located with the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, Barcelona, Spain.
URL: *http://www.csse.monash.edu.au/˜davids/publications/postscript/2011/AMR2011.pdf*

Mohammed, N. and Squire, D. M. (2011b). An improved method for choosing effective Independent Component Filters for CBIR, *Proceedings of the 26th International Conference on Image and Vision Computing New Zealand*, Auckland, New Zealand.

Mohammed, N. and Squire, D. M. (2013a). Efficient and accurate independent component filter-based features for texure similarity, *Proceedings of the 20th IEEE International Conference on Image Processing (ICIP)*, Melbourne, Australia.

Mohammed, N. and Squire, D. M. (2013b). ICFSIFT: Improving collection-specific CBIR with ICF-based local features, *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA 2013)*, Hobart, Australia.

Mohammed, N. and Squire, D. M. (2013c). Improved texture features for CBIR using response scaling and locally normalised convolution, *Proceedings of the 11th International Workshop on Content-Based Multimedia Indexing*, Veszprém, Hungary.

Mokhtarian, F. and Mackworth, A. (1986). Scale-based description and recognition of planar curves and two-dimensional shapes, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* **PAMI-8**(1): 34–43.

Müller, H., Müller, W., Squire, D. M., Marchand-Maillet, S. and Pun, T. (2001). Performance evaluation in Content-Based Image Retrieval: Overview and Proposals, *Pattern Recognition Letters* **22**(5): 593–601. (special issue on Image/Video Indexing and Retrieval).
URL: *http://dx.doi.org/10.1016/S0167-8655(00)00118-5*

Müller, H., Rosset, A., Vallee, J. and Geissbuhler, A. (2003). Comparing feature sets for content based image retrieval in a medical case database, *Proceedings of the Medical Informatics Europe Conference 2003 (MIE2003)*, St. Malo, France.

Müller, W. (2001). *Design and implementation of a flexible Content–Based Image Retrieval Framework - The GNU Image Finding Tool*, PhD thesis, Computer Vision and Multimedia Laboratory, University of Geneva, Geneva, Switzerland.

Ohm, J.-R., Bunjamin, F., Liebsch, W., Makai, B., Müller, K., Smolic, A. and Zier, D. (2000). A set of visual feature descriptors and their combination in a low-level description scheme, *Signal Processing: Image Communication* **16**(1): 157–179.

Oja, E. and Yuan, Z. (2006). The FastICA algorithm revisited: Convergence analysis, *IEEE Transactions on Neural Networks.* **17**(6): 1370–1381.

Ojala, T., Maenpaa, T., Pietikainen, M., Viertola, J., Kyllonen, J. and Huovinen, S. (2002). Outex-new framework for empirical evaluation of texture analysis algorithms, *Proceedings of the 16th International Conference onPattern Recognition, 2002.*, Vol. 1, IEEE, pp. 701–706.

Ojala, T. and Pietikainen, M. (1998). Nonparametric multichannel texture description with simple spatial operators, *Proceedings of the Fourteenth International Conference on Pattern Recognition, 1998.*, Vol. 2, IEEE, pp. 1052–1056.

Ojala, T., Pietikäinen, M. and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions, *Pattern recognition* **29**(1): 51–59.

Ojala, T., Pietikainen, M. and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(7): 971–987.

Ortega, M., Rui, Y., Chakrabarti, K., Mehrotra, S. and Huang, T. S. (1997). Supporting similarity queries in MARS, *Proceedings of The Fifth ACM International Multimedia Conference (ACM Multimedia 97)*, Seattle, WA, USA, pp. 403–413.
URL: *http://www.research.microsoft.com/ yongrui/ps/acm97.ps.Z*

Paisitkriangkrai, S., Shen, C. and Zhang, J. (2008). Fast pedestrian detection using a cascade of boosted covariance features, *IEEE Transactions on Circuits and Systems for Video Technology.* **18**(8): 1140–1151.

Park, S. B., Lee, J. W. and Kim, S. K. (2004). Content-based image classification using a neural network, *Pattern Recognition Letters* **25**(3): 287–300.

Patil, S. and Talbar, S. (2012). Content based image retrieval using various distance metrics, *Data Engineering and Management*, Springer, pp. 154–161.

Pavlidis, T. (1982). *Algorithms for graphics and image processing*, Computer science press.

Petrou, M. and García-Sevilla, P. (2006). *Image Processing: Dealing with Texture*, John Wiley & Sons.

Picard, R. W., Kabir, T. and Liu, F. (1993). Real-time recognition with the entire brodatz texture database, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1993.*, IEEE, pp. 638–639.

Pinto, N., Barhomi, Y., Cox, D. D. and DiCarlo, J. J. (2011). Comparing state-of-the-art visual features on invariant object recognition tasks, *IEEE Workshop on Applications of Computer Vision, 2011*, IEEE, pp. 463–470.

Prasad, B. and Krishna, A. (2011). Statistical texture feature-based retrieval and performance evaluation of CT brain images, *Proceedings of the 3rd International Conference on Electronics Computer Technology, 2011.*, Vol. 2, IEEE, pp. 289–293.

Rao, A. R. and Lohse, G. L. (1992). Identifying high-level features of texture perception, *SPIE/IS&T 1992 Symposium on Electronic Imaging: Science and Technology*, International Society for Optics and Photonics, pp. 424–435.

Rho, S. and Yeo, S.-S. (2013). Bridging the semantic gap in multimedia emotion/mood recognition for ubiquitous computing environment, *The Journal of Supercomputing* pp. 1–13.

Safar, M., Shahabi, C. and Sun, X. (2000). Image retrieval by shape: a comparative study, *Proceedings of the IEEE International Conference on Multimedia and Expo, 2000.*, Vol. 1, IEEE, pp. 141–144.

Salembier, P., Sikora, T. and Manjunath, B. (2002). *Introduction to MPEG-7: multimedia content description interface*, John Wiley & Sons, Inc.

Salton, G. (1971). The smart retrieval system experiments in automatic document processing.

Salton, G. (1992). The state of retrieval system evaluation, *Information processing & management* **28**(4): 441–449.

Sanchez-Marin, F. J. (2000). Automatic recognition of biological shapes with and without representations of shape., *Artificial Intelligence in Medicine* **18**(2): 173.

Saracevic, T. (1995). Evaluation of evaluation in information retrieval, *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM, pp. 138–146.

Sayadi, M., Sakrani, S., Fnaiech, F. and Cheriet, M. (2008). Gray-level texture characterization based on a new adaptive nonlinear auto-regressive filter, *Electronic Letters on Computer Vision and Image Analysis* **7**(1): 40–53.

Schmid, C. (2001). Constructing models for content-based image retrieval, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.*, Vol. 2, IEEE, pp. II–39.

Schober, J.-P., Hermes, T. and Herzog, O. (2005). Picturefinder: Description logics for semantic image retrieval, *Proceedings of the IEEE International Conference on Multimedia and Expo, 2005.*, IEEE, pp. 1571–1574.

Schwarz, M. W., Cowan, W. B. and Beatty, J. C. (1987). An experimental comparison of RGB, YIQ, LAB, HSV, and opponent color models, *ACM Transactions on Graphics (TOG)* **6**(2): 123–158.

Sclaroff, S., Taycher, L. and La Cascia, M. (1997). ImageRover: a content-based browser for the world wide web, *IEEE Workshop on Content-Based Access of Image and Video Libraries*, San Juan, Puerto Rico, pp. 2–9.

Shapiro, L. G., Stockman, G. C., Shapiro, L. G. and Stockman, G. (2001). *Computer Vision*, Prentice Hall.

Shyu, C.-R., Kak, A., Brodley, C. E. and Broderick, L. S. (1999). Testing for human perceptual categories in a physician-in-the-loop cbir system for medical imagery, *Proceedings of the IEEE Workshop on Content-Based Access of Image and Video Libraries, 1999.*, IEEE, pp. 102–108.

Smith, A. R. (1978). Color gamut transform pairs, *SIGGRAPH Computer Graphics* **12**(3): 12–19.

Smith, J. R. and Chang, S.-F. (1996). Tools and techniques for color image retrieval, *IS&T/SPIE Symposium on Electronic Imaging: Science and Technology (EI'96) - Storage and Retrieval for Image and Video Databases IV*, Vol. 2670, San Jose, CA.

Smith, J. R. and Chang, S.-F. (1997). Querying by color regions using the *VisualSEEk* content-based visual query system, *in* M. T. Maybury (ed.), *Proceedings of the IJCAI Workshop on Intelligent Multimedia Information Retrieval.*
URL: *ftp://ftp.ctr.columbia.edu/CTR-Research/advent/public/papers/96/smith96d.ps.gz*

Squire, D. M., Müller, W., Müller, H. and Raki, J. (1999). Content-based query of image databases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback, *Pattern Recognition Letters*, pp. 143–149.

Squire, D. and Pun, T. (1997). A comparison of human and machine assessments of image similarity for the organization of image databases, *Proceedings of the Scandinavian conference on image analysis*, Vol. 1, PROCEEDINGS PUBLISHED BY VARIOUS PUBLISHERS, pp. 51–58.

Stitou, Y., Lasmar, N. and Berthoumieu, Y. (2009). Copulas based multivariate gamma modeling for texture classification, *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.*, IEEE, pp. 1045–1048.

Stricker, M. A. and Orengo, M. (1995). Similarity of color images, *IS&T/SPIE's Symposium on Electronic Imaging: Science & Technology*, Vol. 2420, International Society for Optics and Photonics, pp. 381–392.

Sun, G., Liu, J., Sun, J. and Ba, S. (2006). Locally salient feature extraction using ica for content-based face image retrieval, *Proceedings of the First International Conference onInnovative Computing, Information and Control, 2006.*, Vol. 1, IEEE, pp. 644–647.

Swain, M. J. and Ballard, D. H. (1990). Indexing via color histograms, *Proceedings of the DARPA Image Understanding Workshop*, Pittsburgh, PA, USA, pp. 623–630.

Tamura, H., Mori, S. and Yamawaki, T. (1978). Textural features corresponding to visual perception, *IEEE Transactions on Systems, Man and Cybernetics.* **8**(6): 460–473.

Tang, J., Zha, Z.-J., Tao, D. and Chua, T.-S. (2012). Semantic-gap-oriented active learning for multilabel image annotation, *IEEE Transactions on Image Processing.* **21**(4): 2354–2360.

Teague, M. R. (1980). Image analysis via the general theory of moments, *J. Opt. Soc. Am* **70**(8): 920–930.

ten Brinke, W., Squire, D. M. and Bigelow, J. (2006). The meaning of an image in content-based image retrieval, *in* T. Latour and M. Petit (eds), *2nd International Workshop on Philosophical Foundations of Information Systems Engineering (PHISE 2006), in conjunction with the 18th International Conference on Advanced Information Systems Engineering (CAiSE'06)*, Luxembourg, pp. 710–719.

Tieng, Q. M. and Boles, W. (1997). Recognition of 2D object contours using the wavelet transform zero-crossing representation, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(8): 910–916.

Trojan, N. (2004). *CBIR-based dermatology diagnostic assistant*, Honours thesis, School of Computer Science and Software Engineering, Monash University, Clayton, Victoria, Australia. Supervised by Dr. David Squire.

Tüceryan, M. and Jain, A. (1990). Texture segmentation using voronoi polygons, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(2): 211–216.

Tuceryan, M. and Jain, A. K. (1998). *Texture Analysis*, World Scientific Publishing Co., chapter 2.1, pp. 207–248.
URL: *http://www.cs.iupui.edu/ tuceryan/research/ComputerVision/texture-review.pdf*

van Hateren, J. H. and van der Schaaf, A. (1998). Independent component filters of natural images compared with simple cells in primary visual cortex, **265**(1394): 359–366.

Varma, M. and Zisserman, A. (2005). A statistical approach to texture classification from single images, *International Journal of Computer Vision* **62**(1-2): 61–81.

Varma, M. and Zisserman, A. (2009). A statistical approach to material classification using image patch exemplars, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* **31**(11): 2032–2047.

Vassilieva, N. S. (2009). Content-based image retrieval methods, *Programming and Computer Software* **35**(3): 158–180.

Veltkamp, R. C., Burkhardt, H. and Kriegel, H.-P. (eds) (2001). *State-of-the-Art in Content-Based Image and Video Retrieval [Dagstuhl Seminar, 5-10 December 1999]*, Kluwer.

Vendrig, J., Worring, M. and Smeulders, A. W. (1999). Filter image browsing, *Visual Information and Information Systems*, Springer, pp. 147–155.

Verma, A., Banerji, S. and Liu, C. (2010). A new color SIFT descriptor and methods for image category classification, *International Congress on Computer Applications and Computational Science, Singapore, December*, pp. 4–6.

Vimina, E. and Jacob, K. P. (2012). Image retrieval using local colour and texture features, *Mechanical Engineering and Technology*, Springer, pp. 767–772.

Voorhees, H. and Poggio, T. (1987). Detecting blobs as textons in natural images, *Image Understanding Workshop*, Vol. 2, DARPA, pp. 892–899.

Wang, F. and Dai, Q. (2007). A new multi-view learning algorithm based on ICA feature for image retrieval, *MMM (1)*, pp. 450–461.

Wang, J. Z., Li, J. and Wiederhold, G. (2001). SIMPLIcity: Semantics-sensitive integrated matching for picture libraries, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* **23**(9): 947–963.

Wangming, X., Jin, W., Xinhai, L., Lei, Z. and Gang, S. (2008). Application of image SIFT features to the context of CBIR, *Proceedings of the International Conference on Computer Science and Software Engineering, 2008.*, Vol. 4, IEEE, pp. 552–555.

Witkin, A. P. (1983). Scale-space filtering, *Proceedings of the Eighth international joint conference on Artificial intelligence - Volume 2*, IJCAI'83, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 1019–1022.
URL: *http://dl.acm.org/citation.cfm?id=1623516.1623607*

Wu, B. and Nevatia, R. (2008). Optimizing discrimination-efficiency tradeoff in integrating heterogeneous local features for object detection, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2008.*, IEEE, pp. 1–8.

Wu, M., Zhou, J. and Sun, J. (2012). Multi-scale ica texture pattern for gender recognition, *Electronics letters* **48**(11): 629–631.

Wu, X. and Liu, G. (2007). Application of independent component analysis to dynamic contrast-enhanced imaging for assessment of cerebral blood perfusion, *Medical image analysis* **11**(3): 254–265.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A. and Torralba, A. (2010). Sun database: Large-scale scene recognition from abbey to zoo, *Proceedings of the IEEE conference on Computer vision and pattern recognition, 2010.*, IEEE, pp. 3485–3492.

Xu, C. L. and Zhen, X. T. (2009). Chromatic statistical landscape features for retrieval of color textured images, *Proceedings of the Fourth International Conference on Internet Computing for Science and Engineering, 2009*, IEEE, pp. 98–101.

Xu, K., Georgescu, B., Comaniciu, D. and Meer, P. (2000). Performance analysis in content-based retrieval with textures, *Proceedings of the 15th International Conference on Pattern Recognition, 2000.*, Vol. 4, IEEE, pp. 275–278.

Xue, Z., Long, L. R., Antani, S., Jeronimo, J. and Thoma, G. R. (2008). A web-accessible content-based cervicographic image retrieval system, *Medical Imaging*, International Society for Optics and Photonics, pp. 691907–691907.

Yang, H. S., Lee, S. U. and Lee, K. M. (1998). Recognition of 2D object contours using starting-point-independent wavelet coefficient matching, *Journal of Visual Communication and Image Representation* **9**(2): 171–181.

Yang, J., Jiang, Y.-G., Hauptmann, A. G. and Ngo, C.-W. (2007). Evaluating bag-of-visual-words representations in scene classification, *Proceedings of the international workshop on Workshop on multimedia information retrieval*, ACM, pp. 197–206.

Yarygina, A., Novikov, B. and Vassilieva, N. (2011). Processing complex similarity queries: A systematic approach., *ADBIS (2)*, pp. 212–221.

Young, I. T., Walker, J. E. and Bowie, J. E. (1974). An analysis technique for biological shape. i, *Information and control* **25**(4): 357–370.

Zhang, D., Islam, M. M. and Lu, G. (2012). A review on automatic image annotation techniques, *Pattern Recognition* **45**(1): 346–362.

Zhang, D. and Lu, G. (2002). Generic Fourier descriptor for shape-based image retrieval, *Proceedings of the IEEE International Conference on Multimedia and Expo, 2002. ICME'02.*, Vol. 1, IEEE, pp. 425–428.

Zhang, D. and Lu, G. (2004). Review of shape representation and description techniques, *Pattern recognition* **37**(1): 1–19.

Zhang, D., Wong, A., Indrawan, M. and Lu, G. (2000). Content-based image retrieval using Gabor texture features, *Proceedings of the IEEE Pacific-Rim Conference on Multimedia, 2000*, University of Sydney, Australia.

Zhang, J., Zhao, H. and Liang, J. (2012). Continuous rotation invariant local descriptors for texton dictionary-based texture classification, *Computer Vision and Image Understanding* .

Zhang, R., Zhang, Z., Li, M., Ma, W.-Y. and Zhang, H.-J. (2005). A probabilistic semantic model for image annotation and multimodal image retrieval, *Proceedings of the Tenth IEEE International Conference on Computer Vision, 2005.*, Vol. 1, IEEE, pp. 846–851.

Zhu, Q., Yeh, M.-C., Cheng, K.-T. and Avidan, S. (2006). Fast human detection using a cascade of histograms of oriented gradients, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2006.*, Vol. 2, IEEE, pp. 1491–1498.