Master's thesis

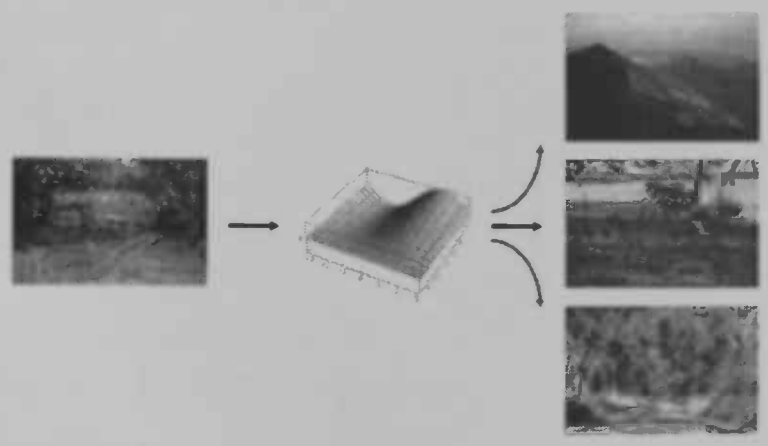
Spatial Pattern Spectra and Content-based Image Retrieval

Arjan Gijsenij

Institute for Mathematics and Computing Science
University of Groningen
The Netherlands

Supervisor: M.H.F. Wilkinson

1st June 2005



Contents

Abstract

Granulometries are powerful and versatile tools in image analysis and pattern spectra, or size distributions, are a simple method of extracting information from an image using these granulometries. One of the drawbacks of the traditional pattern spectra, the lack of spatial information about connected components within images, is addressed in this project by introducing three extensions to the regular area pattern spectra: one based on moments, one based on translation of the components within the image, and one based on multi-scale connectivity. These three extensions are tested in the field of content-based image retrieval: are they able to retrieved images from an image-database, that are similar in some way to a certain, user-provided, query-image? This is a question that is interesting for fields like intelligent multimedia and web searches (search engines).

Chapter 1

Introduction

In image analysis, granulometries are powerful tools for studying how the image content changes as a function of the filter's size parameter. Insight can be gained into the distribution of the details over different size classes, often without prior segmentation of the image. Granulometries were first introduced by Matheron [18], a recent review is given in [41].

Pattern spectra are the simplest way of extracting information out of images using granulometries. They were first introduced by Maragos [17], and show the sum of the gray-levels over the image changes as a function of the size parameter.

On of the drawbacks of these pattern spectra is the lack of spatial information; information about the position of certain components is not included into the classical definition of the pattern spectra. In [3] and [43], this problem is addressed by introducing spatial size distributions and generalised pattern spectra, respectively. In [6], a third method is introduced based on multi-scale connectivity.

The main focus of this thesis will be on these three extensions of the area pattern spectra. They will be implemented, and compared to the area pattern spectra. The comparison will be done in the field of *Content-Based Image Retrieval* (CBIR). CBIR is the task to find images, which are similar in some way to a certain query-image.

CBIR is especially interesting for intelligent multimedia and web searches. It has been an active field of research for decades, but is progressing rapidly because of the tremendous number of digital images, as a result of the rise of the interned. Ever since the introduction of digital images, there exists a gap between low level features of images and high level semantic concepts. Therefore, a large part of the research in this field is focussing on reducing, or even bridging, this gap. These three extensions of the area pattern spectra focus on the same goal.

In Smeulders et al. [30], Goodrum [13] and Aigrain et al. [1], an overview is given about content-based image retrieval. At this moment, several systems exist that provide the possibility of searching a database with images, looking for a certain query-image, including QBIC [9], WebSeek [31] and WebSeer [11]. None of these systems, however, are really satisfactory.

In this thesis, some definitions about images and morphology are given first. After that, granulometries and pattern spectra are introduced. In chapters 3, 4 and 5, the three extensions to the area pattern spectra are discussed, followed by experiments to test these three methods. Finally, in chapter 7, the conclusions are presented.

Chapter 2

Definitions

In this section, some basic definitions will be given, which are used throughout this thesis. First, the basics around images are described, followed by a few basic morphological image operators. After that, granulometries and pattern spectra are introduced. These concepts are the foundations of the feature extraction methods that are described later on in this thesis.

2.1 Images

The image domain $\mathbf{M}$ is defined as a subset of the discrete n-space or the Euclidean n-space: $\mathbf{M} \subset \mathbb{Z}^n$ or $\mathbb{R}^n$ (usually, $n = 2$), respectively. Binary images X and Y are defined as subsets of $\mathbf{M}$, while gray-scale images images f and g are mappings from $\mathbf{M}$ to $\mathbb{Z}$ or $\mathbb{R}$.

Image operators are given as functions; functions defined for binary images are usually denoted by upper case letters, while the same function for gray-scale images is denoted by the corresponding lower case letter. When a binary function is given, usually the gray-scale equivalent can be determined by *threshold decomposition* [39] or *stacking* [14]. If the binary operator is given by $\Gamma_a^\psi(X)$, then the gray-scale equivalent is given by:

$$\gamma_a^\psi(f)(x) = \max\{h | x \in \Gamma_a^\psi(T_h(f))\}, \tag{2.1}$$

where $T_h(f)$ is the result of thresholding f at gray level h:

$$T_h(f) = \{x \in \mathbf{M} | f(x) \geq h\}. \tag{2.2}$$

In several algorithms that are used, an image is divided into three components:

- *level component* or *flat zone* L_h - a connected component of the set of pixels $\{p \in \mathbf{M} | f(p) = h\}$, e.g. a set of connected pixels, all with gray-value *equal* to h.
- *regional maximum* M_h - a level component no members of which have neighbours larger than h.
- *peak component* P_h - a connected component of $T_h(f)$, e.g. a connected component with the gray-value of all pixels larger than or equal to h.

Note that any regional maximum is also a peak component, but the reverse is not always true.

2.2 Openings

The basic morphological operations are erosion (denoted by the $\ominus$-operator) and dilation (denoted by the $\oplus$- operator), and they are defined as follows for binary images ([28]):

$$X \ominus Y = \{a : Y + a \subseteq X\} = \bigcup_{p \in Y} X - p, \tag{2.3}$$

$$X \oplus Y = \{a + b : a \in X, b \in Y\} = \bigcup_{p \in Y} X + p. \tag{2.4}$$

For gray-scale images and structuring elements these operators are defined as in [34]:

$$(f \ominus g)(x, y) = \min_{(i,j)}\{f(x + i, y + j) - g(i, j)\}, \tag{2.5}$$

$$(f \oplus g)(x, y) = \max_{(i,j)}\{f(x - i, y - j) + g(i, j)\}. \tag{2.6}$$

In general, an structural opening is defined by erosion, followed by a dilation. A multi-scale opening (in the binary case) of X by B at scale $n = 0, 1, 2...$ is defined by n consecutive erosions, followed by n consecutive dilations of X by B, where B is used as structuring element. The following definition applies to the binary case, but the gray-scale equivalent can easily be derived:

$$X \circ nB = (\underbrace{(X \ominus B) \ominus B ... \ominus B}_{n\ times})\underbrace{\oplus B \oplus B ... \oplus B}_{n\ times} \tag{2.7}$$

2.3 Granulometries

A very useful tool in morphological image analysis are granulometries. They were first described by Matheron in [18]. Granulometries involve sequences of openings or closings with structuring elements of increasing size, removing image details below a certain size.

A family $\Psi = (\psi_\lambda)_{\lambda \geq 0}$ is called a granulometry, when the following three properties are satisfied.

$$\forall \lambda \geq 0, \quad \psi_\lambda \textit{ is increasing}, \tag{2.8}$$

$$\forall \lambda \geq 0, \quad \psi_\lambda \textit{ is anti-extensive}, \tag{2.9}$$

$$\forall \lambda \geq 0, \mu \geq 0, \quad \psi_\lambda \psi_\mu = \psi_\mu \psi_\lambda = \psi_{\max(\lambda,\mu)} \tag{2.10}$$

A function ψ is said to be increasing, if $X \leq Y \Rightarrow \psi(X) \leq \psi(Y)$, and a function ψ is said to be anti-extensive, if $X \leq \psi(X)$. Furthermore, the last property (equation 2.10) implies that for every $\lambda \geq 0$, ψ_λ is an idempotent transformation ($\psi_\lambda \psi_\lambda = \psi_\lambda$). From these properties, it can be seen that all ψ_λ are openings.

Everything that was said about granulometries so far applies to binary as well as gray-scale images, in any domain. In the context of digital images (2-D discrete images), a granulometry is a sequence of openings γ_n indexed on an integer $n \geq 0$, and the result of each opening in this sequence is by definition smaller than or equal to the previous one. This translates to

$$\forall X \subset \mathbb{Z}, \ \forall n \geq m \geq 0, \ \gamma_n(X) \subseteq \gamma_m(X) \tag{2.11}$$

and

$$\forall f, \ \forall n \geq m \geq 0, \ \gamma_n(f) \leq \gamma_m(f) \tag{2.12}$$

for binary and gray-scale images respectively. A comparison between two gray-scale images is done by pixel-wise comparing each pixel of both images ($\gamma_n(f) \leq \gamma_m(f)$ means that for any pixel p in domain $\mathbf{M}_f$, $\gamma_n(f)(p) \leq \gamma_m(f)(p)$).

2.4 Pattern spectra

A pattern spectrum ([17]) can detect critical scales in an image object and quantify various aspects of its shape-size content, by studying how the image content changes (i.e. the number of foreground pixels) as a function of the filter's size parameter.

The pattern spectrum conveys four (among others) useful types of information about an image:

1. The boundary roughness of the image relative to the structuring element manifests itself as contributions in the lower part of the pattern spectrum.
2. The existence of long capes or bulky protruding parts in the image that consists of patterns like the structuring element shows up as isolated impulses in the pattern spectrum around the positive value of the scale of the structuring element.
3. The maximal degree that the image contains the structuring element, or shapes like the structuring element, can be measured by dividing the pattern spectrum of the image with respect to the structuring element by the area of the image.
4. The negative size portion in the pattern spectrum is useful because big impulses at negative sizes illustrate the existence of prominent intruding gulfs or holes in the image. Although this is something that is not used throughout this project, it is something to look into in the future, since it might provide extra information about the images, that can be obtained in a simple way.

The pattern spectrum of a binary image $X \subseteq \mathbb{R}^2$ relative to a convex binary pattern $B \subseteq \mathbb{R}^2$ as the (differential size distribution) function:

$$PS_X(\lambda) = \frac{-dA(\psi_\lambda)}{d\lambda}, \tag{2.13}$$

where $A(\psi)$ is defined as the area of ψ. Usually, the size parameter λ, which defines the scale, will be ≥ 0. In those cases, the standard opening is used as ψ_λ. By using closings, the scale can be extended to negative sizes λ ($\lambda < 0$).

In order to efficiently use arbitrary structuring elements (denoted by B for binary images, and g for gray-scale), we extend the pattern spectrum ideas to discrete space binary images here (B and g are the structuring elements that are used to compute the granulometries γ_n and γ_{n+1}, and are omitted from these equations):

$$PS_X(n) = A(\gamma_n \setminus \gamma_{n+1}), \tag{2.14}$$

where $S \setminus Q$ denotes set difference; this can be rewritten to

$$PS_X(n) = A(\gamma_n) - A(\gamma_{n+1}). \tag{2.15}$$

For gray-scale images, the discrete function is

$$PS_f(n) = A(\gamma_n) - A(\gamma_{n+1}). \tag{2.16}$$

In the gray-scale case, $A(\gamma)$ means the sum of the gray values in the resulting image:

$$PS_f(n) = \sum_{p \in M} ((\gamma_n(f)(p)) - (\gamma_{n+1}(f)(p))) \tag{2.17}$$

Hence, the discrete pattern spectrum can be obtained via a forward area difference. The B-shapiness of X can be measured by $PS_X(n) \setminus A(X)$, where B is the structuring element that is used to compute the pattern spectrum PS.

Chapter 3

General pattern spectra

Since none of the standard pattern spectra contain any information on the spatial distribution, Wilkinson [42] proposed a parameterisation of this distribution. Although spatial information is not always needed (in general, this is the case when a random distribution of details is analysed), for some problems, retaining the spatial information *is* essential, for instance when the distribution of certain components is sought. The generalised pattern spectra that are introduced by Wilkinson [42] stores information about the size of the components, *as well as* the spatial distribution of these components.

In this section, the general idea behind the generalised pattern spectra is explained first. After that, the implementation of standard pattern spectra and the generalised pattern spectra is presented. Finally, some images are shown along with the extracted features using the presented method.

3.1 Parameterisation of the spatial distribution

The *generalised pattern spectrum* $GPS_{X,M}$ is defined for binary images X and some parameterisation $M(X)$ of the spatial distribution of detail in the image X as follows:

$$GPS_{X,M}(n) = M(\gamma_n \setminus \gamma_{n+1}), \tag{3.1}$$

An obvious choice for this parameterisation is through the use of moments:

$$GPS_{X,M}(n) = m_{pq}(\gamma_n \setminus \gamma_{n+1}), \tag{3.2}$$

where

$$m_{pq}(X) = \sum_{(x,y)\in X} x^p y^q \tag{3.3}$$

is the moment of order $(p+q)$, where p and q are both natural numbers, of image X (in the 2-D case of binary images). This parameterisation is extended to the case of gray-scale images by taking the gray-value of the pixels into account. The moment m_{pq} is defined in the 2-D case of gray-scale images as:

$$m_{pq}(f) = \sum_{(x,y)\in f} x^p y^q f(x,y). \tag{3.4}$$

Rotation, translation and scale invariance of the generalised pattern spectrum is ensured by using some combinations of instances of the method, as explained in the next section.

3.2 Moment invariants

To characterise images, features have to be derived, which is done using *moment invariants*. Moment invariants can be calculated through *geometric moments* m_{pq}, see equation (3.4). Equation (3.4) is in practice uniquely determined by the image function $f(x, y)$ (e.g. the image function $f(x, y)$ is uniquely determined by an infinite sequence of moments), which is stated in the *uniqueness theorem* [15]:

Theorem 3.2.1 (Uniqueness theorem). *If $f(x, y)$ is piecewise continuous and has nonzero values only in a finite part of the (x, y)-plane, then $f(x, y) \Leftrightarrow m_{pq}$ for $p, q = 0, 1, 2, \cdots, \infty$.*

In theory, this only holds for continuous image functions (and the corresponding continuous geometric moments), and an infinite number of moments is required, but in practice, it is also used with discrete image functions and it turns out that gross image shape is represented by a finite number of lower-order moments.

3.2.1 Hu's invariants

The mathematical foundation for two-dimensional moment invariants were first set out by Hu [15]. Hu defines seven descriptor values, that are independent to object translation, scale and orientation, e.g. they are translation, scale and rotation invariant.

Translation invariance is achieved by computing moments that are normalised with respect to the centre of gravity so that the centre of mass of the distribution is at the origin. These moments are called *central moments*. The centre of gravity are calculated using the geometric moments of equation 3.4:

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}}. \tag{3.5}$$

The *central moments* μ_{pq} can be calculated using these image centroids:

$$\mu_{pq} = \sum_{(x,y)} (x - \bar{x})^p (y - \bar{y})^q f(x, y). \tag{3.6}$$

Furthermore, these central moments can be normalised for the effects of change of scale. The *normalised central moments* η_{pq} can be calculated as follows:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\alpha}}, \tag{3.7}$$

where

$$\alpha = \frac{p + q}{2} + 1. \tag{3.8}$$

for $p + q = 2, 3, \ldots$.

From the second and third order moments, a set of seven invariant moments can be derived, called

Hu's moment invariants, and are given by:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} && (3.9)\\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 && (3.10)\\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 && (3.11)\\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 && (3.12)\\
\phi_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) + && (3.13)\\
&\quad (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) && \\
\phi_6 &= (\eta_{20} - \eta_{02})((\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) + && (3.14)\\
&\quad 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) && \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})((\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2) + && (3.15)\\
&\quad (3\eta_{12} - \eta_{30})(\eta_{21} + \eta_{03})(3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2) &&
\end{aligned}$$

It has been shown, that this combination of moment invariants is invariant to translation and scaling, as well as *rotation*.

3.2.2 Complex moments

Another scheme to derive moment invariants is set out in [10], and is based on *complex moments*. They are expressed in terms of geometric moments (equation 3.4):

$$c_{pq} = \sum_{k=0}^{p}\sum_{j=0}^{q}\binom{p}{k}\binom{q}{j}(-1)^{q-j} \cdot i^{p+q-k-j} \cdot m_{k+j,p+q-k-j}, \tag{3.16}$$

where $i^2 = -1$.

It was proven by Flusser and Suk [10] that an infinite number of invariants for any order of moments can be derived from this definition of complex moment. Only a few, however, are mutually independent. The term *basis* is defined to be the smallest set by means of which all other invariants can be expressed, i.e. the set must be *independent* (none of its elements can be expressed as a function of the other elements) and *complete* (any invariant can be expressed by means of the basis elements).
The basis composed from 2nd and 3rd order moments can be defined as follows:

$$\begin{aligned}
\Phi(1,1) &= c_{11}, && (3.17)\\
\Phi(2,1) &= c_{21}c_{12}, && (3.18)\\
\Phi(2,0) &= c_{20}c_{12}^2, && (3.19)\\
\Phi(3,0) &= c_{30}c_{12}^3. && (3.20)
\end{aligned}$$

The basis elements $\Phi(2,0)$ and $\Phi(3,0)$ consist of a real part and an imaginary part, which can be addressed as $\mathrm{Re}(\Phi(2,0))$, $\mathrm{Im}(\Phi(2,0))$, $\mathrm{Re}(\Phi(3,0))$ and $\mathrm{Im}(\Phi(3,0))$, respectively.

In [10], it is also shown that Hu's moment invariants are dependent and incomplete. The dependency is demonstrated by rewriting the third moment invariant of Hu:

$$\phi_3 = \frac{\phi_5^2 + \phi_7^2}{\phi_4^3}. \tag{3.21}$$

This means that ϕ_3 can be excluded from Hu's moment invariants, since it can be calculated using ϕ_4, ϕ_5 ϕ_7.

The incompleteness of Hu's moment invariants follows from the fact, that when trying to recover the geometric moments from the moment invariants $\phi_1, ..., \phi_7$, the sign of the geometric moment m_{11} cannot be determined.

3.3 Implementation

Since the standard pattern spectrum is obtained by taking $p = 0$ and $q = 0$, i.e. m_{00}, the computation of the generalised pattern spectra can be performed by modification of any existing algorithm for area pattern spectra. In [19] the area pattern spectrum is implemented. This implementation was adopted to fit the needs for spatial pattern spectra, using the method of keeping auxiliary data from [43] to store all the data needed to compute the spatial pattern spectra.

Area pattern spectra use the area opening in stead of the classical opening, which is derived from attribute openings (see [7] for thorough discussion on attribute operators). Area openings are discussed in more detail in [40], but are briefly given here. After that, the implementation of the spatial pattern spectrum is described, using the existing implementation of the area pattern spectrum.

3.3.1 Area openings

An area opening (binary as well as gray-scale) is based on a binary connected opening $\Gamma_x(X)$, which extracts the connected component to which x belongs, discarding all others:

$$\Gamma_x(X) = \begin{cases} \text{the connected component containing } x & if\ x \in X \\ \emptyset & otherwise \end{cases} \tag{3.22}$$

The binary area opening of X with scale parameter λ is given by

$$\Gamma^a_\lambda(X) = \{x \in X | A(\Gamma_x(X)) \geq \lambda\}. \tag{3.23}$$

The gray-scale area opening is obtained by assigning each point of the image the highest threshold at which it still belongs to a connected component of area λ or larger:

$$(\gamma^a_\lambda(f))(x) = \max\{h | x \in \Gamma^a_\lambda(T_h(f))\}, \tag{3.24}$$

where $T_h(f)$ is the thresholded image of f and threshold h, as in equation (2.2).

3.3.2 Algorithm

The main idea of the implementation of the area pattern spectrum is to process the image in gray-scale order. In the process, peak components are filled, starting at the regional maxima. A new peak component $P_{h'}$, at gray level $h' > h$ which must be merged with some level component L_h, may be found each time a new gray level is inspected. When this occurs, it means that an area opening with area-parameter $r = A(P_{h'})$ (i.e. the area of the peak component) would remove this component from the sum of gray levels at that point in the image. In this case, the element, identifying the area of the peak component in the array, containing the spectrum to be calculated, should be incremented by $(h' - h)A(P_{h'})$, since the peak component will be assigned the gray-value of the level component it is merged with. This element is found using B, which is a mapping from Λ to $[0 \ldots N_s]$ and N_s being the number of scales that are used for the pattern spectra.

Algorithm 1 Calculation of area pattern spectrum

```
Require: array S contains the sorted pixel list
  InitSpectrum();
  for p = 0 to Length(S) do
    pix = S[p];   {pix is the current pixel to be processed}
    MakeSet(pix);
    for all neighbours nb of pix do
      if ( (f[pix] < f[nb]) ∨
           (f[pix] = f[nb]  ∧ nb < pix) ) then
        Union(nb, pix);
      end if
    end for
  end for
```

3.3.3 Union-find algorithm

An efficient way to keep track of the peak components is needed. Since these are disjoint sets, we can use the union-find algorithm developed by Tarjan [35]. This method provides a general method for efficiently keeping track of disjoint sets.

Tarjan's union-find method uses tree structures to represent sets. All pixels of one component will end up in the same tree (e.g. they are nodes in the same tree), and the parent of all pixels point to the same root pixel. The root of this tree is flagged, by setting its parent to a value equal to -1 times the area of the whole component that is represented by the tree. Hence, a pixel with a negative value as its parent is a root pixel.

In the implementation of the spatial pattern spectra, there are three important functions:

- Makeset(x): create a new singleton $\{x\}$, see algorithm 2.
- Findroot(x:) return the root element of the set containing x, see algorithm 3.
- Union(x, y): compute the union of the two sets containing pixels x and y, see algorithm 4.

Each pixel contains a pointer to its parent pixel. All the trees are stored in an array of the same size as the image. The index of each pixel in this array is `width*y+x`, where `width` is equal to the width of the image and `x` and `y` are the x and y coordinates of the image. In these functions, `spec` is an array that holds the data of the computed pattern spectrum, and B is a mapping from the set of values the attributes of the peak components can assume, to the index range of the pattern spectrum.

When looking back to the previous section, explaining the algorithm to compute the area pattern spectrum, it has to be ensured that the gray value of the root element of a peak component P_H is h. This is done by setting the last pixel that was processed to be the root of the new tree. To this end, all the pixels in the image are sorted by their gray-value; pixels of the same gray level are processed in scan line order. The pseudo code for this algorithm is shown in algorithm 1.

3.3.4 Auxiliary data

After the computation of the area pattern spectrum, which is discussed first by Meijster and Wilkinson [19], (moment) pattern spectra are to be calculated. To do this, I extended this implementation of the

Algorithm 2 The function MakeSet, introduced in section 3.3.3

```
MakeSet(int x)
    parent[x] = -1;
end
```

Algorithm 3 The function FindRoot, introduced in section 3.3.3

```
FindRoot(int x)
    if (parent[x] ≥ 0) then
        parent[x] = FindRoot(parent[x]);
        return parent[x];
    else
        return x;
    end if
end
```

area pattern spectrum in such a way that information about each component, which is necessary for the computation of the moment pattern spectra, is gathered during the opening transform. This information is called *auxiliary data* [43]. Auxiliary data is stored in an array `auxdata`, which consists of N void pointers, to store pointers to any auxiliary data about the peak component (N being the total number of pixel in the image).

The auxiliary data is stored in a record, and it contains a number of values, depending on the attribute used. The `area` and `difference` are stored as integers, and represent the area of the current level component and the difference in gray-value with the next level component, respectively. Furthermore, a number of sums are stored (for instance, the sum of x, the sum of y and the sum of $x \cdot y$), to be able to compute *moments* in post-processing.

Four functions are used to store the auxiliary data, and keep it up-to-date:

- `EmptyAuxData`: initialise an empty structure.
- `DisposeAuxData`: discard the the structure with the auxiliary data.
- `MergeAuxData`: merge two sets of auxiliary data.

Algorithm 4 The *original* function Union, introduced in section 3.3.3

```
Union (int n, int p)
    int r = FindRoot(n);
    if r ≠ p then
        if f[r] ≠ f[p] then
            spec[B(-parent[r])] = spec[B(-parent[r])] - (f[r] - f[p]) × parent[r];
        end if
        Link (r, p);
    end if
end
```

Algorithm 5 The function Link, used by both versions of Union (algorithms 4 and 6)

```
Link (int x, int y)
   parent[y] = parent[y] + parent[x];
   parent[x] = y;
end if
```

Algorithm 6 The *modified version* of the function Union, introduced in section 3.3.4

```
Union (int n, int p)
   int r = FindRoot(n);
   if r ≠ p then
      if f[r] ≠ f[p] then
         spec[B(−parent[r])] = MergeAuxData (auxdata[r],
                                             auxdata[B(−parent[r])]−1],
                                             im[r]−im[pixel]);
      end if
      auxdata[pixel] = MergeAuxData (auxdata[r],
                                     auxdata[pixel],
                                     im[r]−im[pixel]);
      Link (r, p);
   end if
end
```

- `AddToAuxData`: adds additional information to a record that already contains some auxiliary data.

These function are passed to the area opening procedure as pointers, to be able to perform different attribute opening spectra using a single routine. The three important functions, mentioned in section 3.3.3, can be left unchanged, except for the Union-function. This is where the union of two sets is computed, so the auxiliary data will have to be kept consistent, which is ensured by updating `auxdata` just before the call to Link, see algorithm 6.

3.4 Experiments

The method is demonstrated on a couple of images. In this section, the similarity between Hu's moment invariants and Flusser's basis of complex moments is show first. After that, some images are described, along with a description of the performance of the standard pattern spectra on these images. Finally, the generalised pattern spectrum is determined for these images.

3.4.1 Similarity

The similarity between Hu's moment invariants and Flusser's basis of complex moments is shown using the letter Q, see figure 3.1. This letter was translated (fig. 3.1(b)), scaled (fig. 3.1(c)) and rotated (fig. 3.1(d)), resulting in four visually different images.

The results of these two methods are shown in table 3.1. From these results, we can see that both rotation and translation is handled by both Hu's moment invariants and Flusser's basis of complex

(a) (b) (c) (d)

Figure 3.1: The letter Q, used for testing the moment invariants: original (a), translated to the right corner (b), scaled to 150% of the original image (c), rotated 90° counter-clockwise (d). Note that the red part in (b) is *not* part of the actual image, but is used to demonstrate the transformation. The values of the invariants can be found in table 3.1.

Hu	Fig. 3.1(a) = (b) = (d)	Fig. 3.1(c)	Flusser	Fig. 3.1(a) = (b) = (d)	Fig. 3.1(c)
ϕ_1	$1.4013 \cdot 10^{-03}$	$1.4331 \cdot 10^{-03}$	$\Phi(1,1)$	$1.4013 \cdot 10^{-03}$	$1.4331 \cdot 10^{-03}$
ϕ_2	$3.4679 \cdot 10^{-08}$	$3.7263 \cdot 10^{-08}$	$\Phi(2,1)$	$7.4454 \cdot 10^{-12}$	$4.4892 \cdot 10^{-12}$
ϕ_3	$6.1730 \cdot 10^{-11}$	$7.5008 \cdot 10^{-11}$	$\mathrm{Re}(\Phi(2,0))$	$1.1554 \cdot 10^{-15}$	$5.3107 \cdot 10^{-16}$
ϕ_4	$7.4454 \cdot 10^{-12}$	$4.4892 \cdot 10^{-12}$	$\mathrm{Im}(\Phi(2,0))$	$-7.6640 \cdot 10^{-16}$	$-6.8477 \cdot 10^{-16}$
ϕ_5	$-8.7853 \cdot 10^{-23}$	$-5.4266 \cdot 10^{-23}$	$\mathrm{Re}(\Phi(3,0))$	$-8.7853 \cdot 10^{-23}$	$-5.4266 \cdot 10^{-23}$
ϕ_6	$1.1554 \cdot 10^{-15}$	$5.3107 \cdot 10^{-16}$	$\mathrm{Im}(\Phi(3,0))$	$1.3327 \cdot 10^{-22}$	$6.1977 \cdot 10^{-23}$
ϕ_7	$1.3327 \cdot 10^{-22}$	$6.1977 \cdot 10^{-23}$			

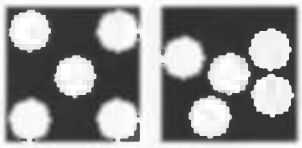
Table 3.1: The values of Hu's moment invariants and Flusser's basis of the objects from figure 3.1. The values of figure 3.1(a), 3.1(b) and 3.1(d) are equal for both invariants. The values of figure 3.1(c) differ slightly because of rounding errors in the scaling process.

moments. When the original image is scaled to a size of 150% of the original image, some minor changes occur in both the methods. This is caused by rounding errors in the scaling-process.

3.4.2 Images

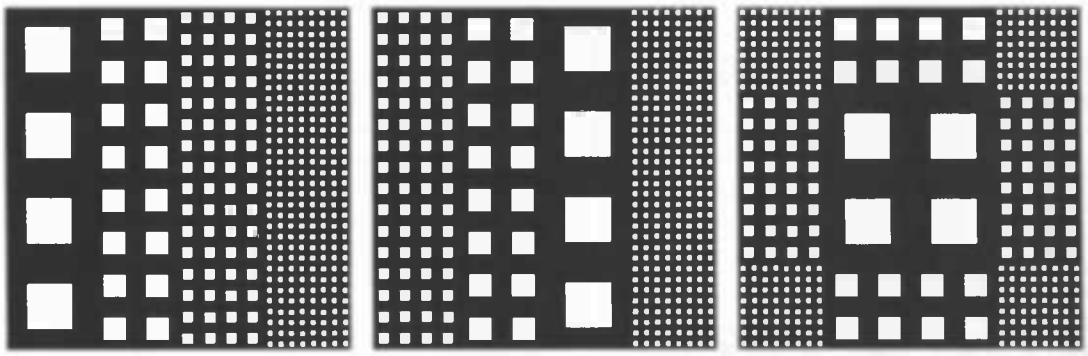
A couple of artificial images were created with the purpose of showing the distinguishing properties of the current method. One couple of images is shown in figure 3.2. These images consist of a number of circles, each with an area of 186 pixels. The only difference is the spatial arrangement of the two images. The standard pattern spectra for these images are equal: 0 if $\lambda \leq 186$ and 1 if $\lambda > 186$ (186 being the area of the circles in both images).

In figure 3.3, three other images are shown. In each image, four groups of blocks are shown, each time rearranged so they are dissimilar; in figure 3.3(a), the blocks are ordered from left (large blocks) to right (small blocks). In figure 3.3(b), the spatial arrangement of the blocks *within* a group with a particular size is maintained, but the four groups are rearranged . In figure 3.3(c), all the separate blocks are completely rearranged: the spatial arrangement of the blocks within a group is different, as well as the spatial arrangement of the groups itself. Since the only difference between these three images is the spatial distribution, the standard pattern spectra is the same as well.



(a) (b)

Figure 3.2: Two test-images; the 5 objects in both images have the same area (186 pixels), but their spatial distribution differs.



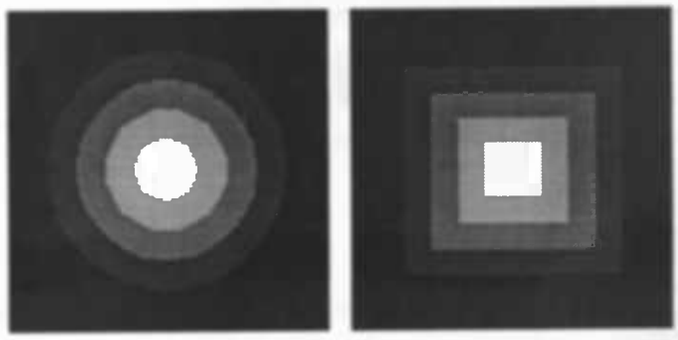
(a) (b) (c)

Figure 3.3: Three test-images; the objects (the squares) that are present in the images are the same for all images, only the spatial distribution differs.

Finally, in figure 3.4, two images are shown, which are clearly different. Both images consist of number of rings, decreasing in gray-value from the inside out. The rings that have the same gray-value in figure 3.4(a) as well as 3.4(b) also have (approximately) the same area. This time, the standard pattern spectra are not quite equal, but the differences between them can be explained by the occurrence of π in the calculation of the area of a circle: the area of a circle can never be exactly the same as the area of a square because of rounding errors in the creation of the (artificial) images.

3.4.3 Results

First, the results of the generalised pattern spectra of the images in figure 3.3 is shown. This method can, opposed to the standard pattern spectra, distinguish between these images, although the difference between the first and the second image does not become clear through the use of Hu's moment invariants. This can be explained by the fact, that Hu's moment invariants are invariant to translation, while this is actually all that is different between image 3.3(a) and 3.3(b); two groups containing the objects of the same size have swapped position, e.g. one group was translated to the left, while another group was translated to the right. Image 3.3(a) and 3.3(c) (and, hence, image 3.3(b) and 3.3(c)) *can* be distinguished by the generalised pattern spectra through the use of Hu's moment invariants.



(a) (b)

Figure 3.4: Two test-images; rings with equal gray-value have (approximately) the same area.

Hu	Outer ring	second ring	middle ring	fourth ring	inner component
ϕ_1, Fig. 3.4(a)	$4.9738 \cdot 10^{-03}$	$4.9737 \cdot 10^{-03}$	$2.4878 \cdot 10^{-03}$	$2.4878 \cdot 10^{-03}$	$2.5284 \cdot 10^{-03}$
ϕ_1, Fig. 3.4(b)	$5.2145 \cdot 10^{-03}$	$5.2142 \cdot 10^{-04}$	$2.6067 \cdot 10^{-03}$	$2.6058 \cdot 10^{-03}$	$2.6424 \cdot 10^{-03}$
ϕ_2, Fig. 3.4(a)	$2.1507 \cdot 10^{-19}$	$1.8746 \cdot 10^{-15}$	$1.0343 \cdot 10^{-17}$	$1.4788 \cdot 10^{-15}$	$3.1414 \cdot 10^{-12}$
ϕ_2, Fig. 3.4(b)	$6.9988 \cdot 10^{-08}$	$6.9634 \cdot 10^{-08}$	$1.7262 \cdot 10^{-08}$	$1.6975 \cdot 10^{-08}$	$1.6673 \cdot 10^{-08}$
ϕ_3, Fig. 3.4(a)	$2.8463 \cdot 10^{-14}$	$5.8569 \cdot 10^{-13}$	$6.6472 \cdot 10^{-14}$	$2.9637 \cdot 10^{-13}$	$2.2415 \cdot 10^{-12}$
ϕ_3, Fig. 3.4(b)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$
ϕ_4, Fig. 3.4(a)	$4.4677 \cdot 10^{-18}$	$7.7115 \cdot 10^{-17}$	$1.2899 \cdot 10^{-18}$	$8.6805 \cdot 10^{-18}$	$4.3451 \cdot 10^{-15}$
ϕ_4, Fig. 3.4(b)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$
ϕ_5, Fig. 3.4(a)	$-1.5932 \cdot 10^{-33}$	$-5.1825 \cdot 10^{-31}$	$-3.7772 \cdot 10^{-34}$	$-1.3923 \cdot 10^{-32}$	$4.2881 \cdot 10^{-28}$
ϕ_5, Fig. 3.4(b)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$
ϕ_6, Fig. 3.4(a)	$-2.0719 \cdot 10^{-29}$	$-3.3388 \cdot 10^{-24}$	$-4.1486 \cdot 10^{-27}$	$-3.3381 \cdot 10^{-25}$	$-7.7012 \cdot 10^{-21}$
ϕ_6, Fig. 3.4(b)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$
ϕ_7, Fig. 3.4(a)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$
ϕ_7, Fig. 3.4(b)	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$	$0.0000 \cdot 10^{-00}$

Table 3.2: The values of Hu's moment invariants of the two images in figure 3.4. The values belong, from left to right, to the outer ring (not the background), the second ring, the middle ring, the fourth ring, and the inner component.

Since Hu's moment invariants are invariant to translation, and the only difference between the first and the second image is the translation of two groups of objects, other features can be determined that *can* distinguish between the two images. One of the possibilities is the second order moment m_{11}, which can easily be determined used the information that has been saved for the calculation of Hu's moment invariants. This moment *can* distinguish between the two images; the values for the two groups that have swapped position is different for the two images.

The images in figure 3.4 can simply be distinguished: the set of Hu' moment invariants is different for the two images, see table 3.2.

Chapter 4

Spatial size distribution

Another extension to the standard size distributions is found in [3]. Ayala and Domingo [3] propose new descriptors for binary and gray-scale images based on newly defined *spatial size distributions* in this article.

In [3], the granulometric size distribution is defined as a probability function with a cumulative distribution function. This results in the following definition for the *granulometric size distribution* GSD for binary and gray-scale images, respectively:

$$GSD_X(\lambda) = 1 - \frac{A(\Psi_\lambda(X))}{A(X)} = \frac{A(X) - A\Psi_\lambda(X))}{A(X)} \tag{4.1}$$

$$GSD_f(\lambda) = 1 - \frac{\int_W \Psi_\lambda f(x)dx}{\int_W f(x)dx} = \frac{\int_W f(x)dx - \int_W \Psi_\lambda f(x)dx}{\int_W f(x)dx} \tag{4.2}$$

where $A(X)$ stands for the area of X.

These functions calculate the area of the image that results from the granulometry Ψ_λ and divide this area by the area of the original image, so in fact the area of set X is compared with the area of its transformed $\Psi_\lambda(X)$.

The granulometric size distribution as defined above only described the amount of detail present at some scale. If any information on the spatial distribution of the detail at some scale has to be included, the comparison between the texture and granulometric transform by using only their shapes, is too simple a choice. The spatial information can be included by comparing two intersections: the intersection of the texture and the texture translated along a vector h, and the intersection of the granulometric transform of the texture and the granulometric transform of the texture translated along the same vector h. This results in the following (cumulative distribution) functions, called *spatial size distribution* SSD:

$$SSD_{X,U}(\lambda, \mu) = \frac{1}{A(X)^2} \int_{\mu U} A(X \cap (X + h)) - A(\Psi_\lambda(X) \cap (\Psi_\lambda(X) + h))dh \tag{4.3}$$

for binary images and

$$SSD_{f,U}(\lambda, \mu) = \frac{\int_{\mu U} \int_W f(x)f(x+h) - \Psi_\lambda(f(x))\Psi_\lambda(f(x+h))dxdh}{(\int_W f(x)dx)^2} \tag{4.4}$$

for gray-scale images, where U is a convex and compact subset containing the origin in its interior and W the window over with equation (4.4) is defined.

Algorithm 7 Pseudo-code for the calculation of the spatial size distribution

```
transVectors = determineTranslationVectors();
for all vectors v in transVectors do
  for all pixels p in image f do
    ssd = ssd + f(p) * f(p + v) - Ψf(p) * Ψf(p + v);
  end for
end for
ssd = ssd/sum_of_all_pixels(f)^2;
```

When looking at gray-scale images, this function produces the difference between the fractions of points that belong to $f(x)$ and its translate $f(x+h)$, and the fraction of points that belong to $\Psi_\lambda(f(x))$ and its translate $\Psi_\lambda(f(x+h))$. Hence, if the granulometry of $f(x)$ (with parameter λ) produces the same image as $f(x)$ itself (which is the case when $\lambda = 0$, for instance), the spatial size distribution will be zero.

4.1 Implementation

In the implementation of the spatial size distribution, the function used to perform the granulometry is an *area opening*, which is derived from attribute openings (see [7] for a thorough discussion on attribute openings and [40] for a more detailed discussion on area openings). The same implementation for the area opening will be used as was used for the implementation of the general pattern spectrum [42], which is described in [19].

In order to implement the spatial size distribution, the definition as given in equation (4.4) has to be rewritten to the discrete case. The result is given below.

$$SSD_{f,U}(\lambda,\mu) = \frac{\sum_{h\in\mu U}\sum_{x\in W} f(x)f(x+h) - \Psi f(x)\Psi f(x+h)}{(\sum_{x\in W} f(x))^2} \tag{4.5}$$

The first thing that has to be done, is the calculation of the set μU, to determine all possible values for h. For U, a disc centered at the origin with a radius of 1 will be used. This set of translation vectors is in figure 7 stored as `transVectors`. After that, the actual calculation of the spatial size distribution (ssd) can be done. The pseudo-code can be seen in algorithm 7.

4.2 Experiments

To test the spatial size distribution, the same images that were introduced in section 3.4.2 are used. First, the relation between λ and μ is shown using images which are obviously different, but share the same granulometric size distribution. The images that are used for this purpose are the five objects with different spatial distribution, figure 3.2. After that, the spatial size distribution is shown for the other images presented in section 3.4.2, and the results are discussed.

4.2.1 Relation between λ and μ

The images which are used to show the relation between λ and μ are shown in figure 3.2. They consist of a number of circles, each with an area of 186 pixels. The only difference is the spatial arrangement of the two images.

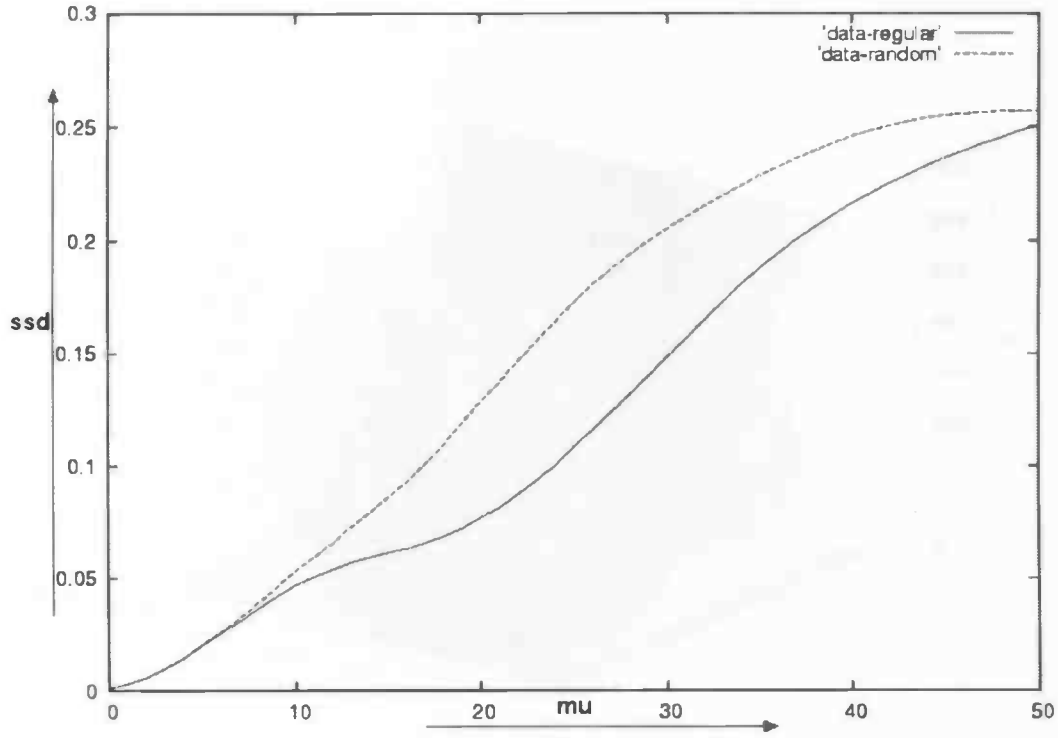


Figure 4.1: The spatial size distribution for figure 3.2(a) (the line marked 'data-regular') as well as figure 3.2(b) (the line marked 'data-random'), for $\lambda = 187$

The granulometric size distribution, e.g. the standard pattern spectrum, for the images is 0 if $\lambda \leq 186$ and 1 if $\lambda > 186$ (186 being the area of the circles in both images). The spatial size distribution for each of these images is 0 if $\lambda \leq 186$. In figure 4.1, the ssd is shown for values of $\lambda > 186$. The value for λ used to create the plots in figure 4.1 is 187, but all values for λ larger than 186 will produce the same plots.

In figure 4.2 one can see the difference between the spatial size distribution of the images in figure 3.2(a) and 3.2(b). For λ-values smaller than 187, the difference is zero, since both spatial size distributions are equal for those values of λ: zero.

It can be seen that the λ-component can not distinguish between the images, whereas the μ-component is clearly different. This is caused by the fact that the objects in the images only have different relative locations from another; the sizes of all objects are the same. From this, it can be seen that the first marginal of the spatial size distribution (the λ-component) is the size component, while the second marginal (the μ-component) is concerned with the spatial arrangement. Consequently, the proposed cumulative distribution (equation (4.3) can be split up into two functions, representing the general expressions of the size component and the spatial arrangement, respectively:

$$F_1(\lambda) = 1 - \frac{A(\Psi_\lambda(X))^2}{A(X)^2} \tag{4.6}$$

$$F_2(\mu) = \frac{1}{A(X)^2} \int_{\mu U} A(X \cap (X + h))dh \tag{4.7}$$

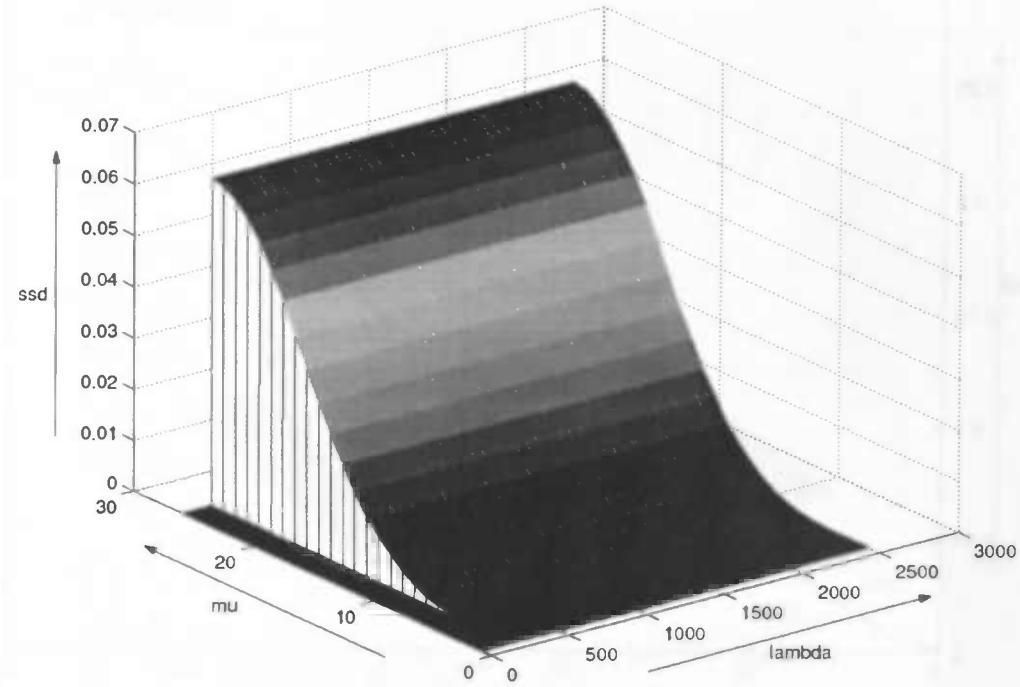


Figure 4.2: The difference between the ssd for figure 3.2(a) and 3.2(b).

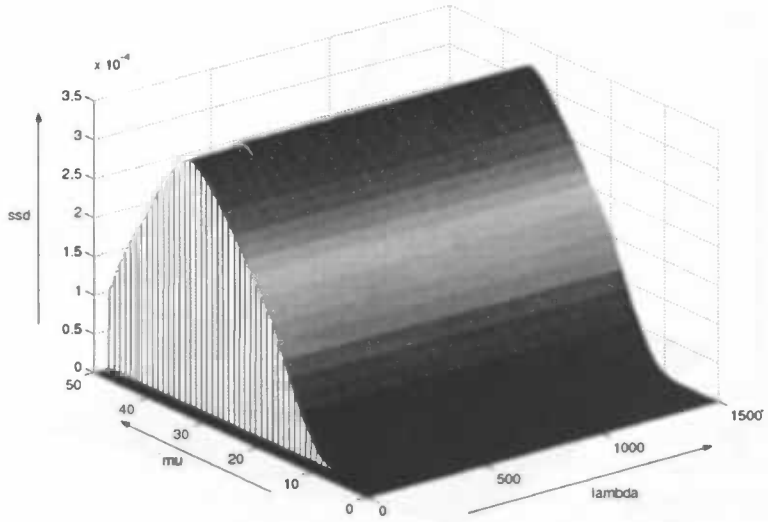
4.2.2 Results and discussion

The first experiments are performed using the images with the groups of blocks, figure 3.3.

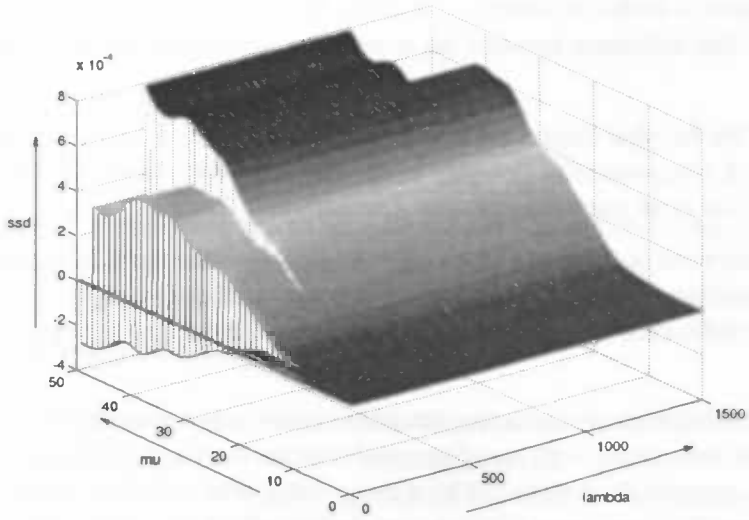
As explained in section 3.4.2, the standard pattern spectrum is identical for all three images. The spatial size distribution *is* different for these images, as expected. In figure 4.3, it can be seen that the spatial size distribution of the first image differs from the second and the third. The two plots are the differences between the spatial size distribution of image 3.3(a) and 3.3(b) (figure 4.3(a), and image 3.3(a) and 3.3(c) (figure 4.3(b)). The difference between image 3.3(b) and 3.3(c) is not shown here, but is similar to figure 4.3(b).

Although the differences are small (order of magnitude of about 10^{-4}), the spatial size distribution *is* a good distinguisher between these images, since the images are somewhat similar: the difference between the first and the second image is only the spatial arrangement of the different groups. The difference between the first and the third (as well as the second and the third) is a bit more obvious, but the difference between the spatial size distributions of these images is more clear as well: the difference between image 3.3(a) and 3.3(c) is about twice are large as the difference between image 3.3(a) and 3.3(b).

The next experiment is performed on the two images in figure 3.4, which are clearly different. Surprisingly, though, the granulometric size distribution *nor* the spatial size distribution can distinguish between the images. The fact that the granulometric size distributions are equal for both images can



(a)



(b)

Figure 4.3: The difference between the spatial size distribution for figure 3.3(a) and 3.3(b), subrefsubfig:plot-blocks12, and the difference between the spatial size distribution for figure 3.3(a) and 3.3(c), (b).

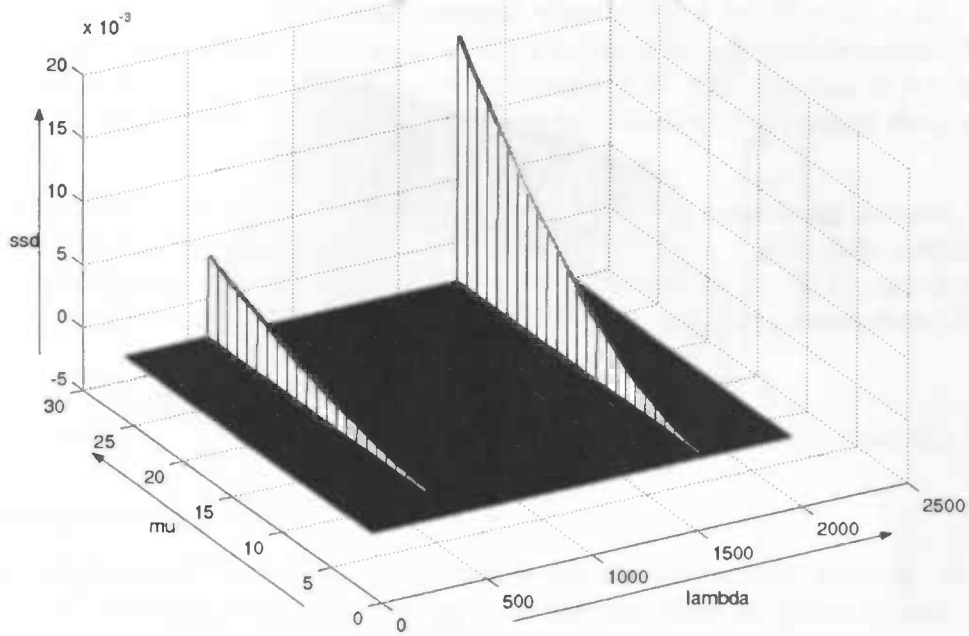


Figure 4.4: The difference between the spatial size distribution for the images in figure 3.4.

be explained by the fact that images are discriminated solely based on the size of the gray-value components. Since all components of one gray level (approximately) have the same area, they also have (approximately) the same granulometric size distribution.

The difference between the two spatial size distributions are shown in figure 4.4. The two peaks are explained by rounding-errors in the creation of the two (artificial) images, as explained before. This is the reason the difference is (in general) not exactly zero, but slightly (neglectibly) around zero.

The spatial size distribution being (approximately) equal is explained by the fact that the spatial arrangement of the components with equal gray-value is the same in both images: the white component is in the middle, completely surrounded by a component with a slightly darker colour. Around these components is another component with an even darker colour, etc. Since the spatial size distribution discriminated images based on the size as well as the spatial arrangement of the components, and these are equal in both images, it can not distinguish between the two images.

From this, a general drawback of the spatial size distribution can be derived: spatial size distributions do not discriminate images based on shape. When the shape of two images is different, but the size and spatial arrangement coincide, the spatial size distribution is not able to differentiate between the two images. This is in contrast with the method of the generalised pattern spectra, as explained in section 3.4.3.

Chapter 5

Multi-scale connectivity

A third extension to the standard size distribution is based on multi-scale connectivity [6]. In this section, some basics about connectivity and multi-scale connectivity will be explained. After that, the connectivity opening and a method to implement this operator is explained, followed by a discussion about how this connectivity opening will be used to extract spatial information from images. Finally, the method is performed on the images introduced in chapter 3.

5.1 Connectivity

According to Serra [29], a family $\mathcal{C} \subseteq \mathcal{P}(E)$ (with $\mathcal{P}(E)$ being a powerset $\mathcal{P}$ of an arbitrary non-empty set E; a powerset of a set S is the set of all S's subsets) is called a connectivity class if it satisfies the following two conditions:

i. $\emptyset \in \mathcal{C}$ and $\{x\} \in \mathcal{C}$ for all $x \in E$,

ii. $\bigcap C_i \in \mathcal{C} \Rightarrow \bigcup C_i \in \mathcal{C}$ for each family $\{C_i\} \in \mathcal{C}$.

Thus, the connectivity on E is defined by the class $\mathcal{C}$, and a *connected component* or *connected set* is any element of $\mathcal{C}$. In other words, $\mathcal{C}$ is the set of all connected sets.

5.2 Multi-scale connectivity

By using connectivity, regions or objects of interest in an image can be defined. However, since important information might not be confined to a particular scale, but may span several scales, Braga-Neto and Goutsias [6] derived a mathematical approach to connectivity in a multi-scale framework.

Two notions are used to define multi-scale connectivity, being *connectivity measure* and *connectivity pyramid*, which are shown to be equivalent by Braga-Neto and Goutsias [6]. The first is used to describe the varying degree of connectivity that may be assigned to a given object, depending on the scale of the object. The second described the idea that connectivity can be defined as a function of scale.

In what follows, Σ denotes a scale set. When $\Sigma = \overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$, Σ is referred to as the continuous-scale space, while when $\Sigma = \mathbb{K} = \{0, 1, \ldots, K\}$, Σ is referred to as the discrete-scale space. The greatest element of Σ is denoted by I_Σ, while the least element is denoted by O_Σ. The scale set *minus* the least element ($\Sigma \backslash \{O_\Sigma\}$) is denoted by Σ_0.

5.2.1 Connectivity measure

A connectivity measure is denoted by the function φ. The quantity $\varphi(A)$ indicated the degree of connectivity. The set A is said to be *fully connected* when $\varphi(A) = I_\Sigma$. If $\varphi(A) = O_\Sigma$, A is said to be *fully disconnected*. When $O_\Sigma < \varphi(A) < I_\Sigma$, A is said to be intermediate connected. This is defined by saying that A is σ-connected if $\varphi(A) \geq \sigma$ for $\sigma \in \Sigma$. The σ-section of φ is defined by $X_\sigma(\varphi) = \{A \in \mathcal{P}(E) | \varphi(A) \geq \sigma\}$ for $\sigma \in \Sigma$, hence the σ-section of φ contains the σ-connected elements.

Two conditions have to be met before a function φ is said to be a connectivity measure. The first condition requires that the least element and all singletons $\{x\}$ for $x \in E$ be fully connected. The second condition requires that the degree of connectivity of a family $\{A_\alpha\} \in \mathcal{P}(E)$, such that $\bigcap A_\alpha \neq \emptyset$, must not be smaller than the least degree of connectivity of the individual elements in $\{A_\alpha\}$:

i. $\varphi(\emptyset) = \varphi(x) = I_\Sigma$ for $x \in E$,

ii. for a family $\{A_\alpha\} \in \mathcal{P}(E)$ such that $\bigcap A_\alpha \neq \emptyset$, we have that $\varphi(\bigcup A_\alpha) \geq \bigcap \varphi(A_\alpha)$.

5.2.2 Connectivity pyramid

A concept closely related to connectivity measure is that of a connectivity pyramid. A mapping $\mathbf{C} : \Sigma_0 \rightarrow \mathcal{P}(E)$ is called a *connectivity pyramid* is the following three conditions are met:

i. $\mathbf{C}(\sigma)$ is a connectivity class in $\mathcal{P}(E)$ for each $\sigma \in \Sigma_0$,

ii. $\mathbf{C}(\sigma) \subseteq \mathbf{C}(\tau)$ for $\sigma \geq \tau$,

iii. $\mathbf{C}(\bigcup \sigma_\alpha) = \bigcup \mathbf{C}(\sigma_\alpha)$ for all nonempty set of scales $\{\sigma_\alpha\} \subseteq \Sigma_0$.

The first condition states, that a connectivity pyramid $\mathbf{C}$ should consist only of connectivity classes. The third condition actually implies the second. The third condition imposes a smoothness constraint on the levels of a connectivity pyramid, while the second requires that the σ-levels (i.e. the connectivity classes $\mathbf{C}(\sigma)$) of a connectivity pyramid be nested, so that fewer elements are connected as one moves upward in the pyramid. Hence, more objects tend to be connected at small scales than at large scales, since larger scales are higher up the pyramid.

5.3 Connectivity opening

When a connectivity class $\mathcal{C}$ is given, containing the connected set C_x with $x \in X$, the binary connected opening Γ_x introduced in section 3.3.1 can be redefined as:

$$\Gamma_x(X) = \bigcup \{C_x \in \mathcal{C} | x \in C_x \in \mathcal{C} \wedge C_x \subseteq X\}. \tag{5.1}$$

Hence, the connected opening Γ_x is defined as the union of all $C_x \in \mathcal{C}$ containing $x \in X$.

One particular case of a multi-scale connectivity operator is the *clustering based connected opening*. This operator is concerned with finding groups of connected components that can be seen as a cluster if their relative distances are below a certain threshold. To verify the relative distances between connected components, an increasing and extensive operator ψ_c (e.g. a dilation) is used. This increasing and extensive operator modifies the connectivity accordingly and the shape and structure (i.e. morphology) of the clusters in the original image is defined by the resulting connectivity class $\mathcal{C}^{\psi_c}$. In

other words, the clustering based connected opening extracts the connected components according to Γ_x in $\psi_c(X)$ (instead of X) and restricts this result to members of X. The operator is given for binary images by:

$$\Gamma_x^{\psi_c}(X) = \begin{cases} \Gamma_x(\psi_c(X)) \bigcap X & \text{if } x \in X \\ \emptyset & \text{if } x \notin X \end{cases} \tag{5.2}$$

The corresponding gray-scale operator can be obtained by means of *stacking*, as explained in section 2.1:

$$\gamma_x^{\psi_c}(f)(x) = \max\{h | x \in \Gamma_x^{\psi_c}(T_h(f))\}, \tag{5.3}$$

where $T_h(f)$ is the result of thresholding f at gray level h.

Another particular case of interest is the *partitioning based connected opening*, which splits one connected component with at least one small narrow "bridge" between parts of the connected components, into several connected components. These bridges are often present due to image noise, background texture or out of focus details. The bridges are determined by modifying the original image with an increasing, anti-extensive and idempotent operator ψ_p (e.g. an opening). Same as for the clustering based opening, the operator ψ_p modifies the connectivity accordingly and the morphology of the clusters in the original image is defined by the resulting connectivity class C^{ψ_p}: the partitioning based connected opening extracts the connected components according to Γ_x in $\psi_p(X)$ (instead of X) and restricts this result to members of x. The operator is given for binary image and gray-scale images, respectively, as follows:

$$\Gamma_x^{\psi_p}(X) = \begin{cases} \Gamma_x(\psi_c(X)) \bigcap X & if\ x \in X \\ \{x\} & if\ x \in X \setminus \psi_p(X) \\ \emptyset & if\ x \notin X \end{cases} \tag{5.4}$$

$$\gamma^{\psi_p}(f)(x) = \max\{h | x \in \Gamma_x^{\psi_p}(T_h(f))\}, \tag{5.5}$$

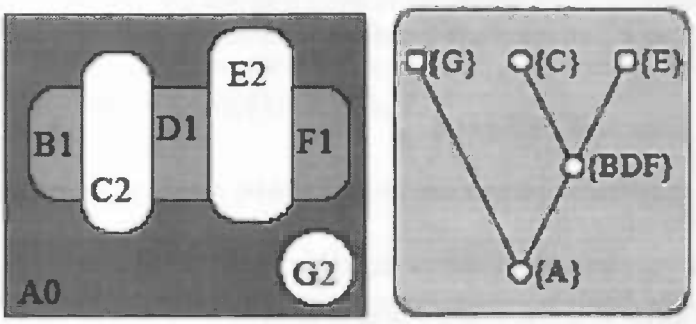
where T_h is the result of thresholding f at gray level h.

5.4 Max-tree

The algorithm that is used to implement the second-order attribute openings is based on the *Max-tree algorithm*, which was first introduced by Salembier et al. [27]. The Max-tree was originally introduced as a versatile data-structure to implement anti-extensive connected set operators; it is a rooted tree, with which the connected components of an image can be stored and accessed in an efficient way. The multi-scale connected opening was implemented using the *Dual Input Max-tree* [24].

The connected components of the image are stored in the Max-tree; peak components are stored as nodes, regional maxima are stored in a separate node, and every layer in the tree corresponds to a separate gray level from the image. An example is take from [27] and can be seen in figure 5.1.

A Max-tree node C_h^k (k is the node index) corresponds to a peak component P_h, but only those pixels in P_h with gray-value h are stored in node C_h^k. Each node (except for the root) points towards its parent $C_{h'}^{k'}$ with $h' < h$; the parent of each node (except for the root) is defined to be the connected component which is (partially) overlapped by the connected component corresponding to the current node. If the current connected components (partially) overlaps more than one component, the connected component that was processed last (before the current one) will be set to be the parent. Since the root node is defined at the minimum gray level h_{min}, there is no connected component with a gray-value smaller than h_{min}, and hence the root node has no parent.



(a) (b)

Figure 5.1: An example of the maxtree, (b), for the corresponding image, (a). Taken from [27].

5.4.1 Construction

The Max-tree corresponding to a certain image is constructed using a recursive flooding algorithm. First, the lowest gray-value present in the image is determined and assigned to h_{min}. Using this, the root node $C^0_{h_{min}}$ is created. After that, the image is thresholded at gray level h_{min}, which causes all pixels with gray-value h_{min} to become background pixels and pixels with gray-value larger than h_{min} to become foreground pixels. Now, every connected component (consisting of foreground pixels) is assigned to a temporary node TC^k_h.

The recursive part of the algorithm consists of processing all temporary nodes, the same way as was described above for the root node. For each temporary node, the lowest gray value is determined, and this value becomes the *local background*. The threshold is set to be the value of the local background and, after thresholding, all connected components consisting of foreground pixels are assigned to new temporary nodes, while the temporary nodes with the connected components containing background pixels become "normal nodes", i.e. nodes in the final tree, and consequently become the parent of the newly created temporary nodes. When all temporary nodes have been processed, all empty nodes are deleted from the tree.

A fast implementation of the Max-tree algorithm is given in [27] and relies on the use of a hierarchical FIFO queue, called `hqueue`. Each separate gray level h is assigned an individual first-in-first-out queue, taken from `hqueue`. These queues are used to define an appropriate scanning and processing order of the pixels: the priority pixels are reassigned to the Max-tree structure, and new pixels retrieved from the neighbourhood of the current priority pixel are assigned to the appropriate entries in `hqueue`. The following functions are used by the main flooding routine (see function 8):

- `hqueue-add(h,p)`: add the pixel p (of gray-value h) in the queue of priority h.
- `hqueue-first(h)`: extracts the first available pixel of queue of priority h.
- `hqueue-empty(h)`: return "true" if queue of priority h is empty.

Furthermore, the following arrays are used:

- `number-nodes`: stores the number of nodes C^k_h at level h.

- `nodes-at-level`: a boolean array that flags the presence of a node still being flooded at level h.
- `ori`: denotes the original gray-value of pixel p.
- `status`: stores information about the pixel status: a pixel p can either be "not-analysed", "in-the-queue" or already assigned to node k at level h, in which case `status[p]=k`.

Using these arrays, we can determine the node to which pixel p belongs: $C_{\texttt{ori[p]}}^{\texttt{status[p]}}$.

As said before, a recursive flooding algorithm is used to create the Max-tree. A pixel with the lowest gray-value (h_{min}) is used as seed; the max-tree is created by calling `flood(`$h_m min$`)`. The procedure has two basic steps: first, the propagation and the updating of `status` is done and second, the parent-child-relations are defined (see algorithm 8 for pseudo-code of the algorithm).

5.4.2 Dual Input Max-tree

The two instantiations of the second-order attribute opening were implemented using a modified version of the Max-tree, called Dual Input Max-tree [24]. This algorithm requires *two* input images: the first is the original image, as would be used as input image for the regular Max-tree algorithm, while the second input image is a copy of the original image, modified by an extensive, clustering, operator (like a dilation) in the case of a clustering based connected opening and by an anti-extensive, partitioning, operator (like an opening) in the case of a partitioning connected opening.

The main idea of the Dual Input Max-tree algorithm is to determine which pixels belong to the same connected component by using the modified image, and then restrict the results to the connected components of the original image. Hence, the tree is shaped by modified image, while the date that is mapped on the Max-tree is taken from the original image. In algorithm 9 the code of the Dual Input Max-tree can be found; the difference between the original Max-tree and the Dual Input Max-tree is underlined.

The seed that is used to start the flooding process, h_{min}, is taken from the modified connectivity map $\psi(X)$. At first, the flooding function proceeds as described earlier: the neighbours of the current pixel are inspected and distributed into the appropriate queues. After updating the status of the current pixel, though, a test condition is added: if there is an intensity mismatch between the same pixel in both images (the original image and the modified image), there are two possibilities:

- ψ is an extensive operator, in which case pixel p is a background pixel in the original image and part of a connected component at the current gray level in the modified image.
- ψ is an anti-extensive operator, in which case pixel p is part of a connected component in the original image which is discarded as a result of the partitioning based connected opening. Hence, pixel p is treated as a singleton

In both cases, the value of the appropriate element in the array `node-at-level` is set to true. When pixel p should be treated as a singleton (i.e. ψ is an anti-extensive operator), the node at the current gray level should be finalised before retrieving the next priority pixel. This finalising is done by first setting the parent of the current node to a pixel with gray-value of pixel p in the modified image, second by setting the status of the current pixel to the node index at level `ori[p]`, and third by altering the arrays `number-nodes` and `node-at-level` accordingly.

After these cases are handled, the flooding procedure continues as described earlier, only each time the regular flooding procedure consults `ori` to look up the intensity of a certain pixel, the modified flooding procedure consults `p_ori` to discover the intensity. The array `p_ori` denotes the array storing the pixel intensities of the modified image.

5.5 Spatial information

Spatial information about the images can be extracted by using the clustering based connected opening. From the original image, a (regular) pattern spectrum can be obtained, while the pattern spectrum of the same image, modified by the clustering based connected opening, might be (slightly) different, since components that are partitioned in the original image might be connected in the modified image. When other maps are used to determine the clustering based opening (e.g. a dilation with a larger structuring element), the components might be clustered in yet other ways, resulting in other pattern spectra as well. Since the operator used by the clustering based connected opening is increasing an extensive, the different pattern spectra tell something about how close the connected components are to each other, and hence, they tell something about the spatial distribution of the connected components.

To obtain spatial information, several modified images have to be obtained. A number of n modified images are created by dilating the original image n times; each time a dilation is completed, one modified image is obtained. The regular pattern spectrum can be extended to the multi-scale connectivity pattern spectrum as follows:

$$MSC_f(\lambda, i) = A(\gamma_\lambda(\gamma^{\psi_i}(f)) - \gamma_{\lambda+1}(\gamma^{\psi_i}(f))), \tag{5.6}$$

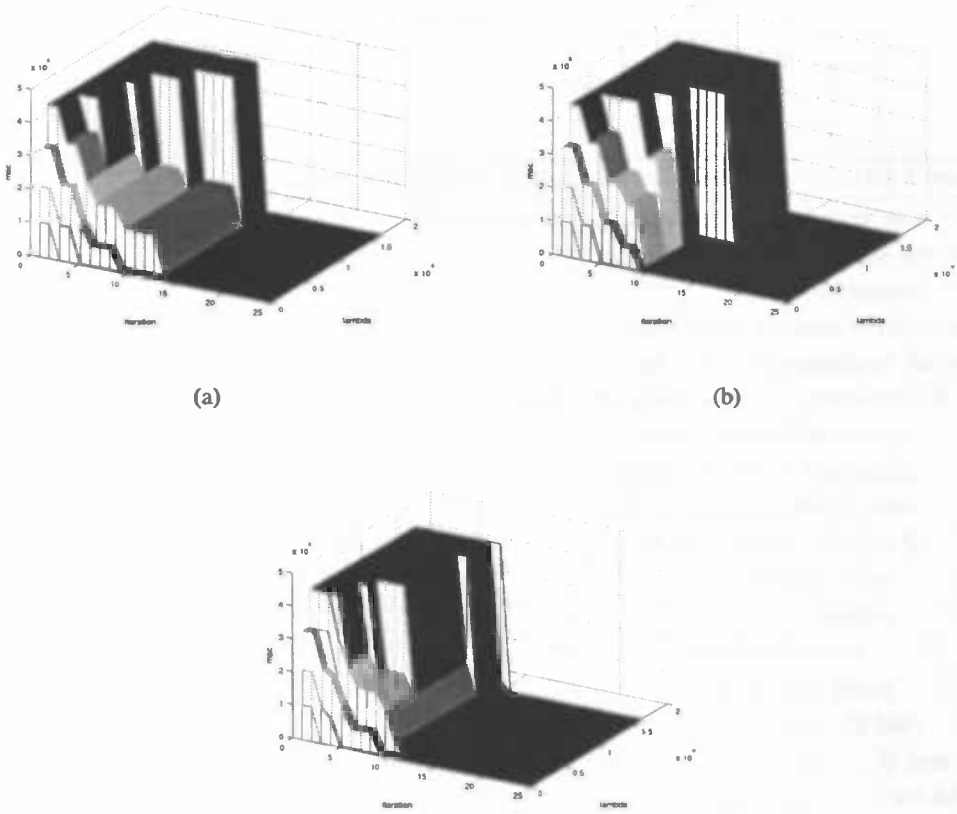
where γ_λ is used to denote the area opening with area parameter λ and γ^{ψ_i} denotes the connected opening with the modified image, resulting from the i-th time operator ψ is performed on the original image, as connectivity map. Note that $\gamma^{\psi_0}(f) = f$, e.g. the connected opening of f as original image and f as modified image is equal to the original image. Hence, it can be seen that the multi-scale connectivity pattern spectrum is an *extension* to the regular pattern spectrum.

While the clustering based connected opening tells something about the closeness of connected components, the partitioning based connected opening can be used to discover something about the tendency of components to be separated by consecutive openings (or erosions with increasing structuring elements). When a connected component is divided into two or more connected components after an erosion with a structuring element of some size, this can be used to characterise the connected component. So, for values of $i \geq 0$, a specific instantiation of the operator ψ is a dilation, while when $i < 0$, an instantiation of ψ is an erosion.

5.6 Experiments

For showing some results for this method, the images in figure 3.2, 3.3 and 3.4 were used.

The images in figure 3.2 can clearly be distinguished by the multi-scale connectivity pattern spectra. The regular pattern spectrum is equal for both images: 5 connected components of 186 pixels each are discovered in both cases. When the value of i in equation (5.6) is increased, though, the pattern spectrum differs: in the case of figure 3.2(a), the 5 objects of 186 pixels each will be seen as one connected component of 930 pixels when $i = 3$, while figure 3.2(b) will do the same when $i = 1$. So, for values of $i < 1$ and $i > 3$, the multi-scale connectivity pattern spectrum for both images is equal;



(a) (b)

(c)

Figure 5.2: The plot of the multi-scale connectivity pattern spectra for the images in figure 3.3. In (a) the plot of figure 3.3(a) is shown, in (b) the plot of figure 3.3(b) and in (c) the plot of figure 3.3(c).

the difference between the two images is characterised by a difference turning point: for figure 3.2(a) this turning point lies at $i = 3$, for figure 3.2(b) it lies at $i = 1$.

The three images containing the blocks, figure 3.3, are also very clearly distinguished by the multi-scale connectivity pattern spectra. This can be seen in figure 5.2: the results are shown only for values of $i > 0$, since the blocks are convex, so the plot would look the same for all three images for values of $i < 0$. In these plots, one can see that the range of the plots is equal for all three images, but the *turning points* (e.g. the points at which "suddenly" a different value is being calculated) differ. In each plot, four plateaus (excluding the ground-plateau) can be distinguished, corresponding to the four different-sized partitions of blocks.

For the final two images, figure 3.4, this is also the case: the range of the plots are the same, but the turning points differ, causing the ability of the multi-scale connectivity pattern spectrum to distinguish between the two images.

Function 8 `flood(h)` : recursive flooding procedure for the Max-tree creation.

```
{First step: propagation and updating of status}
while not hqueue-empty(h) do
  p = hqueue-first(h);
  status[p] = number-nodes[h];
  for all neighbours nb of p do
    if status[nb] == 'not-analysed' then
      hqueue-add(ori[nb],nb);
      status[nb] = 'in-the-queue';
      node-at-level[ori[p]] = true;
      if ori[nb] > ori[p] then
        m = ori[nb];
        repeat
          m = flood(m);
        until m == h
      end if
    end if
  end for
end while
number-nodes[h]= number-nodes[h]+1;

{Second step: define the parent-child-relations}
m = h - 1;
while m ≥ 0 and node-at-level[m]= false do
  m = m - 1;
end while
if m ≥ 0 then
  i = number-nodes[h]-1;
  j = number-nodes[m];
  parent of node C_h^i = node C_m^j;
else
  node C_h^i has no father;
end if
node-at-level[h]= false;
return m;
```

Function 9 `flood(h)`: Dual Input Max-tree

```
{First step: propagation and updating of status}
while not hqueue-empty(h) do
  p = hqueue-first(h);
  status[p] = number-nodes[h];
  if ori[p] ≠ h then
    node-at-level[ori[p]] = true;
    if ori[p] ¿ h then
      i = number-nodes[ori[p]];
      j = number-nodes[h];
      parent of node C^i_ori[p] = node C^j_h;
      status[p] = number-nodes[ori[p]];
      number-nodes[ori[p]] + = 1;
      node-at-level[ori[p]] = false;
    end if
  end if
  for all neighbours nb of p do
    if status[nb] == 'not-analysed' then
      hqueue-add(p_ori[nb],nb);
      status[nb] = 'in-the-queue';
      node-at-level[p_ori[p]] = true;
      if p_ori[nb] > p_ori[p] then
        m = p_ori[nb];
        repeat
          m = flood(m);
        until m == h
      end if
    end if
  end for
end while
number-nodes[h]= number-nodes[h]+1;

{Second step: define the parent-child-relations}
m = h − 1;
while m ≥ 0 and node-at-level[m]= false do
  m = m − 1;
end while
if m ≥ 0 then
  i = number-nodes[h]−1;
  j = number-nodes[m];
  parent of node C^i_h = node C^j_m;
else
  node C^i_h has no father;
end if
node-at-level[h]= false;
return m;
```

Chapter 6

Testing

In this section, the three feature extraction methods will be tested to see how they perform as content-based image retrieval system. First, the design of the experiments will be explained; what database is used, which parameters for the methods are chosen, how are the experiments set up. Furthermore, the performance measures that are used to evaluate the results of the experiments are discussed. After that, results of the three feature extraction methods are shown and discussed. For comparison, the regular (area) pattern spectrum is implemented and tested too, and the commercial program *IMatch* is tested on the used databases as well.

The main purpose of this extensive framework and very clearly specified choices, is to prevent the "enhancement" of the apparent performance, as was described in [21]. Müller et al. [21] described how easy the apparent performance can be "enhanced" by applying techniques that all have a certain justification, but do not really enhance the performance, but only *seem* to enhance the performance. Even though the purpose here is, to compare three methods, and hence all three methods will be subjected to the same tests, some of the performance-improvement techniques might have different effects on the different methods. Furthermore, the goal is not to show how good a certain method is (since the optimal setting will not be determined), but to find out which features described the images best. In order to do so, all features are used in the same system, and are subjected to the same tests, which are described hereafter.

6.1 Design of experiments

Ever since the 1950s, evaluation of retrieval performance is a crucial problem in information retrieval (IR). In 1992, many efforts to provide a common performance test were combined in the TREC, Text REtrieval Conference series [1]. The current state of performance evaluation in content-based image retrieval (CBIR), which is closely related to information retrieval, is far from that in IR: often, the performance of CBIR systems is shown by just printing the results of some queries. This way, the performance can easily be adjusted by choosing queries which give good results; also, the experiments can not be duplicated since usually, the database that is used is either not known or not available, and therefore, different systems can not be easily compared. Therefore, a great need exist for standardisation of the performance measures in CBIR, as was done in IR by introducing TREC. One initiative to begin such a standard is the Benchathlon project [2], but so far, this is not much more than a proposition.

[1] `http://trec.nist.gov`

[2] `http://www.benchathlon.net`

In this section, a number of choices with respect to the experiments are made. These include the choice of the image database, the number of features that each method will determine for every image, the distance function to determine the similarity between two feature sets and the performance measures that can be used to interpret the experiments results. These choice are explained in this section. After that, the experiments that are performed are described.

6.1.1 Databases

For the purpose of testing, a suitable database has to be chosen. Since the purpose of the experiments will be to make a statement about the correctness of the results, consensus should exist about what is a correct result and what is not. This general consensus is called *relevance judgement* or *ground truth*. The ground truth is the list of images that should be returned as retrieval results when a specific image is used as query image. When this ground truth is known, it can easily be determined whether an image is relevant or not by looking at the ground truth. The problem, however, is to determine this ground truth. Unfortunately, the CBIR-community does not have a common database with ground truth that is widely used, like TREC in information retrieval.

In [33] and [5], the relevance judgement is done by humans: for a certain query, humans select the images from the database that he or she finds most similar. However, since this is a very subjective task, no two relevance judgements will be the same. Furthermore, it is a very time-consuming job, especially for large databases. Because of the impracticality of this method, Milanese and Cherbuliez [20] developed a different (less time-consuming) method: they extracted still pictures from videoclips. It is assumed that the uninterrupted recording of a video camera implies continuity of the visual content and that camera operations and subject motions cause gradual change of content. This way, some key-frames can be captured automatically, resulting in one set of relevant images for each video clip.

Fortunately, a number of databases *with* a ground truth exist, although none of these is widely used. One of these is the image database of Washington university [3]. This database is quite extensive, freely available through the internet and an increasing number of researchers are using this database to test CBIR systems. It contains a total of 1333 images, divided over 22 classes. The images are colour images, but since all feature extraction methods are gray-scale operators, they are converted to gray-scale images. The size of the images vary; most images are 378×252 pixels, small images are 320×240 pixels, and large images are 441×294 pixels. They are images of real-life situations, with trees, bushes, rocks, landscapes, buildings, people, etc. Each image belongs to only one class, and this class is the ground truth of the corresponding image.

Another database is COIL-20, which is comparable to the Washington database, as explained in [23]. This database is actually quite often used for other purposes, namely testing classification-systems which is quite similar to CBIR, but yet very different (for classification systems, the main purpose is to determine which *class* is best suited for a certain query-image, while for CBIR systems, the main purpose is to determine which *images* are most alike). It contains 1440 images (all images are of size 128×128), divided over 20 classes, and consists of images of 20 different objects; each object is represented by 72 different images, which are grouped into one class. The difference between these 72 images is, that the object is rotated $5°$ each consecutive image. Therefore, the database contains images of all 20 objects from all around each object. Again, the corresponding class for a certain image is considered to be the ground truth of that image.

[3] `http://www.cs.washington.edo/research/imagedatabase/`

Because the COIL-20 database is used mostly in scientific research, and the Washington database consists of more "real-life" images, the first tests will be performed on the COIL-20 database. After that, the Washington database will be used to verify the results obtained by the first tests.

6.1.2 Parameters

For testing, a number of parameters have to be determined. The choices of these parameters do partly determine the performance of the feature extraction method, although a bad feature extractor can never be tweaked in such a way that it becomes a terrific feature extractor. Often, by tweaking these parameters, slightly better (or worse) performance can be obtained. It is assumed that the choices of the parameters do not affect the outcome of the experiments too much, since none of the methods are tweaked more than the others; the same set-up is used for all three methods.

Number of features

One of the first choices is the number of features that are to be determined for each image. This number should not be chosen too large, since the calculation of the dissimilarity would take too long, nor should this number be chose too small, since the features would not be descriptive enough.

For the general pattern spectra, one single area opening spectrum is determined. This opening spectrum produces all peak components that are present in the image. Since the number of peak components will differ for every image, and the goal is to produce a fixed number of features, it is not possible to simply calculate Hu's moment invariants for every peak component that was detected: a variable number of features for every image would be the result. Therefore, to ensure a fixed number of features, for a number of (fixed) sizes, the peak components are checked if they meet the specific size. If one or more peak components exists, the corresponding Hu's moment invariants are determined; if no such peak component exist, the features are set to 0 for this area. So, for every area, 7 features are determined (either Hu's moment invariants for the corresponding bin, of 7 times 0, since the number of features are to be fixed).

Now, as was explained in the previous section about the databases, the images in the Washington database have a size of about 378×252 pixels. The maximum area theoretically possible is 95256, so if Hu's moment invariants are to be determined for every possible area, a total number of $7 \times 95256 = 666792$ would be the result. Although this would be the most accurate, it is definitely not practical, especially since a peak component with an area of 95256 would only occur once: when the background is marked as peak component. To reduce the number of features, the largest peak component that is reviewed will have a size of 1% of the total area of the image (in this case: 1% of $95256 = 9525$ pixels). Besides if the number of features is too large, another phenomenon that has to be taken into account is the *curse of dimensionality* [4]. It causes systems with large feature vectors to perform relatively badly, because the dimension of the input space is high and too many features are taken into account, that do not contribute to a better performance, but *do* take up resources of the system. In other words, when using a feature vector with too many features, the performance of the systems reduces, while more resources (computing time, memory) are needed. Furthermore, the chance exists that noise in images that occur in the training set is made too important, causing *overfitting*. Overfitting is traditionally defined as training some flexible representation so that it memorizes the data but fails to predict well in the future, see [32] for a discussion of overfitting.

The maximum area is reduced dramatically, but still, if Hu's moment invariants are to be determined for every area between 0 and 9525, it would result in 66675 features, which is still way too many. The

next step to reduce the number of features, is not to determine Hu's moment invariants for every area, but for only a few area's, by taking peak components that do not differ much in area size, together. By increasing the area with 166 each time, the number of features is reduced dramatically to a number of 400.

For the spatial size distribution, an equal bin size is used, in order to correctly compare the results of both methods. For this method, an attribute opening is determined for every area between 0 and the maximum area (which will be set to 1% of the total area of the image, like before). For every attribute opening, a number of features are determined, which depends on the number of values that are used for μ. In order to use (about) the same bin size, the maximum value for μ is set to 15, and each step the value for μ is increased by 2.

For the third method, multi-scale connectivity, the same bin size should be used. This method determines one area pattern spectrum for every MaxTree; the number of features this area pattern spectrum produces can be limited the same way that was done before, by limiting the maximum area and by limiting the number of peak components by grouping peak components that do not differ much in size, together. The same bin size as before can be achieved by setting the number of iterations to 7. A number of 7 iterations, though, is not enough to recognise all peak components. For this purpose, a total number of about 25 or 30 iterations is needed. Therefore, the total number of iterations is set to 28, but the features produced by each iterations are only saved once every four iterations.

For each method, a number of 400 features is chosen to be determined for every image. Because of this number, a number of choices had to be made, which might affect the performance of the methods. Since the area was increased with a constance value, it is assumed that the performance is affected (about) equally for each method.

Distance function

The features of every method are combined in one feature vector for each image. To compare these feature vectors, a *dissimilarity* has to be determined: when two feature vectors are complete the same, the dissimilarity is 0, and the dissimilarity increases as two feature vectors are less coherent. This dissimilarity is computed using a *distance function*. Many exist, as there are many ways to compare two feature vectors. Puzicha et al. [25] give an extensive overview of widely used distance functions, in which the Earth Mover's Distance [26] (EMD) comes forward as best one, although differences are small between most functions. However, since each image is compared to every other image once, and the databases consist of 1333 and 1440 images for the Washington database and COIL-20 database, respectively, the dissimilarity should be computed very quickly, in order to efficiently test the feature extraction methods. Unfortunately, the EMD [4] is quite slow, especially for a large number of features (like 400). Because of need for efficient dissimilarity calculation and the apparent small differences between distance functions, the choice was made to use a special instance of the *Minkowski-form distance* $\mathcal{L}_p$, namely the city-block distance $\mathcal{L}_1$. The dissimilarity can be determined very fast using this function, and it is widely used as distance functions for many purposes. The city-block distance is:

$$\mathcal{L}_1\text{-distance}(\vec{X}, \vec{Y}) = \sum_{i=0}^{N} |\vec{X}(i) - \vec{Y}(i)|, \tag{6.1}$$

where N is the total number of features in each feature vector.

[4] `http://ai.stanford.edu/~rubner/emd/`

6.1.3 Performance measures

As said before, no standard performance measures exist. Many systems are evaluated using precision and recall, but the way this evaluation is presented differs greatly, and therefore, systems are hard to compare, if the evaluation was presented in different articles. In [22], a set of standard measures is proposed, which is similar to those used in TREC which have long been used in IR. The performance measures proposed by Müller et al. [22] are also based on precision and recall [36]. *Precision* is the fraction of retrieved material that is actually relevant: *Recall* is the fraction of the relevant material that is actually retrieved in answer to a search request.

$$\text{precision} = \frac{\text{number of relevant images retrieved}}{\text{total number of images retrieved}}, \tag{6.2}$$

$$\text{recall} = \frac{\text{number of relevant images retrieved}}{\text{total number of relevant images}}. \tag{6.3}$$

Precision and recall attempt to measure the effectiveness of a retrieval system; it is a measure of the ability of the system to retrieve relevant material while at the same time holding back non-relevant material.

The set of performance measures proposed by Müller et al. [22] contains a mix of different measures. Some are rank-based (performance measures 1 and 2), others are single-valued (measures 3 – 7) and yet others are graphical measures (measure 8).

1. Rank_1 – the rank at which the first relevant image is returned;
2. $\widetilde{\text{Rank}}$ – normalised average rank of relevant images: equation (6.4);
3. $P(20)$ – precision after the first 20 images are retrieved;
4. $P(50)$ – precision after the first 50 images are retrieved;
5. $P(N_R)$ – precision after the first N_R images are retrieved, where N_R is the number of relevant images for a given query;
6. $R_P(0.5)$ – recall at the rank where precision drops below 0.5;
7. $R(100)$ – recall after 100 images are retrieved;
8. PR graph – a precision vs. recall graph, which shows how precision decreases as increasingly large fractions of the material are retrieved.

The normalised average rank of relevant images can be calculated as follows:

$$\widetilde{\text{Rank}} = \frac{1}{NN_R}\left(\sum_{i=0}^{N_R} R_i - \frac{N_R(N_R-1)}{2}\right), \tag{6.4}$$

where R_i is the rank at which the ith relevant image is retrieved, N is the total number of images and N_R is the number of relevant images for a given query. When $\widetilde{\text{Rank}} = 0$, the performance is perfect and as it approaches 1 the performances worsens.

Assessment measure

These performance measures are all based on precision and recall. Although the precision and recall are widely used measures for performance evaluation, they have several shortcomings. One of these shortcomings is the great dependence on the cutoff number (e.g. the total number of images being retrieved, the denominator in equation (6.2)); the choice of this number is quite arbitrary and is not easy to make. For small values, the performance might drop because relevant images that are ranked quite high might be discarded as they are below the cutoff number, while for large values none of the relevant images might be discarded, causing the inability of the performance measures to distinguish between many relevant images in the higher or in the lower ranks.

Another shortcoming of these measures is that precision and recall are inversely correlated, which makes is hard to compare systems on multiple numbers. Further more, since precision and recall are dependant on the query and the data-collection as well as the relevance judgement, simple comparison between the measures obtained in different environments is impossible.

To overcome these shortcomings (and others), Dimai [8] proposed the *rank difference trend analysis* (RDTA). The basic idea is to statistically compare the ranking of relevant data obtained by two retrieval algorithms (in this case: two feature extraction algorithms). From this comparison, the following assessment measure is added to the set of performance measures (as was proposed in the previous section):

9. $\mu(A, B)$ - expresses the relative discrimination of A with respect to B. When $\mu(A, B) < 0$, then A is more effective than B, while $\mu(A, b) < 0$ states that A is less effective than B.

The assessment measure μ can be obtained by a statistical comparison of two rank sets. This comparison will only make sense when the two rank sets where acquired in the same environment.

The rank sets of two retrieval methods can be generated fully automatically in three steps:

1. Both algorithms are used to generate ranked lists of retrieved data for a given query image.
2. The ranks of each relevant image (belonging to the given query) are determined using the ranked lists generated in the previous step.
3. These ranks are summarised to one rank set for each algorithm for a given query image.

These three steps are executed for a number of N query images and the rank sets for each query image are bundled in one rank set for algorithm A and one for algorithm B, $\{r_{A_i}\}_{i=1...N}$ and $\{r_{B_i}\}_{i=1...N}$ respectively.

The statistical comparison between two rank sets $\{r_{A_i}\}_{i=1...N}$ and $\{r_{B_i}\}_{i=1...N}$ is done by *linear regression*: $r_{A_i} = b \times r_{B_i}$, e.g. algorithm A performs b times worse than algorithm B. One important note to be made, is that for the evaluation, relevant data with high ranks are less important that relevant data with low ranks. For this purpose, weights are introduced by associating variances to each rank: $\sigma_{A_i} = a \times r_{A_i}$ and $\sigma_{B_i} = a \times r_{B_i}$.

To solve the linear regression model, the chi-square measure is used as maximum likelihood estimator. The χ^2 merit function is:

$$\chi^2(b) = \sum_{i=1}^{N} \frac{(r_{A_i} - r_{B_i})^2}{\sigma_{A_i}^2 + b^2\sigma_{B_i}^2}. \tag{6.5}$$

This equation should be minimised with respect to b. Because of the occurrence of b in the denominator, this has to be done numerically.

The performance measure μ will have to be antisymmetric, i.e. $\mu(A, B) = -\mu(B, A)$, and range between $[-1, 1]$. Therefore, the following parameterisation is chosen:

$$\mu = \frac{4}{\pi} \times \arctan(b) - 1. \tag{6.6}$$

Because of the non-linearity in equation (6.5), the standard errors of μ are not determined by a simple error propagation (the standard errors enables one to determine whether performance differences are significant or not). To obtain stable standard errors, it is considered that $\chi^2(\mu)$ is minimised by μ_0: $\chi^2_{\text{min}} = \chi^2(\mu_0)$. With this assumption, it can be seen that $\Delta\chi^2 = \chi^2 - \chi^2_{\text{min}}$ is distributed as a chi-square distribution with the dimensionality of μ (1 in this case) as degree of freedom. Therefore, to obtain a confidence level of 68.3%, the root of the problem $\Delta\chi^2(\mu) = 1$ has to be determined (which is done numerically as well).

6.1.4 Experiments

The methods are all tested on both databases and the same setup is used in every experiment. For both databases, three feature sets are determined for each image (each feature extraction method produces one feature set). Each feature set of a specific method is compared to every other feature set of that same method in the same database. Next, for each feature set, a rank list is being generated using the distance function that was described earlier, see equation (6.1), to determine the distance between two feature sets. This rank list contains ranks of other feature sets, sorted by the dissimilarity (lowest dissimilarity ranks highest). This way, each image in the database is used as query image for each feature extraction method once, and compared to every other image in the corresponding database. Since the performance measures described earlier require a distinction between relevant images and non-relevant images, the rank list, as well as the ground truth of the specific query, can be used to determine these automatically. The calculation of the performance measures can be fully automated too, so the whole experiment can be automatated. Therefore, all images will be used as query image once so a complete picture of the performance will be acquired, and the bias some methods might have to certain query images is eliminated this way.

6.2 COIL-20

The first experiments are performed on the COIL-20 database. In this section, the results will be presented for all three feature extraction methods. First, all methods will be discussed separately, by looking at the raw results first and discussing these results in more detail after that. Second, an extension is being discussed, concerning histograms. More details about this is given in section 6.2.5. The results of this extension are also discussed here. Finally, the three methods are compared and this comparison is being discussed.

6.2.1 Generalised pattern spectra

The first experiments are done using the generalised pattern spectra. For every image, 400 features are extracted. All images are 128 wide and 128 high, so 400 features corresponds to a bin size of about 30 pixels.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	18.2%	29.5%	37.0%	43.0%	47.6%	50.8%	54.0%	56.7%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	0%	0%	5.0%	10.4%	20.3%	25.6%	31.2%	42.4%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	0%	0.2%	0.8%	2.0%	3.3%	4.8%	7.5%	13.3%
$P(50)$	0%	0%	0%	0.1%	0.3%	0.8%	2.4%	7.2%
$P(N_R)$	0%	0%	0%	0%	0%	0.1%	1.0%	3.5%
$R_P(0.5)$	0%	0%	0%	0%	0%	0.1%	0.3%	0.7%
$R(100)$	0%	0%	0%	0%	0.1%	0.4%	2.1%	12.4%

Table 6.1: The performance measures of the generalised pattern spectra, tested on the COIL-20 database.

Results

Every image in the database is used as query image once, and for all the produced rank lists, the performance measures discussed in chapter 6.1.3 are being computed. Since a total number of 1440 times these performance measures will have to be computed, it is impossible to present all of them seperately. Therefore, a different method to present the results is used. The first performance measure, Rank_1, will ideally produce 1 as result, and as the performance drops, this value increases. The results are presented by calculating the percentage of the queries that produced a value for Rank_1 of ≤ 1, ≤ 2, ≤ 3, ≤ 4, etc.

The second performance measure, $\widetilde{\text{Rank}}$, produces values between 0 and 1; as the performance increases, the value for this measure drops to 0. The results are presented, by calculating the percentage of the queries that produced a value for $\widetilde{\text{Rank}}$ of ≤ 0.05, ≤ 0.1, ≤ 0.15, ≤ 0.2, etc.

The third, fourth, fifth, sixth and seventh performance measures, $P(20)$, $P(50$, $P(N_R)$, $R_P(0.5)$ and $R(100$ respectively, produce values between 0 and 1 as well, but as the performance for these measures increases, the values rises to 1. The results are therefore presented as the percentage of the queries that produced a value for these measures of ≥ 0.9, ≥ 0.8, ≥ 0.7, ≥ 0.6, etc.

In table 6.1, the results are shown for the generalised pattern spectra. In figure 6.16, the precision-recall-graph is shown (the solid line belongs to the generalised pattern spectra). The average precision and recall of all queries is used to determine this graph.

In this table it can be seen that over 18% of the queries (corresponding to about 262 out of the 1440 queries) the first returned results was an image that belongs to the same class as the query-image; in 29.5% the first returned image belonging to the same class as the query-image was found in the first or second image and for over 50% of all queries, the first correct result was found within the first 6 returned images.

The next performance measure is $\widetilde{\text{Rank}}$. 5% of the queries produced a value for this measure that is lesser or equal to 0.15, while about a quarter of all queries resulted in a value of 0.3 or less.

The precision-results are shown next; the precision of the first 20 results for each query is less than 0.8 for more than 99% of all queries. The precision for the first N_R results for each query (where N_R

(a) (b) (c)

Figure 6.1: Three images, that are used as query-image, to test the generalised pattern spectra. The Duck (a) is used as image that produces bad results, the vehicle (b) produces average results and the Tub (c) is retrieved quite well.

is 72 in this database for all images) is even worse: for 99% of the queries this value is less than 0.4, while the precision for the first 50 results for each query is in between these two values: for more than 99% the precision is less than 0.5.

Finally, the recall-results show, that for only 0.1% of the queries a recall for the first 100 results ($R(100)$) of 0.6 is obtained. Furthermore, less than 1% of the queries obtain a recall of 0.3 when the precision drops below 0.5.

Discussion

To present a more concrete picture of the results that are obtained, three images were hand-picked, to be reviewed more closely: one that produced bad results, according to the performance measures (a duck, figure 6.1(a)), one with average performance (a vehicle, figure 6.1(b)) and one that produced very good results (a tub, figure 6.1(c)). For these images, some results are shown, that were produced by the image retrieval experiment. Furthermore, the performance measures for these images are shown. After that, these results, as well as the ones presented in the previous section, are discussed.

The first image that is shown is the duck, see figure 6.1(a). When this image is used as query image (using the features produced by the generalised pattern spectra), the first 10 images that are produced can be seen in figure 6.2. Remarkably, 8 out of the 10 displayed results are from the same class (class 16). The first returned image that belongs to the same class as the query image is on place 209. The performance measures belonging to this query-image are in table 6.2.

The second image is the vehicle, figure 6.1(b). The first 10 results are in figure 6.3. Four of these images are correct; the on position 10 is a car from another class, so it is not marked as a relevant image. In table 6.2, the performance measures are shown.

The third image is the tub, figure 6.4. Out of the first 10 results (see figure 6.4), 5 belong to the same class as the query image, while 2 other images contain an object with somewhat similar texture on its

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	209	0.7174	0	0	0	0	0
Vehicle	5	0.2910	0.2	0.1	0.0845	0	0.0845
Tub	1	0.1189	0.7	0.4	0.3239	0.0141	0.3944
Pipe	87	0.2111	0	0	0	0	0.0423
Stone_jar	2	0.5919	0.2	0.1	0.0704	0	0.0704
Cup	54	0.6868	0	0	0.0141	0	0.0141
Racecar	3	0.2288	0.15	0.18	0.1690	0	0.2113
Duck	16	0.7041	0.05	0.02	0.0141	0	0.0282

Table 6.2: The performance measures for the queries that were hand-picked in this section; the other images are chosen in sections 6.2.2 and 6.2.3 as queries that characterise the spatial size distributions and the multi-scale connectivity, respectively.

label. Again, in table 6.2, the performance measures are shown.

When looking at the results more closely, it can be seen that images with smooth surfaces without texture are harder to recognise than images *with* texture on it. Images from classes 9, 16 and 20 produce significantly better results than images from other classes. In figure 6.5 one example can be seen from each of these classes. This might have something to do with the fact that the maximum size of a peak component is set to be 1% of the total area of the image. Large structures, like the overall shape of an object, are being ignored because of this. This is not tested in this thesis, however, and can therefore not be stated as a certainty.

As one can see, the objects of these images have labels with text on them. One explanation of this might be, that the generalised pattern spectra are very sensitive to illumination effects. The effect of the illumination on the objects that are displayed in figure 6.5 is less than on most of the other objects.

6.2.2 Spatial size distribution

The second experiments are done using the spatial size distribution. The same bin size is used as for the generalised pattern spectra.

Results

The results for this methods are obtained and presented the same way as for the generalised pattern spectra.

In table table 6.3, the results are shown, and in figure 6.16, the precision-recall-graph is shown (the dotted line belongs to the spatial size distributions). The average values for the precision and recall are used.

The number of times the first returned results was an image belonging to the same class as the query image is 633 out of 1440 (44%). Although this result appear to be better than the generalised pattern spectra, the other performance measures show otherwise: over 90% of the queries have a $\widetilde{\text{Rank}}$ higher than 0.35. Furthermore, none of the queries produce a value for $P(N_R)$ and $R_P(0.5)$ higher than 0.3.

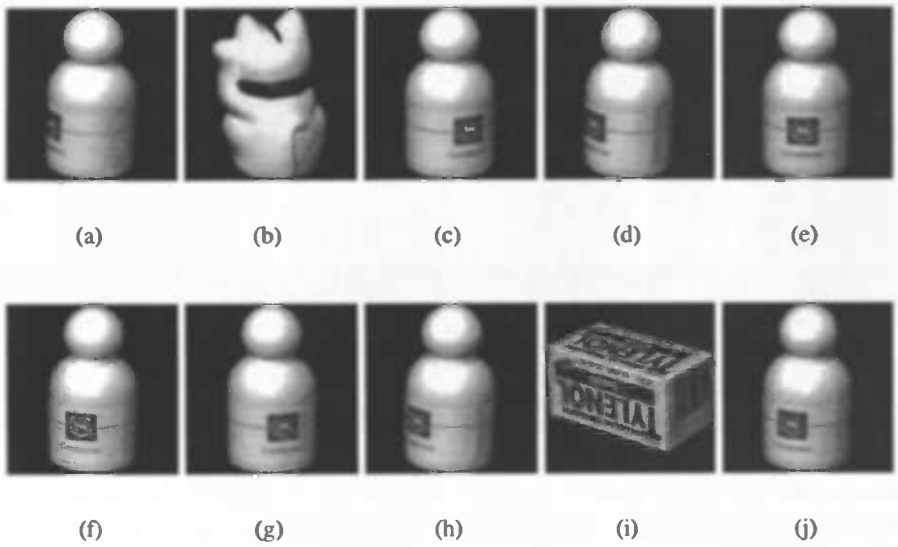


Figure 6.2: The first 10 images that were retrieved by the generalised pattern spectra when the image in figure 6.1(a) was used as query. The canister (class 16) was retrieved eight out of ten times, the other images are from class 4, image (b), and 9, image (i), respectively.

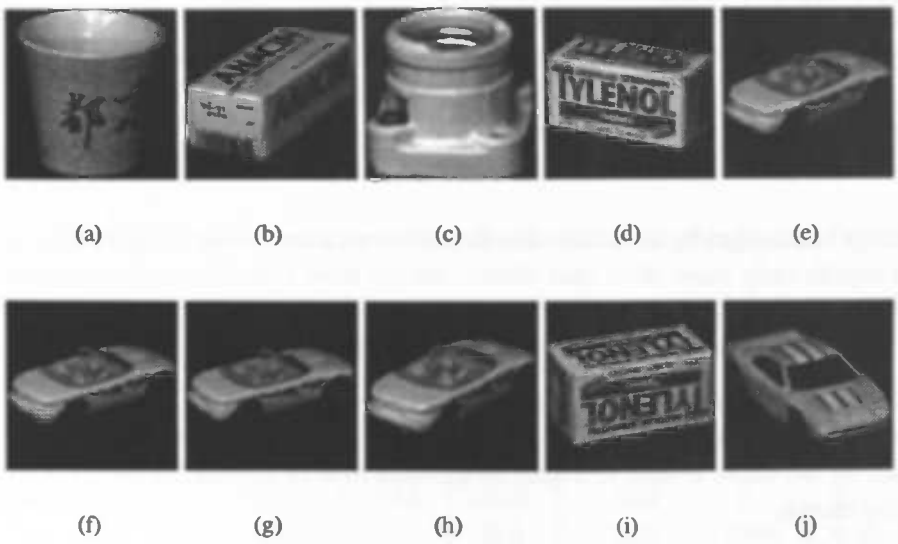


Figure 6.3: These images are retrieved by the generalised pattern spectra when image 6.1(b) is used as query. Four images are from the ground truth of the query-image, images (e), (f), (g) and (h). The other images are from class 12, image (a), class 5, image (b), class 14, image (c), class 9, images (d), and (i) and class 3, image (j).

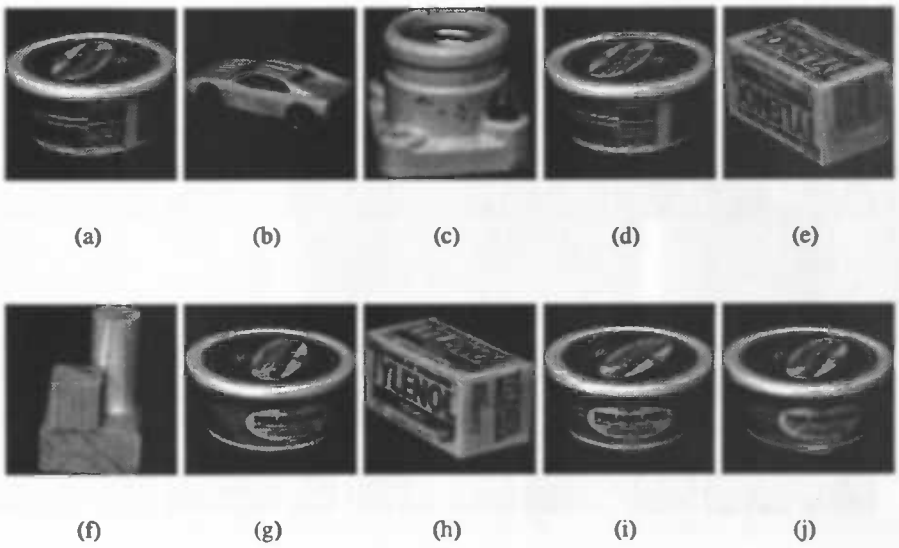


Figure 6.4: These images are retrieved by the generalised pattern spectra, when using the image in figure 6.1(c) as query-image. Five out of the first ten results are relevant images (images (a), (d), (g), (i) and (j)). The other retrieved images are from the classes 3, image (b), 14, image (c), 9, images (e) and (h), and 7, image (f).

Discussion

Three images were hand-picked again, to show some results for this method. Furthermore, the performance measures for the three images that were chosen for the generalised pattern spectra are also computed and shown for this method. The images that are hand-picked are a pipe, a stone jar and a cup, see figure 6.6.

The first image is the pipe, figure 6.6(a). The first 10 results are shown in figure 6.7. No specific class is returned significantly more often than others: the tub from class 20 is returned three times, class 7 is returned twice and the other classes (class 3, 5, 9, 13 and 19) are only present once in the first 10 results. In table 6.4, the performance measures can be seen.

In figure 6.6(b), the second image, producing average results (see figure 6.8), can be seen. Besides the correct class, which is returned three times, the duck from class 1 is returned four times. This might be explained by the round shapes and smooth surfaces in both objects. In table 6.4, the performance measures are shown.

The third image, the cup, is in figure 6.6(c). The first five results that were returned all belong to the correct class, as well as the ninth returned image (see figure 6.9). In table 6.4, the performance measures are shown.

In table 6.4, the performance measures for the three images chosen for the generalised pattern spectra in section 6.2.1 can also be seen, while in table 6.2 the performance measures for the three images chosen in this section are shown. Compared to the results of the generalised pattern spectra, the spatial

(a) (b) (c)

Figure 6.5: These are the images of the objects that are retrieved well by the generalised pattern spectra. The objects are from classes 9, 16 and 20, respectively. Notice that a label with text of some kind exists on each of these objects.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	44.0%	55.1%	62.0%	66.6%	69.7%	72.4%	74.8%	76.3%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	0%	0%	0%	0%	0%	2.4%	8.7%	25.6%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	0%	0%	0%	0.3%	1.2%	2.9%	7.9%	16.0%
$P(50)$	0%	0%	0%	0%	0%	0%	0%	2.4%
$P(N_R)$	0%	0%	0%	0%	0%	0%	0%	0%
$R_P(0.5)$	0%	0%	0%	0%	0%	0%	0%	0%
$R(100)$	0%	0%	0%	0%	0%	0%	0%	0.8%

Table 6.3: The performance measures of the spatial size distributions, tested on the COIL-20 database.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	3	0.4582	0.05	0.02	0.0141	0	0.0141
Vehicle	1	0.3724	0.2	0.22	0.1690	0.0141	0.1831
Tub	1	0.5563	0.2	0.14	0.1127	0.0282	0.1268
Pipe	153	0.6502	0	0	0	0	0
Stone_jar	2	0.3946	0.15	0.06	0.0563	0	0.0704
Cup	1	0.2540	0.5	0.3	0.2535	0.0986	0.2958
Racecar	15	0.3775	0.1	0.08	0.0704	0	0.0986
Duck	1	0.4567	0.15	0.14	0.1127	0.0423	0.1408

Table 6.4: The performance measures for the queries that were hand-picked in this section; the other images are chosen in sections 6.2.1 and 6.2.3 as queries that characterise the generalised pattern spectra and the multi-scale connectivity, respectively.

(a) (b) (c)

Figure 6.6: Three images that are used as query-image to test the spatial size distributions. The Pipe (a) is used as image that produces bad results, the stone jar (b) produces average results and the Cup (c) is retrieved quite well.

size distribution produces four times a worse result. One remark can be made about the spatial size distributions: the results fluctuate less than the results of the generalised pattern spectra, which have some very good results *and* some very bad ones, where the results of the spatial size distribution are all slightly better or worse than average. This is actually a useful characteristic, since a user will be able to more or less *predict* the outcome of a certain query. It is better for a system to perform average all the time, than very good sometimes and very bad other times.

This method produces better results when images from classes 12, 16 and 18 (see figure 6.10 for an instance of each class) are used. The objects in these images have round shapes and smooth surface.

Another thing that can be noticed, is that texture is actually recognised quite well: when a certain image is used as query, the images that are returned have a similar texture, e.g. when the query-image has a smooth surface, most of the returned results will have smooth surfaces as well.

6.2.3 Multiscale connectivity

Next, the multi-scale connectivity method is tested. Again, the same bin size is used as the other two methods.

Results

The results are obtained and presented the same way as before for the spatial size distribution and the generalised pattern spectra. The results are shown in table 6.5 and figure 6.16 (the dashed line belongs to multiscale connectivity).

These results actually appear to be quite good. Over 93% of the queries produced an image of the same class as first returned result, while almost 40% produced a value for $\widetilde{\text{Rank}}$ of at most 0.05. Furthermore, for 43% of the queries, the precision after 20 returned results is 100%, meaning that all

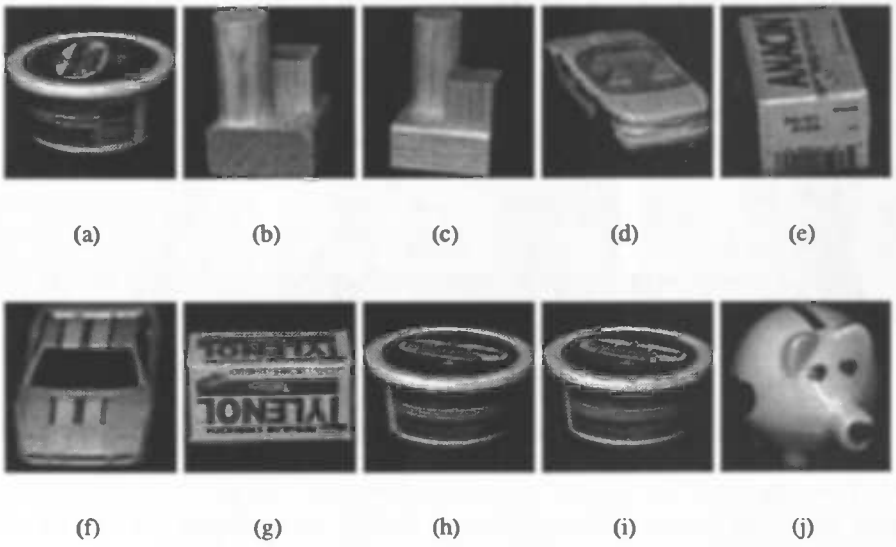


Figure 6.7: These images were retrieved by the spatial size distributions when the Pipe in image 6.6(a) was used as query-image. The retrieved images do *not* belong to the same class; they are from classes 20, images (a), (h) and (i), class 7, images (b) and (c), class 19, image (d), class 5, image (e), class 3, image (f), class 9, image (g), and class 13, image (j).

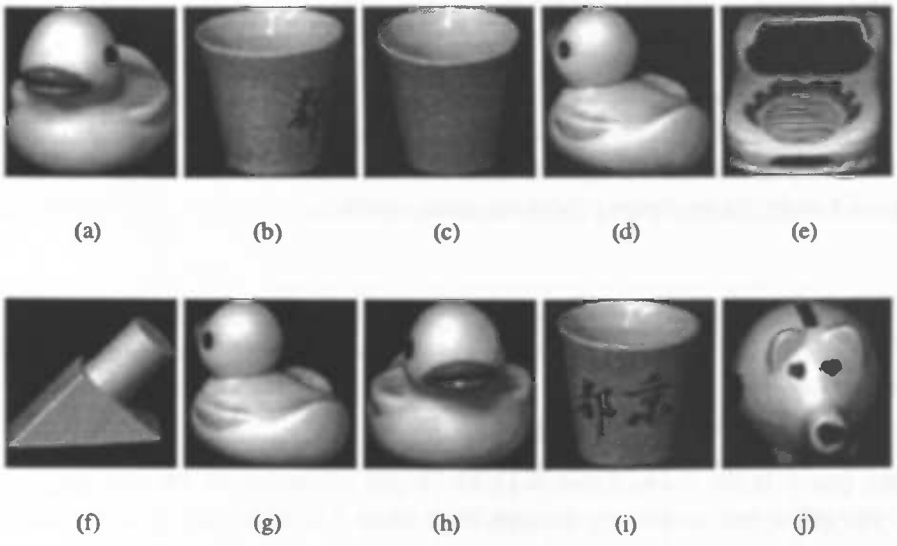


Figure 6.8: The image in figure 6.6(b) was used as query-image to produce these results, obtained by using the spatial size distributions for retrieval. Three images belong to the ground truth, images (b), (c) and (i). The other images belong to classes 1, images (a), (d), (g) and (h), class 6, image (e), class 2, image (f), and class 13, image (j).

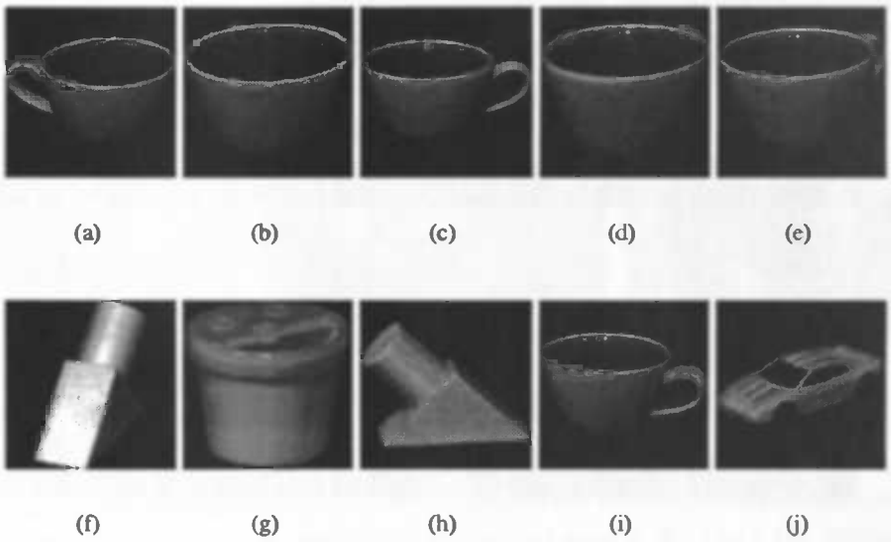


Figure 6.9: When the image in figure 6.6(c) is used as query and the spatial size distributions for retrieval, the first 5 retrieved images are from the ground truth of the query-image, as well as image (i). The other images are from class 2, images (f) and (h), class 15, image (g), and class 3, image (j).

20 returned results belonged to the same class as the query image. In 6.1% of the queries, all relevant images are returned before any other image is returned.

Discussion

Finally, for the multi-scale connectivity method, again three images are hand-picked. Also, the performance measures are calculated for the images chosen in the previous sections, as well. The images are a race car, a duck and the tub again (which was also discussed for the generalised pattern spectra in section 6.2.1 as being an image producing good results.

The first image, figure 6.11(a), produces some remarkable results (see figure 6.12): nine out of the first ten returned results belong to class 13 (the piggy-bank). No apparent explanation for this is found at this point; maybe it has something to do with the stripes of the racecar in image 6.11(a) and the slit of the piggy-bank in which the money goes.

The second image is the duck, figure 6.11(b). Eight times out of the first ten, a correct image is retrieved; the other two results are images from class 11, which has a curvature at the top of the object.

The final images is the same image that was used to show good results for the generalised pattern spectra: the tub in figure 6.11(c). All images form the correct class were retrieved first, before any other image was returned. Hence, the first ten retrieved images in 6.14 all belong to the correct class.

In table 6.6, the performance measures for these three images are shown.

(a) (b) (c)

Figure 6.10: These are objects that are retrieved well by the spatial size distributions. They are objects from class 12, 16 and 18, respectively. Notice the smooth surfaces and the round shapes of these objects.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	93.3%	95.6%	96.9%	97.2%	97.5%	97.8%	97.8%	98.1%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	39.9%	52.4%	59.7%	69.2%	78.0%	85.3%	90.2%	93.7%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	43.0%	51.1%	56.6%	63.9%	69.4%	76.2%	82.4%	88.1%
$P(50)$	14.5%	21.7%	30.6%	39.6%	50.5%	58.1%	65.7%	75.2%
$P(N_R)$	6.1%	15.1%	19.7%	26.2%	36.3%	47.6%	57.8%	68.7%
$R_P(0.5)$	19.8%	24.4%	28.4%	34.6%	41.1%	47.4%	54.0%	59.2%
$R(100)$	16.3%	21.0%	25.8%	35.2%	45.3%	53.5%	63.8%	74.8%

Table 6.5: The performance measures of the multi-scale connectivity method, tested on the COIL-20 database.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	1	0.0877	0.9	0.6	0.4930	0.4930	0.5493
Vehicle	1	0.4257	0.25	0.1	0.0986	0.0423	0.1268
Tub	1	0.0007	1.0	1.0	1.0	1.0	1.0
Pipe	1	0.1647	0.5	0.32	0.2817	0.1268	0.3521
Stone_jar	1	0.0080	1.0	0.84	0.7746	1.0	0.9155
Cup	1	0.0195	0.9	0.64	0.5915	0.8732	0.7183
Racecar	63	0.4938	0	0	0.0141	0	0.0141
Duck	1	0.0562	0.65	0.6	0.4930	0.4648	0.5775

Table 6.6: The performance measures for the queries that were hand-picked in this section; the other images are chosen in sections 6.2.1 and 6.2.2 as queries that characterise the generalised pattern spectra and the spatial size distributions, respectively.

(a) (b) (c)

Figure 6.11: Three images, that are used as query-image, to test the multi-scale connectivity method. The Racecar, image (a) is used as image that produces bad results, the Duck (b) produces average results and the Tub (c) is retrieved quite well. Notice that the image of the Duck that is used this time is different from the one that was used for the generalised pattern spectra, while the Tub is the same image.

In table 6.6, the results of the performance measures for the multi-scale connectivity method for the images chosen in the previous sections can be seen. In table 6.2 and table 6.4, the results of the performance measures for the generalised pattern spectra and spatial size distributions, respectively, for the images chosen in this section can be seen. The multi-scale connectivity produces better results than the other two methods, except for the racecar.

When queries are images from class 1, 12, 13, 14, 16, 17, 18 or 20, the results are very good (see figure 6.15 for an instance of each of these objects).

No apparent tendency for texture or smooth surfaces follows from these classes, but what *can* be seen, is that these classes contain objects that have a certain curvature. When images with objects that consist of straight lines are used as query-image, the performance is not as good as when objects with a certain curvature are used. However, this might as well be just a coincidence, since there is no apparent explanation for this fact.

6.2.4 Overview

From the results so far, it can be seen that the multi-scale connectivity method produces that best results. Next comes the generalised pattern spectra. Although, on average, the results for this method are similar to the spatial size distribution, I think the generalised pattern spectra produce better results, since some queries perform very well, approaching the results of the multi-scale connectivity method. The results of the spatial size distributions are more close together. In figure 6.16, the precision-recall-graphs for the three methods are shown in one graph. From this, it is confirmed that the generalised pattern spectra and the spatial size distribution are close together. However, for low precision-values (< 0.1), the recall for the spatial size distribution is higher. Overall, the multi-scale connectivity are by far the best.

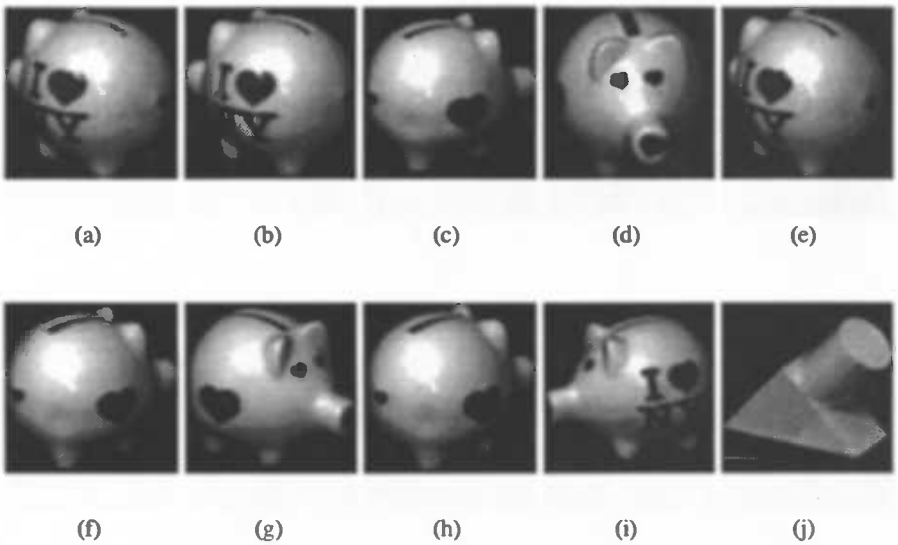


Figure 6.12: These images are retrieved by the multi-scale connectivity method when using the image in figure 6.11(a) as query-image. The first nine retrieved images are all from class 13, image (j) is from class 2.

Each of the three methods have different classes they recognise best: both the generalised pattern spectra and the spatial size distributions have a certain tendency towards textures. The generalised pattern spectra perform well on images from classes 9, 16 and 20 (figure 6.5, while the spatial size distribution perform well on images from classes 12, 16 and 18 (figure 6.10). The difference between the two is, that the generalised pattern spectra can recognise the presence of a certain texture on any object, while the spatial size distribution can recognise texture on, especially, round objects.

The third method, multi-scale connectivity, can recognise curved objects especially well. Images from classes 1, 12, 13, 14, 16, 17, 18 and 20 all are curved in a specific way, see figure 6.15. Most of the other objects contain straight lines, which are apparently harder to distinguish by this method. This might be a coincidence, since I have not found a satisfying explanation for this.

6.2.5 Histograms

Results for especially the generalised pattern spectra and spatial size distribution are not good; the multi-scale connectivity method is, by far, the best method. This might be caused by the fact, that the bin size is too large. Although the bin size is (about) the same for all methods, one might be more sensitive to a too large bin size than others. If the bin size would be set to be 1, though, the number of features would be far too large (> 100000) to be of any use, even when the maximum area of a peak component is 1% of the total area of the image (which is still used while determining these histograms). Therefore, in order to reduce this bin size to 1 anyway, a method for reducing the number of features is needed. One of the simplest is the use of histograms (this good performance this method produced, as is shown in the next sections, was actually found heuristically); some other methods are Principal Component Analysis (PCA), [16], and Multi-Dimensional Scaling (MDS), [44]. For a certain choice of binning, all calculated features are assigned a specific bin according to a predefined

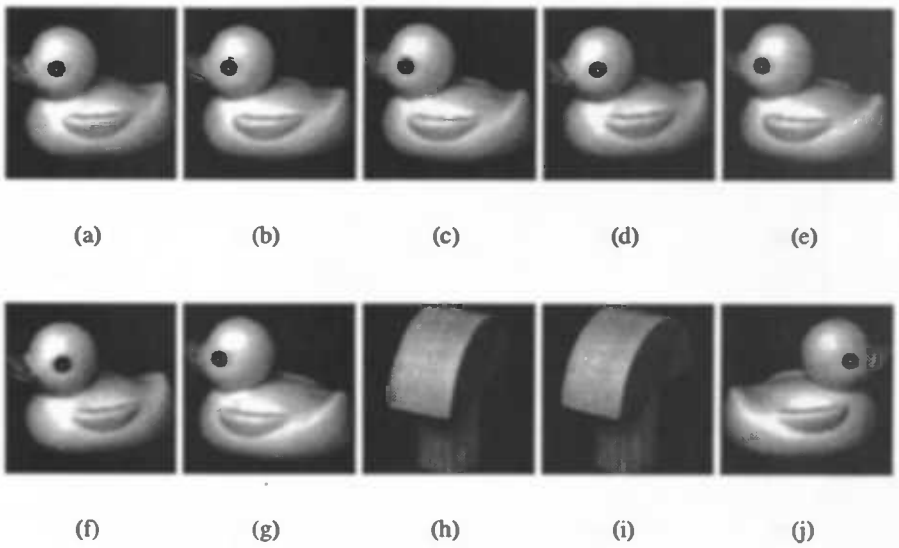


Figure 6.13: When using image 6.11(b) as query-image and multi-scale connectivity for retrieval, the first seven images that are returned belong to the ground truth of the query-image, as well as image (j). The other images, image (h) and (i), are from class 11.

mapping; all bins correspond to a certain interval, and every time a feature lies within a certain interval, the bin corresponding to this interval is incremented by one.

In this section, all three methods are tested with the histograms. The performance measures are shown for all queries as before by computing the percentage of the queries that produced a value for a specific performance measure above or below a certain value. After that, the images that were hand-picked before are now used again, to show some concrete results.

Generalised pattern spectra

The generalised pattern spectra compute seven features for each step (the seven moment invariants of Hu). These moment invariants are put together in the histogram; they are all mapped onto the same histogram. By far the most moment invariants lie between $|10|$ and $|10^{-23}|$. A logarithmic mapping is used, because the number of features should be limited and the features should be divided uniformly over the histogram.

Now, when computing the histograms for all the images using the generalised pattern spectra and a bin size of 1, the number of bins of the histograms should be determined. Since previously, a number of 400 was used, this will be used for the number of bins too; the number of features, thus, still is 400, but can easily be reduced by taking 2, 3 or more bins together. For now, 400 bins will be used, resulting in 400 features.

In table 6.7 the performance measures are shown when all the images are used as query-image once, and in figure 6.17, the PR-graph is shown, using average values for the precision and recall. In this graph, bot the original version (the solid line), as well as the histograms version (the dotted line) is shown.

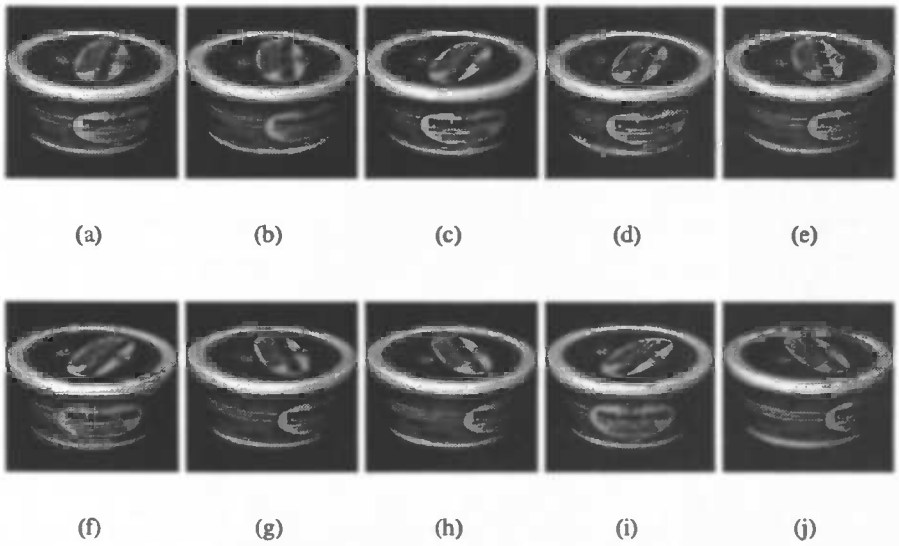


Figure 6.14: All these images belong to the ground truth of the query-image as in figure 6.11(c), *and* are retrieved before any other image is retrieved, when using the multi-scale connectivity method.

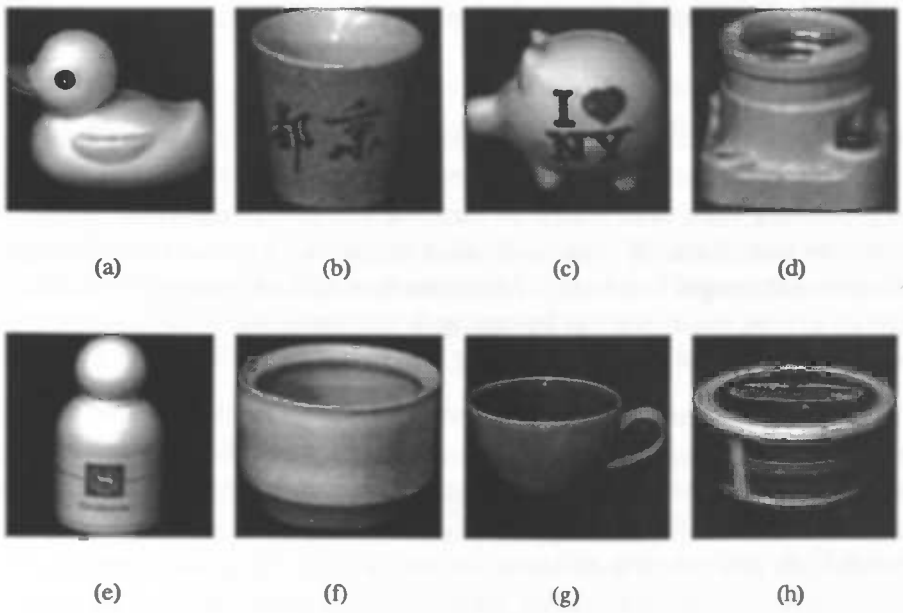


Figure 6.15: Images from these classes are retrieved very well by the multi-scale connectivity method. They are objects from classes 1, 12, 13, 14, 16 17, 18 and 20, respectively. Notice the smooth surfaces and curvatures of the objects.

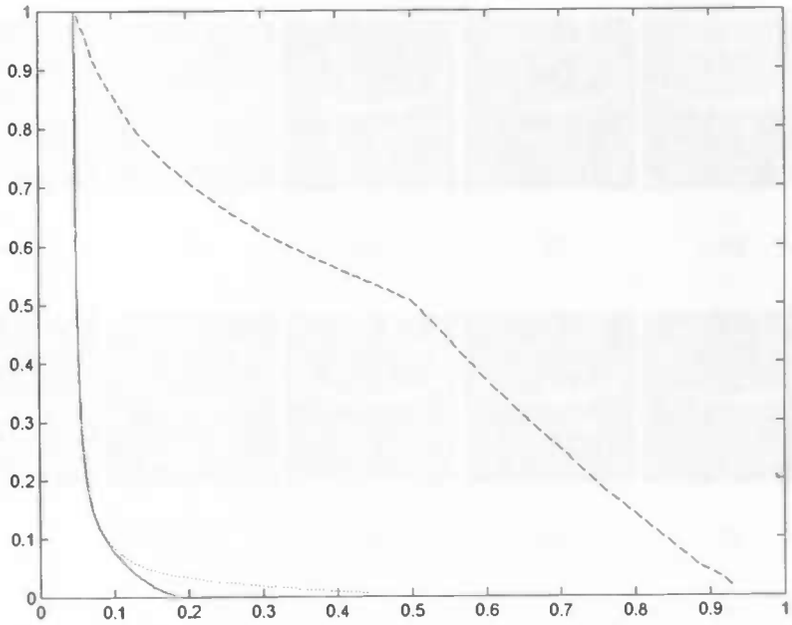


Figure 6.16: The precision-recall-graph for the generalised pattern spectra (the solid line), the spatial size distributions (the dotted line), and the multi-scale connectivity method (the dashed line), tested on the COIL-20 database. All queries are used to determine this graph, by taking the average precision and recall values.

Compared to the results in table 6.1, these results are very good: over 95% of the queries retrieved a relevant image within the first two results (against only 55.1% for the original version). Over half of the queries produces a value of at most 0.15 for $\widetilde{\text{Rank}}$ and 314 queries (22.2%) retrieved 20 relevant images within the first 20 results. The recall when the precision drops below 0.5 *and* the recall after the first 100 retrieved images is at least 0.7 for more than 20% of the queries. Further more, from the precision-recall-graphs, the dotted line belonging to the histograms version is always *above* the solid line belonging to the original version, indicating that the first performs better than the latter.

The eight hand-picked images also produced very good results for this method, see table 6.8.
Only the vehicle and racecar produced slightly worse results, all the other results are better than the original version of the generalised pattern spectra. When looking at the separate classes, class 16 and 20 perform well (just as for the original form). Class 9, however, does not perform just as well, where classes 14 and 17 *do* perform very well.

Spatial size distribution

The mapping that is used when computing the histograms is a logarithmic approach, as for the generalised pattern spectra, for the same reason. Also, the number of features (and thus, the number of bins) is set to 400. Looking at table 6.9, the performance measures can be seen for this method.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	90.5%	95.3%	96.0%	97.1%	97.8%	98.1%	98.2%	98.5%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	20.8%	39.7%	50.7%	62.7%	71.8%	76.3%	84.2%	90.8%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	22.2%	27.9%	35.8%	45.1%	52.8%	61.7%	71.9%	81.8%
$P(50)$	10.4%	16.9%	19.1%	21.7%	27.4%	38.2%	50.1%	66.9%
$P(N_R)$	0%	9.7%	16.3%	19.1%	21.7%	27.4%	38.2%	50.1%
$R_P(0.5)$	10.1%	16.8%	18.1%	20.1%	22.0%	25.6%	32.4%	40.2%
$R(100)$	6.1%	15.6%	17.9%	20.1%	24.2%	33.7%	48.8%	66.3%

Table 6.7: The performance measures of the generalised pattern spectra, with the histograms extension, tested on the COIL-20 database.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	1	0.1616	0.25	0.22	0.2113	0.0563	0.2958
Vehicle	4	0.4143	0.15	0.14	0.1127	0	0.1268
Tub	1	0.0016	1.0	1.0	0.9155	1.0	1.0
Pipe	1	0.1030	0.75	0.4	0.2958	0.2535	0.3380
Stone_jar	1	0.0558	0.95	0.62	0.5493	0.5775	0.5916
Cup	1	0.0754	0.7	0.44	0.3521	0.2535	0.4366
Racecar	4	0.3583	0.05	0.06	0.0563	0	0.0845
Duck	1	0.1386	0.5	0.31	0.2535	0.1268	0.3239

Table 6.8: The performance measures for the queries that were hand-picked in the previous sections, retrieved by the generalised pattern spectra, extended with the histograms, as explained in this section.

Apparently, there are no (large) differences between the two version. The same can be seen when looking at the performance measures for the hand-picked queries. These are not shown here, since they do not provided valuable extra information: some queries performed better, others worse. Overall, it seems, there are no major differences between these two methods.

Multiscale connectivity

When constructing the histograms for this method, a linear mapping is used, since the values of the features are much closer together. The histogram consists of 400 bins again, resulting in 400 features. For this method, the performance actually dropped: first, more than 95% of the queries retrieved a relevant image within the first two results, while these histograms only come to the same result for less than 80% of the queries. Furthermore, all the other performance measures show a similar outcome: when using the original version, more queries produced very good results for the performance measures than the version with the histograms, see table 6.10.

Because these results are quite obvious, the performance measures for the hand-picked queries are omitted. These do not provide any extra information, except that they confirm the results that are obtained so far.

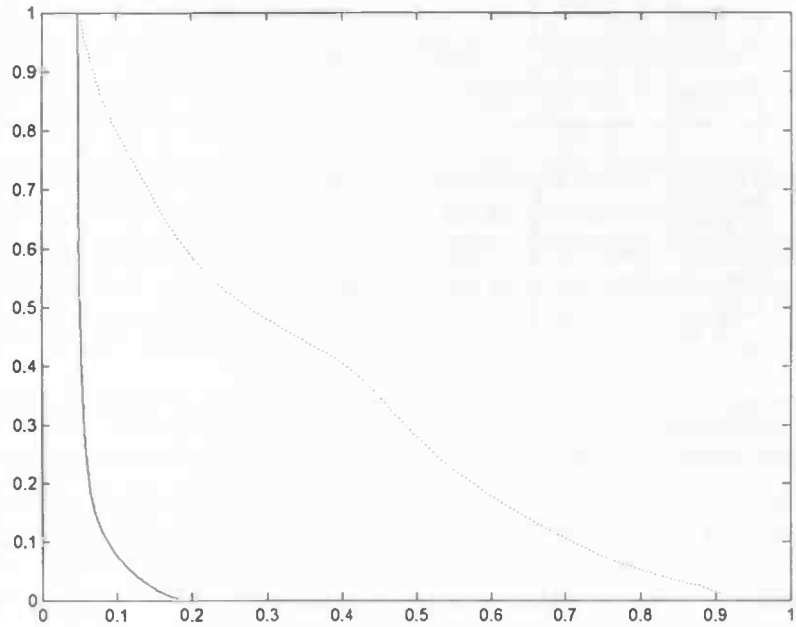


Figure 6.17: The precision-recall-graph for the original version of the generalised pattern spectra (the solid line), as well as the version extended by the histograms (the dotted line), tested on the COIL-20 database. All queries are used to determine this graph, by taking the average precision and recall values.

6.2.6 Comparison

In this section, the methods will be compared to each other using the rank difference trend analysis, explained in section 6.1.3. Since this measure compares two rank lists, and for every query one rank list exists for every method, comparing two methods with each other results in comparing two rank sets for all the queries; for instance, for query A, the rank set belonging to method X is compared to the rank set belonging to method Y. This is done for every query, in order to obtain a complete picture of the assessment measure.

The RDTA produces a value between -1 and 1: a value < 0 indicates that the first method is less effective than the second, a value > 0 indicates that the first method is more effective than the second. Furthermore, from this value, a statement can be made about both systems: a value of of $|0.2|$ means that one system is 1.4 times better then the other, a value of $|0.4|$ means that one system is 2 times better than the other.

Because 1440 values (1440 is the number of images in the COIL-20 database) are obtained when comparing two systems, an average value is being determined first. After that, the number of times one system is more effective versus the number of times the other system is more effective is computed. Finally, the percentage of the queries that produced a value of at least 0.2 and 0.4 is computed.

The variable which determines the weight is set to 0.3 in all cases. Furthermore, it became evident that the error level was at most 0.0001 in all cases. Therefore, it is ignored in future evaluation of the

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	37.6%	47.5%	54.5%	59.3%	62.8%	65.9%	68.2%	69.9%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	0%	0%	0%	0%	0%	2.3%	5.6%	27.1%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	0%	0%	0%	0.3%	1.8%	4.7%	7.0%	17.2%
$P(50)$	0%	0%	0%	0%	0%	0%	0%	5.3%
$P(N_R)$	0%	0%	0%	0%	0%	0%	0%	0.8%
$R_P(0.5)$	0%	0%	0%	0%	0%	0%	0%	0%
$R(100)$	0%	0%	0%	0%	0%	0%	0%	3.1%

Table 6.9: The performance measures of the spatial size distributions, with the histograms extension, tested on the COIL-20 database.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	72.2%	79.9%	83.0%	85.4%	87.6%	89.0%	89.7%	90.8%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	31.2%	40.2%	50.3%	61.9%	70.3%	78.7%	85.6%	92.2%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	23.0%	29.7%	34.9%	41.4%	49.9%	57.4%	63.6%	70.6%
$P(50)$	8.1%	14.3%	18.1%	23.3%	32.0%	42.2%	51.0%	60.3%
$P(N_R)$	1.3%	9.9%	12.7%	16.7%	21.5%	31.9%	43.1%	53.6%
$R_P(0.5)$	11.7%	15.3%	19.2%	21.5%	25.5%	30.0%	35.7%	41.4%
$R(100)$	9.0%	13.1%	17.5%	23.0%	31.3%	40.2%	48.5%	58.3%

Table 6.10: The performance measures of the multi-scale connectivity, with the histograms extension, tested on the COIL-20 database.

assessment measure in this chapter.

Original version vs. histogram version

First, the original version of each method is compared to the version with the histograms. The generalised pattern spectra appeared, in section 6.2.5, to gain from the histograms. When computing the RDTA, this notion is confirmed: the version with the histograms is, on average, more than two times better than the original version. In 16% of the queries, the original version is more effective, and in 84% of the queries the histogram version is more effective. Furthermore, the original version is at least 1.4 times better than the histogram version in only 7.6% of the queries (corresponding to 109 queries). In 57.2% of the queries, though, the version with the histograms is at least 2 times more effective than the original version. So it can safely be concluded, that the version with the histograms perform better than the original version.

The notion arose, in section 6.2.5, that both versions of the spatial size distribution performed equally well. This notion is confirmed by the RDTA: the mean value that was computed is less than 0.01. Although the number of times the original version appeared to be more effective is about 64%, against

only 36% of the queries the histogram version appeared to be more effective, the differences between the two methods is very small. The original method is only in 4.4% of the queries at least 1.4 times more effective, while the histograms version is at least 1.4 times more effective in 10.7% of the queries. Because of these small differences, it can be concluded that both methods perform equally well (or worse).

In section 6.2.5, it became quite clear that the original version of the multi-scale connectivity method performed better than the histogram version. This is confirmed by the RDTA: the average value for the RDTA implies that the original version is more than 2 times more effective than the histograms version, on average. In 88% of the queries, the original version performed more efficient, while in 58.7% the original version is at least twice as effective.

Generalised pattern spectra vs. Spatial size distribution vs. Multi-scale connectivity

Now, when comparing the three different methods (using the histogram version in case of the generalised pattern spectra, and the original versions in the other two cases), it becomes obvious that the spatial size distributions are no match for the other two methods: the generalised pattern spectra are on average more than twice as effective, while the multi-scale connectivity method is on average almost three times as effective.

When comparing the generalised pattern spectra with the multi-scale connectivity method, the latter is more effective: on average, the RDTA is about 0.15, in favour of the multi-scale connectivity method. In 32% of the queries, the generalised pattern spectra are more effective, in 68% the multi-scale connectivity method. Although the assessment measure seems to be clear, the generalised pattern spectra *are* at least 2 times more effective for 109 queries (7.6%). The multi-scale connectivity method, however, is at least 2 times more effective for 26.7% of the queries.

6.2.7 Area pattern spectra

All the proposed methods are actually extensions to the standard area pattern spectra. Therefore, it would be interesting to see, how these methods perform, with respect to the basic area pattern spectra. In this section, this method (the original version, as well as the extension using the histograms, as explained before and used for the other methods, using the same parameter settings that were determined before) is being tested the same way as for the other methods: first, an overview is given of the performance, by looking at the performance measures for all queries. After that, the queries that were used before are now used again to look more closely to he results of the area pattern spectra. Finally, it will be compared to the other three methods to see if the proposed extensions actually provide extra information regarding content-based image retrieval.

Results

The performance measures for the area pattern spectra are shown in table 6.11, the version with the histograms can be seen in table 6.12. In figure 6.18, the corresponding PR-graphs are shown (the solid line corresponds to the original version, the dotted line to the version with the histograms).

The difference between these two lies within the rank-based measures and the precision/recall measures. The original version has higher values for the precision and recall measures (almost one quarter of the queries have a value of at least 0.5 for the precision after the first 20 results and 159 queries (11.0%) have a recall after 100 retrieved images of at least 0.5), while the version with the histograms

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	31.7%	37.9%	41.7%	44.4%	46.3%	48.0%	49.6%	51.2%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	5.8%	14.4%	23.7%	35.2%	47.8%	60.0%	65.1%	68.6%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	1.5%	4.2%	9.5%	14.2%	18.4%	23.8%	28.5%	33.9%
$P(50)$	0%	0.6%	1.6%	3.8%	6.7%	11.5%	20.7%	30.5%
$P(N_R)$	0%	0%	0.1%	1.4%	3.8%	6.9%	12.8%	24.7%
$R_P(0.5)$	0%	0.8%	2.1%	3.5%	4.9%	6.6%	9.0%	12.2%
$R(100)$	0%	0.1%	1.3%	3.7%	6.5%	11.0%	20.6%	33.8%

Table 6.11: The performance measures of the area pattern spectra, tested on the COIL-20 database.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	36.3%	51.7%	61.5%	66.8%	70.1%	74.3%	77.7%	80.5%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	10.8%	32.0%	45.6%	61.3%	73.1%	80.9%	85.8%	91.5%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	0%	0.6%	1.9%	6.6%	13.8%	21.8%	31.3%	43.5%
$P(50)$	0%	0%	0.1%	0.6%	4.6%	14.5%	23.8%	35.8%
$P(N_R)$	0%	0%	0%	0%	0.7%	7.0%	21.3%	32.0%
$R_P(0.5)$	0%	0%	0.3%	0.7%	2.1%	3.7%	5.8%	8.1%
$R(100)$	0%	0%	0.3%	1.6%	8.3%	21.9%	32.9%	46.3%

Table 6.12: The performance measures of the area pattern spectra, with the histograms extension, tested on the COIL-20 database.

have a higher Rank_1 and $\widetilde{\text{Rank}}$ (more than half of the queries returned a relevant image within the first two results and almost one third of the queries produced a value of at most 0.1 for the $\widetilde{\text{Rank}}$). The precision-recall-graph of the version using the histograms is better than the original version, see figure 6.18.

Discussion

In table 6.13 and 6.14, the performance measures of the original version and the histogram version, respectively, are shown for the queries that were hand-picked before. Coincidentally, most of these queries were better retrieved by the histogram version.

Compared to the original versions of the generalised pattern spectra and the spatial size distribution, the original version of the area pattern spectra appear to have a similar performance. No extra information is gain by these two extensions. The multi-scale connectivity method, however, *is* much better than the area pattern spectra.

Looking at the histogram versions, the generalised pattern spectra perform significantly better than any of the two versions of the area pattern spectra. The spatial size distributions fall behind again, and

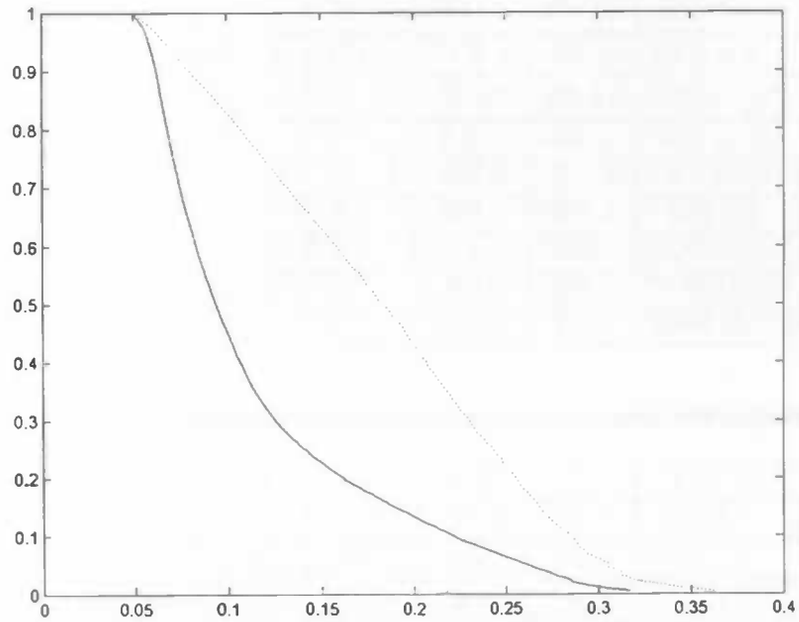


Figure 6.18: The precision-recall-graph for the area pattern spectra, the original version (the solid line), as well as the extended version with the histograms (the dotted line), tested on the COIL-20 database. All queries are used to determine this graph, by taking the average precision and recall values.

appear to be of no extra value in the field of content-based image retrieval. Nevertheless, the method might be interesting for other fields of research, like texture classification.

6.3 Washington database

Now all methods are tested and evaluated on the COIL-20 database, the next experiments are performed on the database of the Washington university. This is a database, consisting of some more real-life images, as opposed to the COIL-20 database, which is a set of images of objects in an artificial setting. The size of the database and the classes is quite similar to the COIL-20 database: the Washington database consists of 1333 images, in 22 classes. These classes, however, do not contain an equal number of images. Most classes contain about 48 images, but some classes only contain about 30 images, where other classes contain over 100 images. Therefore, it is a bit harder to compare different classes with each other, to see which classes are recognised better. For instance, a class containing only 30 images will never reach a value of 1.0 for the precision after 50 retrieval results: the maximum number would be 0.6 for that class. This will be taken into account when looking at the separate classes during evaluation.

Another difference between the two databases is, that the images in the COIL-20 database all were the same size: 128×128 pixels. Not only are the images in the Washington database not all the same size (they range from 320×240 to 441×294), they are much large than the images in the COIL-20

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	225	0.4951	0	0	0	0	0
Vehicle	15	0.2849	0.05	0.04	0.0563	0	0.0563
Tub	22	0.3174	0	0.08	0.0563	0	0.1268
Pipe	44	0.4553	0	0.02	0.0282	0	0.0282
Stone_jar	65	0.1954	0	0	0.0141	0	0.0423
Cup	1	0.1228	0.25	0.16	0.1268	0.0141	0.1831
Racecar	25	0.3658	0	0.08	0.0563	0	0.0704
Duck	125	0.4132	0	0	0	0	0

Table 6.13: The performance measures for the queries that were hand-picked in the previous sections, retrieved by the area pattern spectra.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	12	0.1559	0.15	0.2	0.2254	0	0.3099
Vehicle	1	0.1076	0.15	0.3	0.3239	0.0141	0.4085
Tub	1	0.0447	0.5	0.42	0.4366	0.1408	0.5352
Pipe	6	0.0976	0.15	0.16	0.1549	0	0.2817
Stone_jar	59	0.2871	0	0	0.0282	0	0.0282
Cup	7	0.1677	0.25	0.26	0.2394	0	0.3239
Racecar	1	0.2808	0.15	0.26	0.2254	0.0282	0.2676
Duck	2	0.1900	0.2	0.2	0.1972	0	0.2958

Table 6.14: The performance measures for the queries that were hand-picked in the previous sections, retrieved by the area pattern spectra, extended with the histograms.

database. Therefore, the bin size will be larger if the same number of features is used; the bin size will be kept the (about) the same for all methods, however.

In this section, the results will be presented for all three feature extraction methods. Since the results of the COIL-20 database showed, that a smaller bin size might increase the results, a version with histograms is used in the experiments from the beginning. First, all methods are discussed separately (looking at the original versions as well as the histograms version). After that, the methods are compared to each other, and this comparison is being discussed.

6.3.1 Generalised pattern spectra

First, the generalised pattern spectra are tested. Experiments confirmed the conclusion that was made before, about the histograms: the smaller bin size that is obtained by using histograms improves results significantly. Therefore, only this version is being discussed in this section.

Results

For this database, the same test-setup is used: every image in the database is used as a query-image once, so a total of 1333 query-images are tested. The results can be seen in table 6.15 and figure 6.28 (the solid line).

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	39.1%	51.5%	58.4%	63.8%	67.5%	70.4%	72.5%	74.0%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	5.6%	20.9%	30.1%	42.8%	58.1%	76.2%	88.6%	94.7%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	8.6%	14.3%	17.7%	19.7%	23.1%	26.9%	33.8%	42.4%
$P(50)$	4.5%	10.1%	15.2%	18.5%	19.8%	22.8%	28.8%	37.1%
$P(N_R)$	0%	0%	2.3%	14.2%	18.8%	22.0%	25.6%	34.9%
$R_P(0.5)$	0%	11.6%	15.1%	16.7%	17.3%	18.0%	18.2%	18.8%
$R(100)$	0%	0%	2.3%	4.4%	5.3%	7.2%	10.1%	33.0%

Table 6.15: The performance measures of the generalised pattern spectra, with the histograms extension, tested on the Washington database.

In this table, it can be seen that for over 500 queries (39.1%) the first retrieved image was an image that belongs to the ground truth of that query-image, while for over a 1000 queries the first *relevant* image (relevant image is used for images that belong to the ground truth for the corresponding query-image) is found within the first 10 retrieved images. The normalised average rank ($\widetilde{\text{Rank}}$) is less than 0.1 for more than one fifth of the number of queries. For 8.6% the precision after 20 retrieved images is 100%, and for 4.5% the precision after 50 retrieval results is 100%. This last number is should be put into perspective, since only 617 images are part of a class that can even retrieve at least 50 relevant images.

The recall results are also quite dependent on the number of images in the ground truth. When there are over 100 images in a class, which is the case in three classes, a recall after 100 retrieved images of 1.0 can never be obtained. Although three classes does not seem that much, they represent nearly 36% of the database. The largest class, the class with images from Greenland contains 255 images, can obtain a maximum recall after 100 retrieved image of 0.39 (if all 100 retrieved images are relevant ones). Therefore, a total of one third of the number of queries obtaining a value of 0.3 for this performance measure is not that bad.

The recall when precision drops below 0.5 ($R_P(0.5)$) is a more suitable performance measure. The value of this measure is at least 0.8 for more than 200 queries (15.1%).

Discussion

This method is further discussed using three hand-picked images. In figure 6.19, the hand-picked images can be seen.

The first image is an image from the class *Campus-in-fall*. The class consists of images with buildings, trees, grass, sidewalks and people. The concerning image (campusinfall19) consists of a building behind some leafless trees, along with a small sidewalk and some grass. In figure 6.20, the first 10 retrieved can be seen. Although these results do not appear in the ground truth of the query-image, they can not be called bad results. What is returned, are images with trees, sidewalks and buildings,

(a) (b) (c)

Figure 6.19: Three images, that are used as query-image, to test the generalised pattern spectra on the Washington database. The image from the class Campusinfall, image (a) is used as image that produces bad results, the image from Yellowstone (b) produces average results and the image from the class Columbiagorge (c) is retrieved very well.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Campusinfall	145	0.4675	0	0	0	0	0
Yellowstone	4	0.1749	0.45	0.28	0.2979	0	0.3617
Columbiagorge	1	0.0416	1.0	1.0	0.8537	0.8659	0.8537
Iran	17	0.2787	0.05	0.08	0.0853	0	0.1250
Australia	64	0.3117	0	0	0	0	0.1034
Greenland	1	0.0487	1.0	0.98	0.7244	0.9370	0.3425

Table 6.16: The performance measures for the queries that were hand-picked in this section; the other images are chosen in section 6.3.3 as queries that characterise the multiscale connectivity.

which are exactly what appears on the query-image. Since they are classified to be in different classes, though, they are marked "wrong", and hence, the performance is bad for this query-image.

The second image is an image from the class *Yellowstone*, which consists of images with people, geysers, hills, trees and other kinds of nature. The query-image is an image with some hills and trees, and on the foreground massive rocks. In figure 6.21, the first 10 retrieved images are shown. Some of these are from the correct class (5 out of the first 10). The other images are images contain mostly trees, sidewalks and buildings. If one does not know about the classes, most of the retrieved images *could* be correct.

The third image is from the class *Columbiagorge*. This class consists of images of buildings, water and people. The query-image consists of some houses, a sidewalk and some bushes between the houses and the sidewalk. All the images that are retrieved, see figure 6.22, are images that appear in the ground truth for the query-image. All the performance measures for these three images can be seen in table 6.16.

Two classes have specifically very good values for the performance measures: *columbiagorge* and *greenland*. The first class is explained when discussing the third hand-picked image. The *greenland*-class consists of images from Greenland: a lot of nature (hills, mountains with snow, and (frozen) lakes), some villages and people. When images from these classes are used as query-image, the

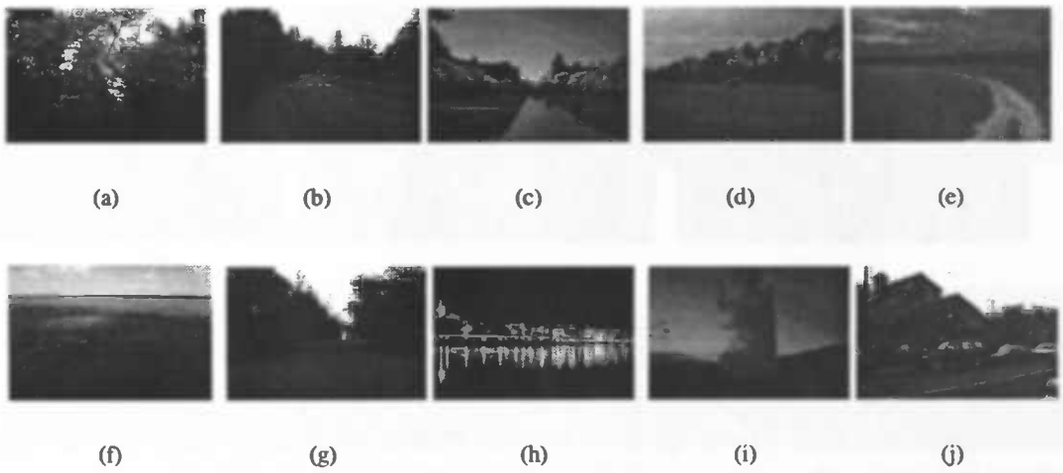


Figure 6.20: These images are retrieved by the generalised pattern spectra when the image in figure 6.19(a) is used as query. Although none of these results belong to the ground truth, most of them *do* have some common objects. The retrieved images are from the following classes: Cherries (images (a) and (c)), Arborgreens (images (b) and (g)), Cannonbeach (images (d) and (j)), Sanjuans (image (e)), Australia (image (f)) and Geneva (image (h)).

performance is quite good. Other classes do not have good performance measures, but this might be caused by the (subjective) ground truth: the images that are returned for these classes are not by definition wrongly retrieved, but simply do not appear in the ground truth and are therefore not recognised as being correct.

6.3.2 Spatial size distributions

Next, the spatial size distribution is tested. Even though in the previous sections it appeared that this method cannot compete with the other two methods, this method might perform well on another database. Experiments showed that the original version performed better, so only these results are shown and discussed here.

Results

In table 6.17 the results are shown for all queries, and in figure 6.28 the precision-recall-graph is shown (the dotted line).

These results are still not good; only in about half of the queries the first relevant image is retrieved with the first eight retrieved images, and 8% of the queries produced a value of at most 0.2 for the normalised average rank. At first sight, the precision after the first 20 retrieved results looks not bad (7.0% produces a value of 1.0), but a closer look reveals that only 17 images produced a value of at least 0.5 (93 out of the total of 110 images that obtain at least 0.5 obtain 1.0, so 17 images obtain a value between 0.5 and 1.0).

The same observation goes for the other measures, be it a little less extreme. For instance, the value of the recall when the precision drop below 0.5 is 0.8 for 4.4% of the queries. For 8.0% this value is

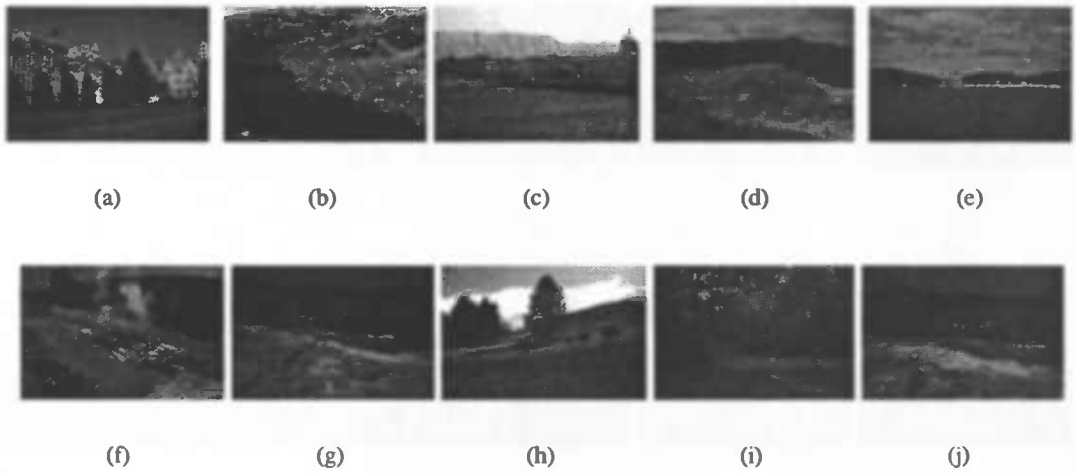


Figure 6.21: The first ten retrieved images by the generalised pattern spectra are shown here. Five images belong to the ground truth, being images (d), (f), (g), (h) and (j). The other images are from classes Campusinfall, image (a), Swissmoutains, image (b), Geneva, image (c), Sanjuans, image (e) and Arborgreens, image (i).

at least 0.1, which means that a total number of 47 queries have a $R_P(0.5)$ between 0.1 and 0.8, and 59 queries have a value of at least 0.8.

Because of these bad performance measures, further discussion is omitted from this chapter; the spatial size distribution appear not to be of any use to the field of content-based image retrieval. Nevertheless, as said before, it might be a good feature extraction method to other fields of research. For content-based image retrieval, however, it is not.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	18.5%	26.3%	31.8%	36.6%	40.3%	44.0%	48.9%	52.0%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	2.5%	3.5%	4.3%	5.0%	8.0%	11.4%	17.5%	46.6%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	7.0%	8.0%	8.0%	8.1%	8.1%	8.3%	10.1%	15.1%
$P(50)$	5.0%	7.3%	7.4%	7.6%	7.7%	7.8%	8.0%	12.3%
$P(N_R)$	0%	2.9%	4.6%	5.6%	6.5%	6.9%	7.4%	7.6%
$R_P(0.5)$	0%	2.9%	4.4%	5.5%	6.3%	6.8%	7.3%	7.4%
$R(100)$	0%	2.8%	4.6%	5.6%	6.5%	6.9%	7.4%	8.9%

Table 6.17: The performance measures of the spatial size distributions, tested on the Washington database.

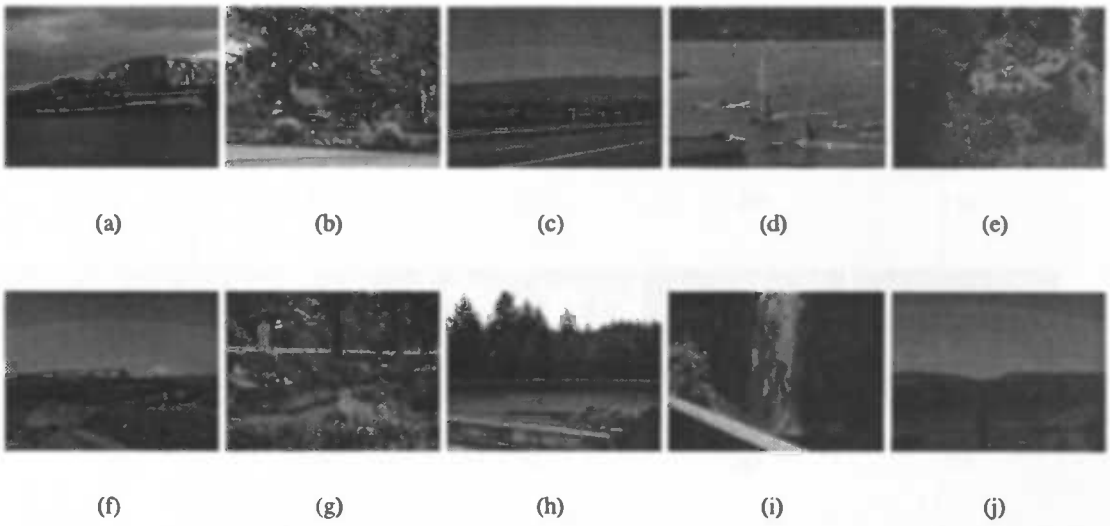


Figure 6.22: All these images are from the class Columbiagorge, and are retrieved by the generalised pattern spectra before any other image is retrieved, when using the image in figure 6.19(c) as query-image.

6.3.3 Multi-scale connectivity

Finally, the multi-scale connectivity method is tested. On the COIL-20 database, the version with the histograms did not improve performance. Strangely, though, the results *are* better when using the histograms version on the Washington database. Because of this strange fact, results of both methods are shown in this section.

Results

In table 6.18 and 6.19 the results can be seen for the multi-scale connectivity method and the version with the histograms respectively. Furthermore, the precision-recall-graphs for these methods are shown in figure 6.23

As one can see, the results of the version with the histograms appear to be better than the original version. Nearly twice as many queries of the histogram version retrieve a relevant image as first result, and 55.8% obtain a value of at most 0.15 for the $\widetilde{\text{Rank}}$ (against only 3.8% for the original version).

About one third of the queries have a precision after 20 retrieved images of at least 0.7, and nearly a quarter of the queries have a precision after N_R image (where N_R is the number of images in the ground truth) of at least 0.7. These values are much lower for the original version: 1.3% and 0% respectively.

The recall results for the histograms version are also better: 20.2% of the queries have a recall of 1.0 when the precision drops below 0.5 and 82 queries have a recall of 1.0 after 100 retrieved images.

Further more, it can be seen from the precision-recall-graphs in figure 6.23 that the dotted line (which belongs to the version with the histograms) is always above the solid line (belonging to the original

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	24.9%	35.6%	43.3%	48.5%	53.3%	57.5%	60.1%	63.8%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	0%	0%	3.8%	7.1%	11.9%	27.9%	45.2%	63.6%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	0%	0%	0.2%	1.3%	3.7%	7.8%	12.3%	22.0%
$P(50)$	0%	0%	0%	0%	0.2%	2.9%	9.2%	15.7%
$P(N_R)$	0%	0%	0%	0%	0%	1.0%	3.5%	13.1%
$R_P(0.5)$	0%	0%	0%	0%	0.5%	0.8%	1.1%	1.3%
$R(100)$	0%	0%	0%	0.1%	2.3%	3.2%	4.3%	8.7%

Table 6.18: The performance measures of the multi-scale connectivity, tested on the Washington database.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	46.5%	56.1%	62.3%	66.9%	71.0%	73.7%	76.7%	79.0%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	26.6%	34.3%	55.8%	66.1%	85.6%	93.6%	97.7%	98.9%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	13.6%	23.9%	30.7%	33.2%	35.2%	37.7%	40.6%	47.6%
$P(50)$	6.2%	19.7%	26.6%	29.0%	32.9%	34.7%	37.4%	42.6%
$P(N_R)$	6.2%	6.2%	18.2%	24.4%	28.8%	31.0%	35.9%	41.3%
$R_P(0.5)$	20.2%	21.3%	24.0%	25.9%	26.3%	26.9%	28.9%	29.7%
$R(100)$	6.2%	6.2%	6.2%	6.6%	13.8%	19.6%	30.0%	51.8%

Table 6.19: The performance measures of the multi-scale connectivity, with the histograms extension, tested on the Washington database.

version).

Discussion

Even though it is still strange, the histograms version obviously *do* perform better. Therefore, this version will only be discussed further; the original version is not discussed here. This method is being examined more closely by looking at the results of the query-images, shown in figure 6.24. In table 6.20, the performance measures for these queries are shown. Also, in this table, the results for the images chosen for the generalised pattern spectra are shown. Furthermore, in table 6.20, the performance measure belonging to these three images are shown for the generalised pattern spectra.

The first image is an image from the class *Iran*. This class consists mostly of images with houses, mosques, walls with tiles and other buildings. The query-image that was picked consists of a mosque with a lot of people in front of it. The retrieved images are shown in figure 6.25. All these results are from other classes; the consist of buildings and people. Although these images are not in the ground

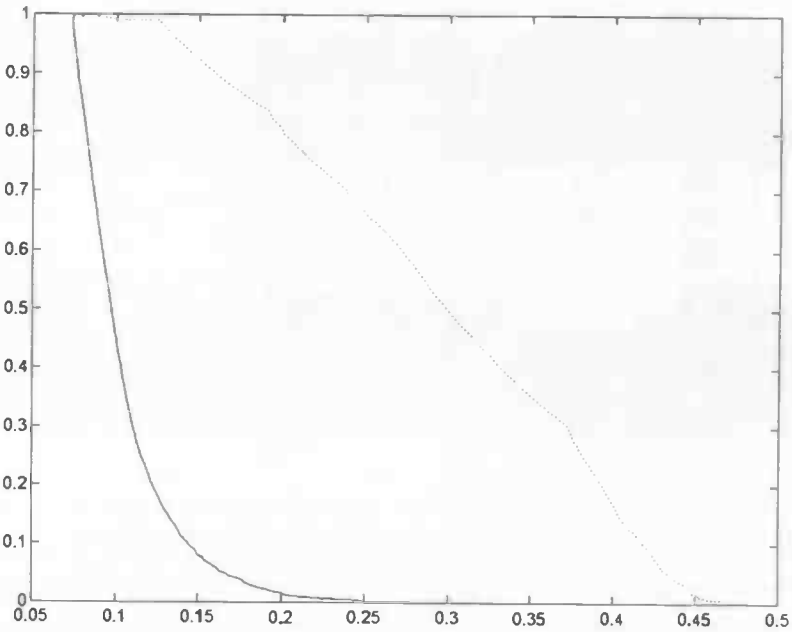


Figure 6.23: The precision-recall-graph for the multi-scale connectivity, the original version (the solid line), as well as the extended version with the histograms (the dotted line), tested on the Washington database. All queries are used to determine this graph, by taking the average precision and recall values.

truth of the query-image, they do have some similarities with the query-image.

The second image is from the class *Australia*. Most images consist of bushes, rocks, beach and water. Some have certain animals, people or building on them. The query-image is a picture of a kangaroo, with bushes around it. Some of the returned images are from the ground truth; other images contain the sea or some bushes and trees.

The third image is from the class *greenland*. In the previous section, the contents of this class was explained. All the returned images (at least the first 10) are from the ground truth, see figure 6.27.

Images from classes *Cambridge*, *Columbiagorge* and *Greenland* are retrieved very well; other images have less good values for the performance measures. The class *Cambridge* consists mostly of buildings from cambridge: palaces, bridges, shops and other types of buildings are shown, the contents of the other classes are discussed before.

6.3.4 Comparison

In this section, the performance of the generalised pattern spectra is being compared to that of the multi-scale connectivity method. The spatial size distributions were concluded not to be suitable for content-based image retrieval, and is therefore not compared to the other two methods.

(a) (b) (c)

Figure 6.24: Three images, that are used as query-image, to test the multi-scale connectivity on the Washington database. The image from the class Iran, image (a) is used as image that produces bad results, the image from Australia (b) produces average results and the image from the class Greenland (c) is retrieved very well.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Campusinfall	16	0.3079	0.1	0.08	0.0851	0	0.1489
Yellowstone	10	0.2235	0.05	0.12	0.1064	0	0.1915
Columbiagorge	1	0.0007	1.0	1.0	1.0	1.0	1.0
Iran	282	0.4254	0	0	0	0	0
Australia	3	0.0842	0.35	0.24	0.2759	0	0.6552
Greenland	1	0.0195	1.0	0.96	0.8346	1.0	0.3701

Table 6.20: The performance measures for the queries that were hand-picked in this section; the other images are chosen in section 6.3.1 as queries that characterise the generalised pattern spectra.

For comparison, the rank difference trend analysis is being used again. This time, the database consists of 1333 images, so 1333 RDTA-values are obtained when comparing the two methods.

Generalised pattern spectra vs. Multi-scale connectivity

In section 6.3.1 and 6.3.3 it appears that the multi-scale connectivity method produces better retrieval results than the generalised pattern spectra. The RDTA confirms this: for 85% of the queries, retrieval results are better for this method. For 298 queries (22.4%) the results are at least twice as effective, while for more than half of the queries the RDTA is at least 0.2 (e.g. 1.4 times more effective).

Area pattern spectra

Again, the area pattern spectra are used as reference point, to determine if the extensions provide extra information that is essential in the retrieval process.

For this database, both the generalised pattern spectra and the multi-scale connectivity benefit from the extensions. The generalised pattern spectrum is about 1.5 times more effective on average, and in 84% of the queries, a more effective retrieval result is produced. The multi-scale connectivity method benefits even more: in 97% of the queries, the retrieval results are more effective and on average, the multi-scale connectivity method is more than twice as effective.

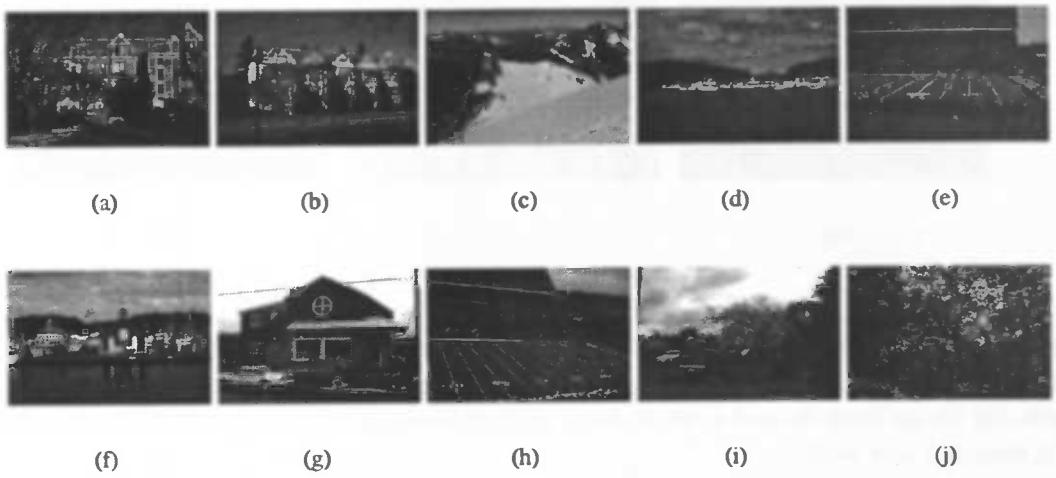


Figure 6.25: When using the multiscale connectivity method, these images are returned when using image 6.24(a) as query. None of these images are from the same class; the belong to the following classes: Campusinfall, images (a) and (b), Swissmountains, image (c), Sanjuans, images (d) and (f), Football, images (e) and (h), Cannonbeach, image (g), Cherries, image (i), and Arborgreens, image (j).

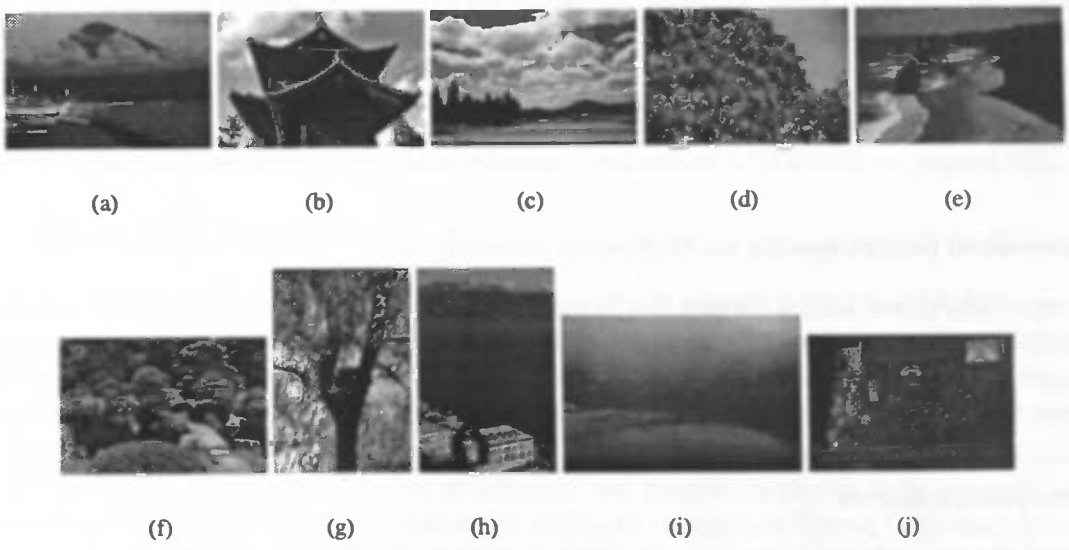


Figure 6.26: The multiscale connectivity method retrieves these ten images when using image 6.24(b) as query-image. Images (c), (d), (e) and (g) belong to the ground truth of the query-image. The other images belong to the classes Indonesia, image (a), Japan, images (b) and (f), Italy, image (h), Greenland, image (i), and Iran, image (j).

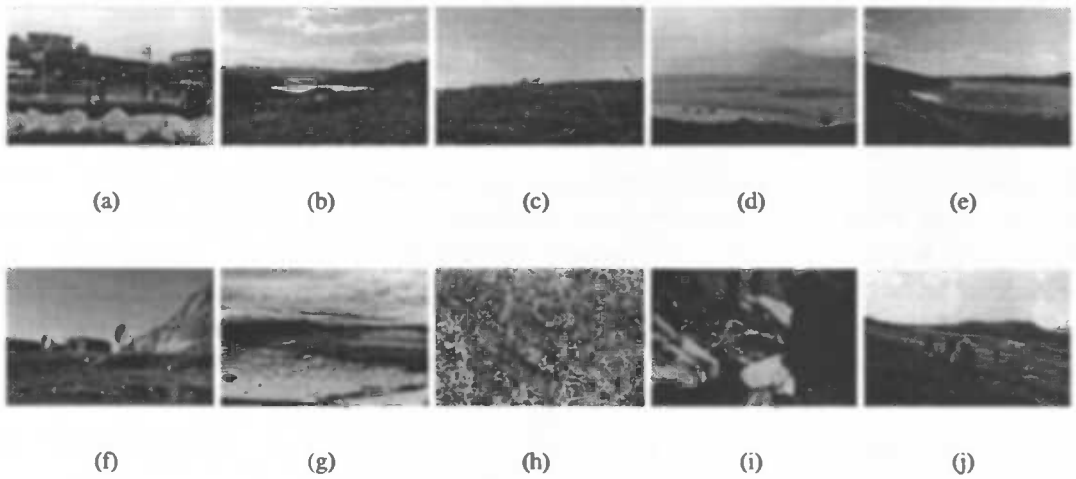


Figure 6.27: The images are retrieved before any other image is retrieved by the multi-scale connectivity. They all belong to the ground truth of the query-image, shown in 6.24(c).

6.3.5 Overview

After performing the experiments discussed in this section, it can be concluded that the spatial size distributions are *not* suited for content-based image retrieval. The performance does not increase with respect to the area pattern spectra (which is the basis of the spatial size distributions), and the performance falls far behind the performance of the other two methods, on both the COIL-20 and the Washington database (see the precision-recall-graphs in figure 6.28).

The multi-scale connectivity method produces the best values for the performance measures, which can also be seen in the precision-recall-graph in figure 6.28: the dashed line of the multi-scale connectivity method is always (slightly) above the solid line of the generalised pattern spectra. On average, this method is about 1.5 times more effective than the generalised pattern spectra when experimenting on the Washington database. This database, however, is divided into 22 classes, which is done very subjectively. Although there is actually no other way than subjectively classifying the images, this is an important fact that can not be ignored. When 10 people would classify the images, probably 10 different classification would be obtained. Therefore, it is more interesting to look more closely at the results of certain queries.

When looking at queries and the retrieval results, it can be seen that most images that are retrieved can somehow be justified for being retrieved. Even though they might not appear in the ground truth of the query, there are often resemblances between the retrieved images and the query-image. Another classification of the images will produce different performance measures. Nevertheless, the multi-scale connectivity method appears to be more effective in retrieving images.

The multi-scale connectivity method produced better results for the Washington database when decreasing the bin size to one and creating a histogram of the values. Strangely enough, this was not the case in the COIL-20 database. One explanation for this might be that the images in the COIL-20 database are much smaller than the images in the Washington database. The size of the images is a decisive factor in determining the bin size, and hence larger images means a larger bin size. It can be

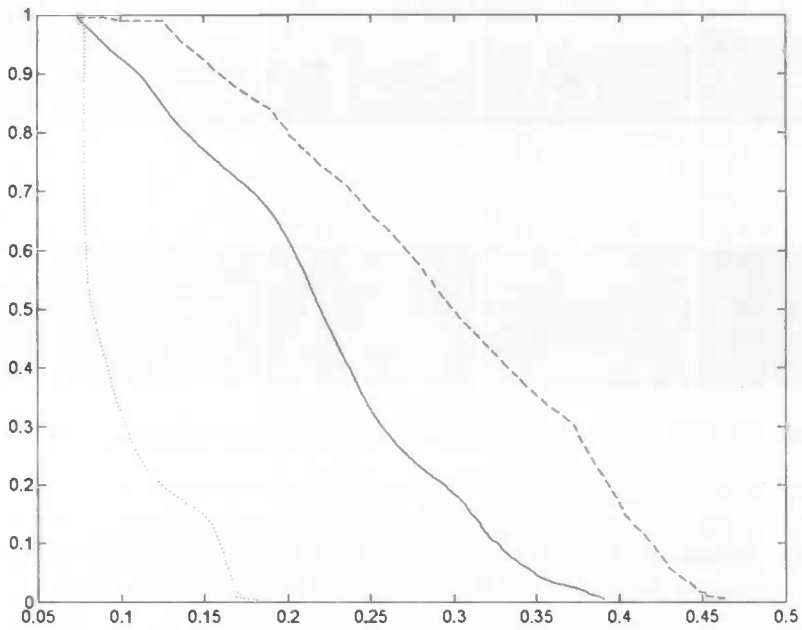


Figure 6.28: The precision-recall-graph for the generalised pattern spectra (the solid line), the spatial size distributions (the dotted line), and the multi-scale connectivity method (the dashed line), tested on the Washington database. All queries are used to determine this graph, by taking the average precision and recall values.

concluded, that the multi-scale connectivity method is sensitive to the bin size, like the generalised pattern spectra.

6.4 Other CBIR-software

Now that is established that the spatial size distribution is unsuited for content-based image retrieval, and that multi-scale connectivity as well as generalised pattern spectra *can* be used for this purpose, it is interesting to see whether the performance of these two methods can compete with existing software. There are quite a few applications which let a user search a database with images, looking for images that are similar to a certain query-image; an overview of these applications is given in [38], [37] and [12]. From this collection of software, one program is chosen to be compared to the methods that were presented before: IMatch. According to Venters and Cooper [38], this application is competitive with respect to the software that is available at this moment; all applications perform about equally (see [38] for a comparison of different systems). A trial version of this program can be downloaded [5], a full version costs $59.95.

In this section, IMatch is explained shortly. After that, some experiments are performed and the performance is discussed. Finally, the performance is compared to that of the generalised pattern

[5] `http://www.photools.com/`

spectra and the multi-scale connectivity.

6.4.1 IMatch

IMatch is developed by Mario M. Westphal, and is an application with which a user can perform image retrieval on their own datasets. Because IMatch is commercial software, no extensive documentation about the technical details is available. Therefore, no information can be given about the distance function that is used, nor about the number of features that is being determined for each image. For retrieval, several methods can be used. In this section, IMatch is tested with the standard match features, based on colour and shape (quick). This method identifies specific shapes, shape distributions and colour distributions in an image and orders the result set based on the overall distance.

6.4.2 Experiments

IMatch is tested using the same setup as before: each image from a database is used as query once, and for all these queries the performance measures are being determined. The results for the COIL-20 database can be seen in table 6.21, the results for the Washington database are shown in table 6.22. One disadvantage of IMatch is that only images are returned that are found to be similar, other images are not retrieved. For some of the performance measures, however, ranks for *all* images from the ground truth should be known; if these images are not returned, these ranks are not known. Therefore, an estimation has to be made about these ranks. In the worst-case-scenario, these images have the lowest possible ranks, in the best-case-scenario, these images have the highest possible ranks. Because the performance measures for the worst-case and the best-case scenario differ slightly, the average ranks are determined, and these are used to calculate the performance measures. For instance, if the ground truth of a certain query-image contains 50 relevant images, but only 49 are returned, the worst-case scenario would be that the final image from the ground truth would be the last image from the database (so the rank for this image would be 1440 in the case of the COIL-20 database, and 1333 in the case of the Washington database). The best-case scenario is, that rank of the final relevant image would be 1 higher than the total number of retrieved images; if 500 images are returned, the best-case scenario would assign a rank of 501 to the final relevant image. In the average case, the rank would be the average of the rank of the worst-case and best-case. This is the rank that is used to calculate the performance measures in table 6.21 and 6.22.

The results for the COIL-20 database are extremely good, especially compared to the generalised pattern spectra and the multi-scale connectivity method. For all the queries, the first retrieved image was an image from the ground truth of the query-image, and for nearly 900 queries (61.9%), the normalised average rank is at most 0.1. Even more queries (63.3%) have a precision after 20 retrieved images of 1.0, meaning that all 20 retrieved images are part of the ground truth of the query-image. The precision after the first 50 retrieved images is 1.0 for more than one third of all the queries, and for 27.4%, all images from the ground truth are returned before any other image is returned.

The Washington database is retrieved less good. This time, almost half of the queries produced an image from the ground truth as first retrieval result, and only 11.2% of the queries has got a normalised average rank of at most 0.1. The precision (after 20, 50 as well as N_R retrieved images) is 1.0 for about 5% of the queries, and for the recall-values, the same thing can be said.

When looking at the specific queries more closely (the queries the were chosen for the generalised pattern spectra and the multi-scale connectivity methods before are used again), see table 6.23 and table

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	100%	100%	100%	100%	100%	100%	100%	100%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	54.4%	61.9%	73.3%	86.8%	89.4%	90.4%	92.8%	94.3%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	63.3%	70.1%	73.7%	76.7%	81.5%	88.2%	94.2%	98.6%
$P(50)$	34.7%	44.4%	50.8%	57.2%	63.8%	69.3%	81.1%	90.4%
$P(N_R)$	27.4%	30.2%	36.7%	44.9%	52.8%	60.3%	69.7%	85.8%
$R_P(0.5)$	34.0%	39.1%	45.3%	50.3%	55.3%	61.2%	66.7%	73.5%
$R(100)$	30.1%	34.5%	42.8%	49.6%	56.4%	65.9%	79.9%	88.6%

Table 6.21: The performance measures of IMatch, tested on the COIL-20 database.

	≤ 1	≤ 2	≤ 3	≤ 4	≤ 5	≤ 6	≤ 7	≤ 8
Rank_1	49.4%	59.5%	65.5%	68.8%	72.2%	75.1%	76.9%	78.4%
	≤ 0.05	≤ 0.1	≤ 0.15	≤ 0.2	≤ 0.25	≤ 0.3	≤ 0.35	≤ 0.4
$\widetilde{\text{Rank}}$	7.4%	11.2%	17.7%	31.3%	48.3%	59.8%	67.7%	77.1%
	≥ 1.0	≥ 0.9	≥ 0.8	≥ 0.7	≥ 0.6	≥ 0.5	≥ 0.4	≥ 0.3
$P(20)$	5.2%	7.5%	11.4%	14.7%	22.3%	29.9%	38.6%	47.7%
$P(50)$	5.0%	5.0%	5.6%	7.6%	12.6%	18.5%	28.5%	38.3%
$P(N_R)$	5.0%	5.0%	5.0%	5.3%	6.5%	9.2%	19.7%	34.3%
$R_P(0.5)$	5.0%	5.0%	5.1%	6.3%	7.4%	7.8%	9.4%	14.7%
$R(100)$	5.0%	5.0%	5.5%	7.5%	8.0%	11.7%	22.6%	34.0%

Table 6.22: The performance measures of IMatch, tested on the Washington database.

6.24 for the queries of the COIL-20 and the Washington database respectively, about the same thing is noticed. The results for the COIL-20 database are very good, while the results for the Washington database fall a little bit behind.

All five queries that were picked from the COIL-20 database are retrieved very well: the performance of the retrieval of the vehicle from class 19 is worst of these queries; out of the 20 images that were retrieved first, 12 were images from the same class. The other queries are retrieved very well.

The six queries that were picked from the Washington database are retrieved not quite as well: the performance for some queries is not good, especially the image from the class Iran. Actually only the images from the class Columbiagorge and Greenland are retrieved very well, the others have mediocre performance measures.

6.4.3 Comparison

The RDTA is used to compare IMatch with the generalised pattern spectra and multi-scale connectivity. Since for this assessment measure, too, all the ranks of the relevant images have to be known, the missing ranks will have to be estimated again.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Duck	1	0.1318	1.0	0.60	0.4366	0.4366	0.4648
Vehicle	1	0.1634	0.6	0.42	0.3521	0.1831	0.3944
Tub	1	0.0073	1.0	1.0	0.9859	1.0	0.9859
Racecar	1	0.1512	0.9	0.52	0.3944	0.3803	0.4225
Duck	1	0.2046	0.8	0.48	0.3662	0.3521	0.3944

Table 6.23: The performance measures for the queries that were hand-picked in sections 6.2.1 and 6.2.3, retrieved by IMatch.

	Rank_1	$\widetilde{\text{Rank}}$	$P(20)$	$P(50)$	$P(N_R)$	$R_P(0.5)$	$R(100)$
Campusinfall	1	0.1416	0.2500	0.2000	0.2128	0.0638	0.3404
Yellowstone	1	0.2688	0.1500	0.1200	0.1064	0.0638	0.2128
Columbiagorge	1	0.3122	0.1000	0.1600	0.1707	0.0366	0.2195
Iran	157	0.6429	0	0	0	0.0208	0
Australia	1	0.3433	0.0500	0.0400	0.0690	0.0690	0.2069
Greenland	1	0.2362	0.6500	0.6400	0.3780	0.2165	0.2008

Table 6.24: The performance measures for the queries that were hand-picked in sections 6.3.1 and 6.3.3, retrieved by IMatch.

The RDTA for the COIL-20 database is very dependant on the method used for determining the missing ranks of the relevant images. When the average case is used, the multi-scale connectivity appears to be more effective: in 60% of the queries, the multi-scale connectivity method is more effective, in 40% of the queries IMatch is, while the multi-scale connectivity method is at least twice as effective as IMatch in 35% of the queries (compared to IMatch being twice as effective in only 19% of the queries). When the best-case estimation is used, however, IMatch is more effective for about 63% of the queries, and the multiscale connectivity method is more effective in only 31% of the queries. IMatch is at least more than twice as effective in about one third of the queries, and the multi-scale connectivity method is at least twice as effective in only 11% of the queries.

The generalised pattern spectra are about equivalent to IMatch when the average case is used to determine the missing ranks. When the best-case is used, however, IMatch is far more effective. For both methods (the generalised pattern spectra as well as the multi-scale connectivity), it seems better to look at the other performance measures, when comparing IMatch to them. From these measures, the only measure that shows great differences between the best-case scenario and the worst-case scenario is the normalised average rank. The other values differ slightly, but not significantly much. Therefore, a more accurate statement can be made about these values than the RDTA. From the performance measures shown in table 6.21 and the performance measures belonging to the queries that were hand-picked from the COIL-20 database (in table 6.23), it can be concluded that IMatch is more effective than the generalised pattern spectra. The performance of the multi-scale connectivity method approached IMatch, but is yet not quite as good. IMatch outperforms both methods on the COIL-20 database.

For the Washington database, the difference between the average estimation of the missing ranks and the best-case estimation is not that large. Therefore, the RDTA can safely be used for comparison. The

difference between the Washington database are in favour of the generalised pattern spectra and multi-scale connectivity. When using the average estimation of the missing ranks of the relevant images, the generalised pattern spectra are more effective in 70% of the queries, and in 85% of the queries, the multi-scale connectivity method is more effective than IMatch. Even when the rank list is estimated according to the best-case scenario, the generalised pattern spectra and multi-scale connectivity are are effective: the first is more effective in 64% of the queries, the latter is more effective in 75% of the queries.

Moreover, more than a quarter of the queries are retrieved at least twice as effective by the generalised pattern spectra, while the multi-scale connectivity retrieve nearly half of the queries twice as effective. When looking at the performance measures of the queries that were hand-picked, as shown in table 6.24, it is confirmed that IMatch is less effective than the generalised pattern spectra and multi-scale connectivity. The overall performance of these two methods it better, and the performance of most of the mentioned queries is better than when IMatch is used to retrieve these queries.

Overall, it can be said that the generalised pattern spectra and multi-scale connectivity method are at least competitive to existing CBIR-software. The very good performance of IMatch on the COIL-20 database and the somewhat disappointing performance on the Washington database might be explained, by the fact that often, systems are tuned for a certain database, like COIL-20. But even with not tuning or tweaking of the generalised pattern spectra and multi-scale connectivity method, their performance is competitive even for the COIL-20 database.

Chapter 7

Conclusions

In this thesism three extensions were presented to the standard area pattern spectrum. These three extensions were tested to see how they perform as CBIR system. Furthermore, the area pattern spectrum and a commercial application, IMatch, were tested as CBIR-system too, in order to compare the three extensions with a basic principle (the area pattern spectrum) and a state-of-the-art application (IMatch).

First, the generalised pattern spectrum was discussed. This extension of the area pattern spectrum makes use of Hu's moment invariants in order to incorporate spatial information besides the size information of components in an image (which is extracted by the area pattern spectrum). The size information is gained through the use of moment m_{00} in addition to Hu's moment invariants, the spatial information is gain through the use of higher order moments. Furthermore, an algorithm for the computation of the generalised pattern spectra for gray-scale images was developed. It is shown in section 3.4 that these generalised pattern spectra *can* distinguish between certain images that the area pattern spectra cannot separate.

Second, the spatial size distribution was discussed. This method extracts the spatial information by translating connected components within an image by a certain vector h; the intersection of the specific component with the translated component is compared to the intersection of the granulometric transform of the specific component with the granulometric transform of translated component. Although it is shown in section 4.2 that these spatial size distributions *can* distinguish between certain images that cannot be separated by the area pattern spectrum, it is also shown in the same section, that the spatial size distributions do not discriminate between images solely based on shape. When the spatial component and size component of two objects are the same, but the shape of these objects differs, the spatial size distributions for these two objects will be the same. The generalised pattern spectra *can* distinguish between such objects.

The third extension that was discussed is based on multi-scale connectivity. By applying consecutive clustering based connected openings on a certain image, and determining the area pattern spectrum for each resulting image, spatial information about the original image is obtained. If two peak components are close together, they might become connected by a clustering based attribute opening, and hence, the area pattern spectrum changes. In section 5.6 is is shown that certain images that cannot be distinguished by the area pattern spectrum in fact *can* be distinguished by this method.

After discussing the three extensions are presented, a framework was developed in which these three methods were tested as CBIR-system. Unfortunately, no such framework is available as standard. That is why nearly all evaluations of CBIR-systems in many scientific articles are of limited value, since

two performance evaluations are not comparable if the framework differs, and hence, there cannot be said if one systems performs better than another. In section 6.1, such a framework is introduced and explained. Performance measures are discussed and an assessment measure is introduced, with which two CBIR-systems can be compared quantitatively. These measures together are used to compare the three extensions of the area pattern spectrum with each other, with the area pattern spectrum itself, and with IMatch, a commercial CBIR-application. Furthermore, two databases are introduced, on which these comparisons are performed: COIL-20 and the Washington database.

Next, these experiments are performed. From these experiments it becomes clear that the spatial size distributions are not suited for content-based image retrieval. The results are very poor for the COIL-20 database, as well as the Washington database. Furthermore, it can be seen, that the bin size is an important variable: the bin size determines the number of features, and because a large number of features is impractical, a trade-off has to be made between the number of features and the performance. When using histograms as a simple method of reducing the number of features, the generalised pattern spectra are competitive with the third extension, based on multi-scale connectivity, however, the latter method *is* significantly better than the generalised pattern spectrum, on both the COIL-20 database and the Washington database. The extra information that is gained by extending the area pattern spectrum is of value: both method outperform the regular area pattern spectrum as content-based image retrieval system. Finally, both methods are at least competitive to IMatch; for the COIL-20 database, IMatch performs a little bit better, but for the Washington database, both methods perform better, especially the method based on multi-scale connectivity.

The results that are based on the Washington database, however, should be considered a bit more closely. This database is classified into 22 different classes. This classification is quite subjective: if 10 different persons would perform this classification, it is very likely that 10 completely different classifications would be obtained. Therefore, besides looking at the performance and assessment measures, a closer look should be given to concrete queries, in order to give a more extensive argumentation about which system is better.

Overall it can be concluded that the generalised pattern spectra and the method based on multi-scale connectivity are absolutely promising as content-based image retrieval systems. In their most basic form, they are at least competitive, and maybe even better than commercial software, and both methods can be implemented quite effectively. Some work, however, has to be done on both methods. One of the major disadvantages is the lack of colour; they are both based on morphological operators, and these are not very suitable for handling colour-images. One possibility of incorporating colour, is to add a colour-histogram. Another method is described in [2], and involves colour size distributions. Another thing that has to be looked into, is the bin size. Obviously, the bin size is an important variable. However, when the bin size is set too small, a huge amount of features is the results. Therefore, a method has to be found that effectively and efficient can reduce the number of features. This might be done by Principal Component Analysis (PCA) [16], Multi-Dimensional Scaling (MDS) [44], or some other method. Further more, the maximum area of the peak components has to be looked into. This maximum size might have some influence on the recognition of the large structures. A solution for incorporating a larger maximum size, while still maintaining a rather low number of features, is the use of a logarithmic binning, as was used to determine the histograms. This way, for larger peak components, fewer bins are created, since large peak components simply occur less often than small peak components.

Bibliography

[1] P. Aigrain, H. Zhang, and D. Petkovic. Content-based representation and retrieval of visual media: a state-of-the-art review. *Multimedia Tools and Applications*, 3(3):179–202, 1996.

[2] J. Angul and J. Serra. Morphological color size distributions for image classification and retrieval. In *Advanced Concepts for Intelligent Vision Systems*, pages 46–53, Ghent, Belgium, 2002.

[3] G. Ayala and J. Domingo. Spatial size distributions: applications to shape and texture analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(12):1430–1442, 2001.

[4] R. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, Princeton, 1961.

[5] J.A. Black, G. Fahmy, and S. Panchanathan. A method for evaluating the performance of content-based image retrieval systems based on subjectively determined similarity between images. In *Proceedings of the International Conference on Image and Video Retrieval*, pages 356–366. Springer-Verlag, 2002.

[6] U. Braga-Neto and J. Goutsias. A multiscale approach to connectivity. *Comp. Via. Image Understand.*, 89(1):70–107, 2003.

[7] E.J. Breen and R. Jones. Attribute openings, thinnings and granulometries. *Computer Vision and Image Understanding*, 64:377–389, 1996.

[8] A. Dimai. Assessment of effectiveness of content nased image retrieval systems. In *Proceedings of the Third International Conference on Visual Information and Information Systems*, pages 525–532, 1999.

[9] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video ontent: the qbic system. *Computer*, 28(9):23–32, 1995.

[10] J. Flusser and T. Suk. Construction of complete and independent systems of rotation moment invariants. In *Computer Analysis of Images and Patterns 2003*, pages 41–48, Groningen, The Netherlands, August 2003.

[11] C. Frankel, M.J. Swain, and V. Athitsos. Webseer: an image search engine for the world wide web. Technical report, University of Chicago, 1996.

[12] Th. Gevers and A.W.M Smeulders. Image search engines an overview, 2003. University of Amsterdam.

[13] A. Goodrum. Image information retrieval: an overview of current research. *Informing Science*, 3(2):63–66, February 2000.

[14] H.J.A.M. Heijmans. Theoretical aspects of graylevel morphology. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6):568–582, 1991.

[15] M.K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions in Information Theory*, IT-8:179–187, 1962.

[16] I.T. Jolliffe. *Principal component analysis*. Springer, 1986.

[17] P. Maragos. Pattern spectrum and multiscale shape representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(7):701–715, 1989.

[18] G. Matheron. *Random Sets and Integral Geometry*. Wiley, New York, 1975.

[19] A. Meijster and M.H.F. Wilkinson. Fast computation of morphological area pattern spectra. In *Proceedings of Int. Conf. Image Proc.*, pages 668–671, Thessaloniki, Greece, October 7–10 2001.

[20] R. Milanese and M. Cherbuliez. A rotation, translation, and scale-invariant approach to content-based image retrieval. *Visual Communication and Image Representation*, 10(2):186–196, June 1999.

[21] H. Müller, S. Marchand-Maillet, and T. Pun. The truth about corel – evaluation in image retrieval. In *Proceedings of the International Conference on the Challenge of Image and Video Retrieval*, pages 38–49, 2002.

[22] H. Müller, W. Müller, D.McG. Squire, S. Marchand-Maillet, and T. Pun. Performance evaluation in content-based image retrieval: overview and proposals. *Pattern Recognition Letters*, 22(5): 593–601, 2001.

[23] S.A. Nene, S.K. Nayar, and H. Murase. Columbia object image library (coil-20). Technical Report CUCS-006-96, Columbia University, 1996.

[24] G.K. Ouzounis and M.H.F. Wilkinson. Second-order connected atribute filters using max-trees. In *Proc. Int. Symp. Mathematical Morphology 2005*, pages 65–74, April 18–20 2005.

[25] J. Puzicha, J.M. Buhmann, Y. Rubner, and C. Tomasi. Empirical evaluation of dissimilarity measures for color and texture. In *ICCV '99: Proceedings of the International Conference on Computer Vision*, pages 1165–1172, Washington, DC, USA, 199. IEEE Computer Society. ISBN 0-7695-0164-8.

[26] Y. Rubner, C. Tomasi, and L.J. Guibas. A metric for distributions with applications to image databases. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, pages 59–66, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 81-7319-221-9.

[27] P. Salembier, A. Oliveras, and L. Garrido. Anti-extensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7(4):555–570, apr 1998.

[28] J. Serra. *Image Analysis and Mathematical Morphology*, volume I. Academic Press, London, England, 1982.

[29] J. Serra. Connectivity on complete lattices. *Journal of Mathematical Imaging and Vision*, 9(3): 231–251, 1998.

[30] A.W.M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain. Content-based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, 2000.

[31] J.R. Smith and S. Chang. Image and video search engine for the world wide web. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 84–95, 1997.

[32] M. Smith. *Neural networks for statistical modeling*. John Wiley & Sons, Inc., New York, NY, USA, 1993. ISBN 0442013108.

[33] D.McG. Squire, W. Müller, H. Müller, and J. Raki. Content-based query of image databases, inspirations from text retrieval: inverted files, frequency-based weights and relevance feedback. In *The 11th Scandinavian Conference on Image Analysis*, pages 143–149, Kangerlussuaq, Greenland, June 7–11 1999.

[34] S.R. Sternberg. Grayscale morphology. *Comput. Vision Graph. Image Process.*, 35(3):333–355, 1986.

[35] R.E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, 22(2): 215–225, 1975.

[36] C.J. van Rijsbergen. *Information retrieval*. Dept. of Computer Science, University of Glasgow, 1979.

[37] R. Veltkamp and M. Tanase. Content-based image retrieval systems: a survey. Technical report, Utrecht University, 2000.

[38] C.C Venters and M. Cooper. A review of content-based image retrieval systems, 2000. Manchester Visualization Centre, University of Manchester.

[39] L. Vincent. Morphological area opening and closing for grayscale images. In *Proc. NATO Shape in Picture Workshop*, pages 197–208, September 1992.

[40] L. Vincent. Grayscale area openings and closings: their applications and efficient implementation. In *Proc. EURASIP Workshop on Mathematical Morphology and its Application to Signal Processing*, pages 22–27, Barcelona, Spain, May 1993.

[41] L. Vincent. Granulometries and opening trees. *Fundamenta Informaticae*, 41(1-2):57–90, 2000.

[42] M.H.F. Wilkinson. Generalized pattern spectra sensitive to spatial information. In *Proc. 16th Int. Conf. Pattern Rec.*, volume 1, pages 21–24, Québec City, Canada, August 11–15 2002. IEEE Computer Society.

[43] M.H.F. Wilkinson and J.B.T.M. Roerdink. Fast morphological attribute operations using tarjans union-find algorithm. *Proc. Int. Symp. Mathematical Morphology 2000*, pages 311–320, June 2000.

[44] F.W. Young. *Multidimensional scaling: history, theory and applications*. Erlbaum, 1987.