Chapter 2

Background Work

This Chapter provides general background material appropriate for the problems and potential applications outlined in Chapter 1. A basic introduction to some of the typical and very specific approaches used in image recognition leads into the more general topics of shape representation, feature vectors, clustering and classification techniques.

The automatic visual inspection problem, that of detecting anomalies, defects or faults in materials, is outlined. This is followed by an introduction to spatial frequency methods and the design of filters necessary for much research in texture analysis and image segmentation. Texture theories, discrimination, retrieval and representation are briefly outlined, prior to numerous segmentation techniques. These include histogram methods to deal with tone images, edge detection methods, region growing and split merge algorithms, frequency filtering and energy models such as Markov random fields (MRFs). Image segmentation is important since it directly tackles the decomposition of images into sub-images based on image content.

The representation of colour for the purpose of image processing or recognition is discussed along with segmentation algorithms specific to colour. This naturally leads to the more general and rapidly growing topic of content based image retrieval (CBIR). The chapter is concluded with discussion on syntactic image recognition.

In summary this chapter provides a background into some of the standard and more recent approaches to intelligent image processing or image recognition. Hence, it does not deal with standard image processing topics such as image enhancement or restoration. Details pertaining to these is omitted. The scope is also restricted to individual two dimensional images and typically grey scale images. Hence details concerning optical flow techniques, three dimensional reconstructions and object location are inappropriate.

Overall this chapter serves to provide a basic understanding and overview of the literature in terms of pattern recognition issues, defect detection, texture, segmentation, colour representation, CBIR and syntactic methods.

2.1 Traditional approaches to and fundamental issues of image recognition and classification

Perhaps one of the more common methods used to tackle the recognition or classification problem is to use an off the shelf tool, such as a neural network,[1] with very little thought to what is actually going on. An extreme example is to take a set of single digits, each a 20 by 20 pixel binary image, and use the raw pixel values as inputs into a standard multilayer perceptron. This amounts to taking a set of labelled vectors in a 400 dimensional feature space and attempting to separate each class, or digit, by a number of hyperplanes. There is no consideration for translational, rotational or scale invariances, although these can be learnt by brute force given enough samples.

A realistic neural network example [15] involves the identification of facial features (eyes, nose and mouth). This is achieved through an intelligent and reasonably well thought out local feature extraction process to obtain input vectors. Back-propagation is then used to train a neural network with a hidden layer of 10 neurons, with 97 of 101 images used for training and 4 for testing. The justification for the need for a hidden layer and its properties is most interesting, "These intuitions came from reading descriptions of other successful neural networks." Finally, the network was trained approximately 500 times through the training data to achieve stability.

This example represents a typical, but well engineered, image recognition project. The reasons for success and failure are hidden inside the heavily trained neural network, making the *intelligent* processing extremely difficult to analyse and improve.

2.1.1 Ad Hoc and specific examples

The main advantage of the neural network outlined above is its ability to solve the problem in a general way without making assumptions about the nature of its inputs. The other extreme is to hand-craft methods specific to one application, making assumptions that may not be justifiable. There are countless examples, a few of which are outlined below.

In [16], the segmentation of a traffic scene is achieved by locating "all detected red and yellow objects" and grey objects in an attempt to find traffic signs. The results appear impressive, although there are no examples that contain cars or flowers which are red or yellow. Similar techniques are used in [17] and [18] to automatically locate both shaded and non-shaded road through analysis in different colour spaces. Again the results are reasonable, although the sky is often misclassified as road and the images presented are conveniently free of buildings with natural scenes surrounding the roads.

In both examples techniques have been derived through a manual analysis of the colour spaces using assumptions specific to the problem being addressed.[2] These systems are not

[1] For an enlightening history and general introduction to the properties of several types of neural networks refer to [14].

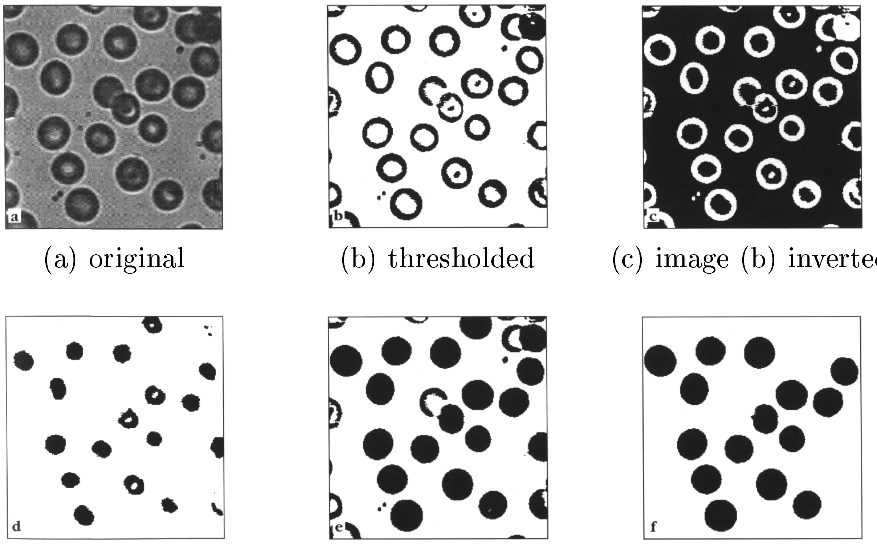
[2] In these cases the assumptions may be considered unreasonable due to the uncontrolled operating environment. Similar methods are used in the identification of Southern Right whales from aerial pho-

general and cannot be directly applied to other problems without making fundamental changes.

Object segmentation and counting

Specific approaches are often used in the segmentation of binary or grey scale images into disjoint objects. In optical character recognition (OCR) systems characters are commonly separated by examining pixel connectedness, in the extreme case looking for vertical lines of white pixels [19]. Object counting systems make use of similar techniques and may employ mathematical morphology [20, p157–264], convolution or local histograms.

A fast scheme for separating white blood cells [21] uses local histograms of the images' green component. The histograms are taken within a window size specific to the application. These histograms are used with a tailor-made algorithm to zoom in on the centre of cells. This convoluted algorithm is far from ideal as concluded in [21], "It should be noted that some problems may occur in the case of touching white blood cells." This type of problem is illustrated in Figure 2.1 showing an alternative, simple, yet specific method for counting objects. One of the more general solutions is to decompose each shape into near convex parts as done in [22] in which results are agreeable with human intuition.



(a) original — (b) thresholded — (c) image (b) inverted

(d) black sections of (c) not on the edge — (e) image (b) logically ORed with (d) — (f) round objects not touching the edge

FIGURE 2.1: Light microscope image of red blood cells. Taken directly from [23] with descriptions shortened.

Another object counting system which is far less reliable and provides only a rough estimate is presented in [24]. This extreme case suggests that "if lighting conditions are chosen carefully, most industrial applications can be reduced to histogram based algorithms and

tographs [11] where the environment is more suitably controlled.

cellular logic operations." The system which counts, or rather estimates, bottles passing on a conveyor belt counts the bottles by determining connected components on a small input strip. Even with strict lighting control the system suffers from reflections, spots on the conveyor belt and other miscellaneous phenomena.

A final object counting and sizing system uses aerial photographs to count and size citrus groves for tax purposes [25]. Three approaches are used, a discrete method similar to counting white blood cells, the biomass approach and a hybrid combination. The biomass method is extremely simple, estimating the number of objects by the total object area and size of each object, i.e. $N = T/A$. The hybrid method is more interesting, attempting to locate trees individually, but makes use of a template to identify individual trees from connected objects. In the system presented the template, referred to as a floating circle, is explicitly defined for each image by a supervised operator prior to processing.

2.1.2 Template matching

Many image recognition systems benefit from assuming prior knowledge of an image's content. The use of the floating circle method to separate connected objects [25] suggests the advantage of using predefined templates to search for specific objects or characteristics.

In general, a match between a template in an image can be achieved by minimising the squared difference at a particular template location, which is equivalent to maximising the cross-correlation (or *area correlation*) as shown below.

$$\begin{aligned}
\min(\text{diff}^2(\mathbf{x},\mathbf{y})) &= \min(\textstyle\sum_i \sum_j [\mathbf{T}(i,j) - I(i-\mathbf{x}, j-\mathbf{y})]^2) \\
&= \min(\textstyle\sum_i \sum_j |\mathbf{T}(i,j)|^2 + \sum_i \sum_j |\mathbf{I}(i,j)|^2 \\
&\qquad -2\textstyle\sum_i \sum_j \mathbf{T}(i,j)\mathbf{I}(i-\mathbf{x}, j-\mathbf{y})) \\
&= \max(\textstyle\sum_i \sum_j \mathbf{T}(i,j)\mathbf{I}(i-\mathbf{x}, j-\mathbf{y}))
\end{aligned}$$

Although the cross-correlation attains its maximum value when the displaced template coincides with the observed image, there are still problems with multiple perfect matches or when one template is a sub-image of another. An example of the first case is in cutting non-defective regions of leather given a binary *defect map*. In [26] this is not properly solved, but rather strips are cut according to each template match on a first come first served basis and hence the solution is in general non-optimal.

A practical example of template matching which suffers from the second problem mentioned above is the recognition of traffic signs in real-world scenes [27]. This system deals with invariances to size, rotation around the viewing plane and rotation into the viewing plane, as well as translation. The five dimensional solution space is handled using simulated annealing, reducing the search time from more than 10 hours to 15 seconds at the expense of an optimal solution. The authors make use of a normalised correlation and strongly advocate the use of template matching in recognition tasks. The results are impressive given the noisy images, although the *slow* sign is more often than not misclassified as a *yield* sign. This is because the triangular *yield* sign is a subset of the diamond shaped *slow* sign.

One of the more sophisticated systems which uses extensions to template matching is the face recognition system at Massachusetts Institute of Technology (MIT). In order to recognise faces under varying poses, templates are used to locate facial features such as eyes and nose [28]. Subsequent work uses interpolation in a higher level feature space to provide mechanisms for both recognising and reconstructing an image based on pose, with intended applications in low bandwidth teleconferencing [29]. Further extensions involve recognising individual faces from previously unseen views [30] and shape and textural considerations [31].

Another face recognition system uses an adaptive Hough transform to locate cheeks, chins and eyebrows [32]. This system makes assumptions about the colour of skin in order to locate edges and then further falsely assumes facial symmetry. It is significantly inferior to the MIT systems mainly because it relies on the location of clear edges. Results indicate the successful location of *individual* features of 50–70%.

The Hough transform

The Hough transform is basically a glorified template matching system used to locate a specified family of curves or shapes in an image. It converts the shape finding problem to a maximisation problem in the appropriate restricted parameter space or Hough space, given a prespecified shape (e.g. a line, circle or ellipse). The operation is global, sensitive to image resolution (object thickness), susceptible to noise and incidental features in real images, and computationally expensive for objects described with more than a few variables.

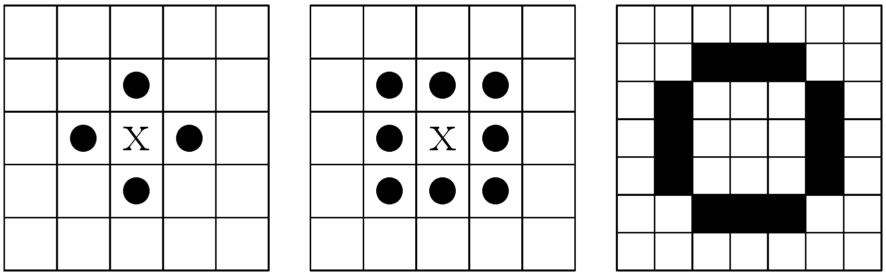
Typically the Hough transform is applied to finding lines (not line segments of curves) by counting the pixel match of all possible lines or equivalently considering each pixel as a curve in the Hough accumulator space. The *parameterisation* used for lines is either $y = mx+c$ or $\rho = x\cos\theta + y\sin\theta$ since it can cope with vertical lines. In fact neither of these is a parametrisation, as discussed in [33]. The general equation of a line $(ax + by + c = 0)$ normalised to make $(a^2 + b^2 + c^2 = 1)$ produces the surface of a sphere in $\mathbb{R}^3$ with triplets (a, b, c) and $(-a, -b, -c)$, i.e. opposite points on the sphere, corresponding to the same line. This is equivalent to the topological space $\mathbb{RP}^2$ (the real projective plane) which cannot be parameterised with two numbers or even embedded in $\mathbb{R}^3$ as a result of the Jordan Curve Theorem [34, p11.22]. In practice this is slightly problematic, since the accumulator space is typically specified by $\rho \in [0, \infty)$ and $\theta \in [0, \pi)$ where $\theta = 0$ forms approximately the same line as $\theta = \pi^-$. This discontinuity affects the maximisation problem which requires the examination of local regions.

Further details and variations can be found in [20, p578–588] and [35, p240–264]. A comparison between using the Hough transform and the methods for stroke detection outlined in Section 1.1.2 is presented in [8] and [33].

2.1.3 Shape representation

The description of shape is one of the foremost issues in pattern recognition. Various aspects of this domain have been discussed, but no satisfactory general theory exists [36]. Ideally a representation should be robust, have desirable invariances, support meaningful metrics and posses features which support clustering or grouping. As well as supporting invariances, the representation should provide the option to distinguish between objects which differ in a relevant way. For example, although Zernike moments have an inherent rotational invariance, they are separable into two components one of which describes rotation.

Before considering shape representations, definitions of connectedness and disjoint region identification are addressed. Figure 2.2 illustrates 4-connectedness, 8-connectedness and the potential problem or paradox of 8-connectedness. This paradox [37, p358] is overcome in [38] by using 8-connectedness for foreground, or black pixels, and 4-connectedness for background pixels. An alternative, which is not addressed in the literature, is to use 6-connectedness with connections along either one of the diagonals. This would separate Figure 2.2(c) into 1 background and 2 foreground regions. A number of different algorithms for "connected component labelling" are presented in [20, p28–48].



(a) 4-connectedness (b) 8-connectedness (c) paradox

FIGURE 2.2: Pixels (•) that are 4-connected and 8-connected to the centre pixel (X) and a binary image which exhibits the connectedness paradox. Using 8-connectedness (c) contains 1 connected black region and 1 connected white region, whereas using 4-connectedness it contains 4 black regions and 2 white regions. For grouping connected objects neither is desirable.

Many methods for describing shape exist, categorised broadly as region descriptors or boundary descriptors. These are discussed in turn. Throughout it is assumed that connected objects are being measured and connectedness issues are overlooked.

Region descriptions

Low order geometric moments are arguably the most common, intuitively meaningful and general methods for simple shape descriptions. These can be defined for a continuous finite region in an arbitrary dimension, although these are outlined here for the two dimensional case for simplicity. The zeroth, first and second order central moments correspond to the area, centroid and deviation, elongation or spread of a region. In the continuous and discrete two dimensional the $(p+q)^{\text{th}}$ moments applicable to binary and grey scale image regions are:

$$\mathbf{M}_{p,q(absolute)} = \int\int f(\mathbf{x},\mathbf{y}) x^p y^q dx dy \quad \forall\, p,q = 0,1,2,...$$

$$\mathbf{M}_{p,q(absolute)} = \sum f(\mathbf{x},\mathbf{y}) x^p y^q \quad \forall\, p,q = 0,1,2,...$$

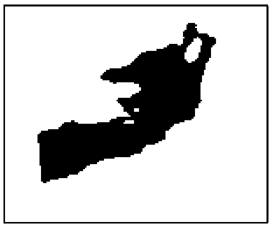
For non-zeroth order moments the normalised moment is more useful, for example the centroid is defined by $\overline{x} = \frac{\mathbf{M}_{1,0}}{\mathbf{M}_{0,0}}$ and $\overline{y} = \frac{\mathbf{M}_{0,1}}{\mathbf{M}_{0,0}}$.

$$\mathbf{M}_{p,q(norm)} = \frac{\mathbf{M}_{p,q}}{\mathbf{M}_{0,0}} = \frac{\sum f(\mathbf{x},\mathbf{y}) x^p y^q}{\sum f(\mathbf{x},\mathbf{y})} \quad \forall\, p,q = 0,1,2,...$$

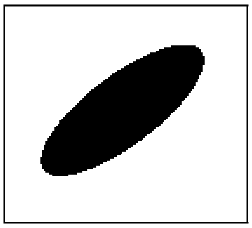
To achieve translational invariance, points are considered with respect to the centroid using the *central* geometric moment.

$$\mathbf{M}_{p,q} = \frac{\sum f(\mathbf{x},\mathbf{y})(x-\overline{x})^p (y-\overline{y})^q}{\sum f(\mathbf{x},\mathbf{y})} \quad \forall\, p,q = 0,1,2,...$$

The second order central moments form the covariance matrix of a set of points where $\mathbf{M}_{2,0} = \sigma^2_{\mathbf{x}}$ and $\mathbf{M}_{0,2} = \sigma^2_{\mathbf{y}}$ measure the variance or spread of the objects in the $\mathbf{x}$ and $\mathbf{y}$ directions and $\mathbf{M}_{1,1}$ is the covariance. These values are frequently used in statistics.



(a) object



(c) simple reconstruction

represents	order	value
number of pixels	0th	$N = 912645$
centre of mass	1st	$\begin{bmatrix} \mu_{\mathbf{x}} \\ \mu_{\mathbf{y}} \end{bmatrix} = \begin{bmatrix} 62.59 \\ 55.70 \end{bmatrix}$
covariance matrix	2nd	$\mathbf{C} = \begin{bmatrix} 704.9 & -433.9 \\ -433.9 & 458.0 \end{bmatrix}$

(c) geometric moments

FIGURE 2.3: Shape reconstruction of a region using central geometric moments.

Figure 2.3 provides an illustration of the low order geometric moments of a simple shape. Figure 2.3(c) shows the information captured by just 6 real numbers. For alternative descriptions of geometric moments refer to [14, p83–86], [40, p427–430], [37, 377–381] and [41, p514–517]. For an optical implementation see [42].

Zernike moments offer an alternative to geometric moments which have been successfully applied to character recognition [43]. First, the shape is normalised to lie within the unit disc with its centroid at the origin. A set of *Zernike moments of order p with repetition q* arise by integrating over the disc with the Zernike basis [14]. The Zernike basis is formed by applying the Gram-Schmidt Orthogonalisation Process [44, p273–280] to the basis functions $V_{p,q} = r^p e^{\mathbf{j}q\theta}$ and is of the form $Z_{p,q}(\rho,\theta) = R_{p,q}(\rho)e^{\mathbf{j}q\theta}$ for a radial basis function R. [45][3] The obvious advantage of this form is the separation into a radial component and a phase component allowing rotationally invariant comparison between shapes. The disadvantage is the number of moments that are required to successfully distinguish between shapes, as illustrated in Figure 2.4. Since the Zernike functions are orthonormal, a deterministic reconstruction illustrates the exact shape captured by this representation.

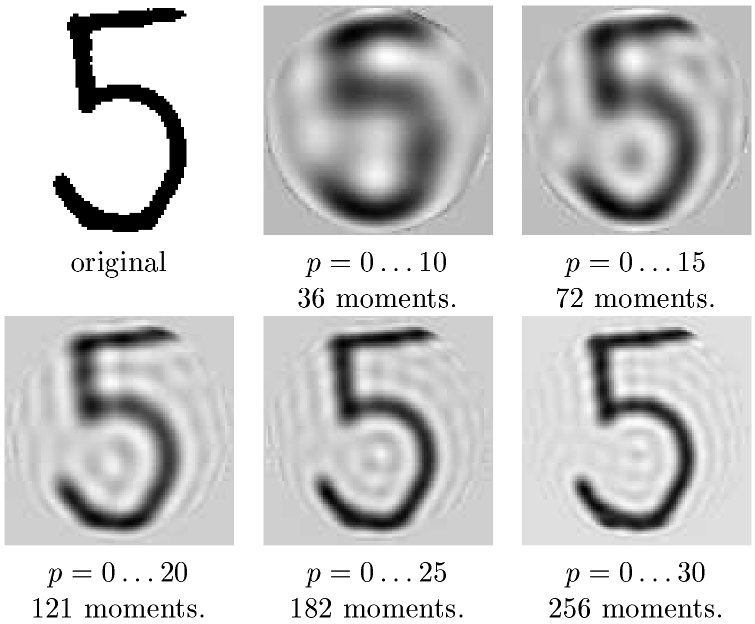
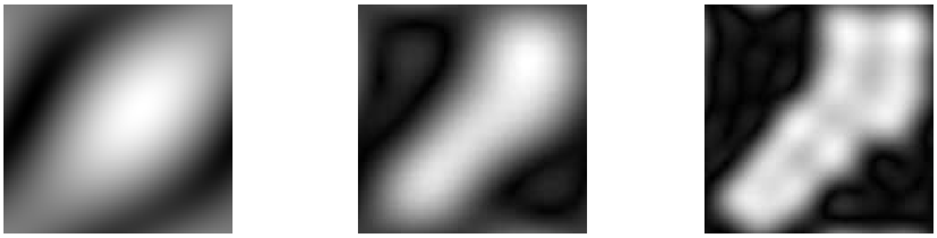


FIGURE 2.4: Shape reconstruction of a digit using Zernike moments. Provided by the author of [43].

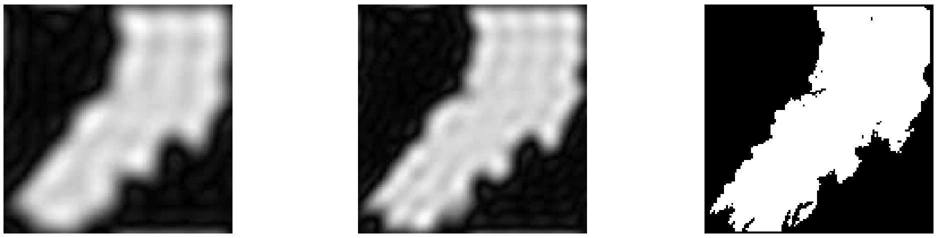
As is apparent from the above, the Zernike basis was designed by using functions with properties to inherently allow rotational separation. In principle any other set of orthonormalised functions could be used. For example, the family of sine and cosine functions of different frequencies could be used, although fitting these to binary shape information

[3] Most of the equations have been omitted since they are very long and less meaningful that the underlying ideas. For a more comprehensive coverage refer to [45, p767–772].

seems non-intuitive and requires many numbers for a reasonable representation. This set of basis functions is used in the Fourier transform. As seen in Figure 2.5 it is not well suited to shape representation since it requires many numbers and does not posses any useful invariances. Perhaps most important to note is the ability to describe shape using an arbitrary set of basis functions.



(a) 4 complex numbers (b) 16 complex numbers (c) 64 complex numbers



(a) 144 complex numbers (b) 256 complex numbers (c) original shape

FIGURE 2.5: Shape reconstruction of a region using low frequency FFT coefficients.

Representing shape in binary images using quad-trees provides a means for perfect shape reconstruction and appears to be efficient in terms of compression, but is not useful for comparing different shapes. In a quad-tree each region, beginning with the original shape, is determined to be either black (0), white (1) or a combination of the two in which case it is separated into 4 regions each represented by the same process. The quad-tree may also be lossy by accepting regions which are mostly black or white.

Several miscellaneous shape features are outlined in [37] including perimeter, area, radii, number of holes (topological considerations [41, p504–506]), corners, bending energy, roundness, symmetry, centre of mass, orientation, bounding rectangle, best fitting ellipse and eccentricity. Of these not directly covered by moment techniques, radius features are widely used and tend to lead to topological and symmetry considerations.

Jain [37] defines R_{min} and R_{max} simply as "the minimum and maximum distances, respectively, to boundary from the center of mass". A more complete description is to measure the radii from the centroid at a number of evenly spaced angles [46]. It is clear that this measure is only suitable for convex, or *near convex*, shaped objects. Furthermore, most shape representation and recognition systems are far more reliable when limited to objects which are convex. Consequently numerous algorithms have been derived to split objects into convex parts which are either disjoint [38] [22] or allow overlap between parts [47]. For further discussion of the representation of convex parts refer to [36] [22] and [47].

Boundary and one dimensional descriptions

Chain codes which have similar properties to quad-trees are commonly used. They use the 8-connectedness of the boundary, storing a value between 0 and 7 according to the direction between adjacent boundary pixels. This representation allows for perfect shape reconstruction, but is not particularly useful for comparing shapes, lacking all but translation invariance and being susceptible to noise.

Skeleton and thinning algorithms [35, p149–162] and other methods based on mathematical morphology [20] are also common. Although these seem promising they lack robustness, often producing significantly different results for similar objects as illustrated in Figure 2.6.

FIGURE 2.6: The lack of robustness of skeletonisation for representing shape.

Using the Fourier Transform of the boundary positions (as a one dimensional complex signal with points of form $\mathbf{p} = \mathbf{x} + \mathbf{j}\mathbf{y}$) offers an attractive representation. The representation is invariant to translation by considering the DC component (equivalent to the centre of mass), can be made scale invariant by choosing a predefined number of boundary points and has rotational invariant properties by separating phase from magnitude. Furthermore it is robust to subtle shape changes and can be represented by just a few numbers, depending on the desirable quality of reconstruction. Current standard content based image retrieval systems do not make use of this shape representation.

A final representation worth mentioning is the use of edge histograms. As outlined in [48] this is inherently invariant to translation and can be made invariant to size and rotation by histogram normalisation and histogram rotation respectively. Some slightly questionable smoothing functions are used to achieve reliable rotational invariance and the system does not handle matching parts of shapes.

Measurements, robustness and closing comments

The use of *shape metamorphosis* is one of the more novel approaches to comparing two shapes [49]. Essentially one image is morphed to another with a cost, or dissimilarity measure, based on the stretching and bending energies of points. The method is suggested to be physically based and benefits from not requiring extensive shape statistics [49].

Although this provides a measure difference between two shapes it does not allow the grouping or clustering of a number of shapes into a single class. The same can be said about many of the representations outlined above and those that trade this for invariant properties. Geometric moments are one of the few descriptors which given **N** shapes can describe size, centre, elongation and spread in terms on an average or other statistical measures.

The robustness of a description is also important. Geometric moments and boundary Fourier descriptors are robust to noise and small changes, whereas Zernike moments, region Fourier descriptors, quad-trees, chain codes, skeletonisation and morphological techniques are generally not. In fact the Zernike moment is fairly robust once a unit disc has been fitted, but the fitting of this disc is extremely important and generally sensitive to noise.

To summarise, geometric moments which are the most simple and commonly used have desirable properties including simplicity and compactness, invariances, robustness and cluster support. However, meaningful measurements are limited to low order moments where the shape descriptions are simple and they do not handle concave objects well. Edge detection techniques are also common, but highly dependent on the performance and precision of the edge detector. The Fourier boundary descriptor appears to have attractive qualities but is rarely used.

In conclusion the appropriate method for representing shape remains unclear with no conclusive answers emerging. Descriptors tend to be application dependent or specific to broad problem areas such as compression or recognition. In some cases the ability to decompose objects into convex subparts is desirable.

2.1.4 Features and feature vectors

The extraction of image features has been extensively studied for well over 30 years. A "classical approach" [50, p22-30] outlined back in 1970 suggesting that the performance of a pattern recognition system crucially depends on the performance of the feature extractor. No significant additions have been made to this statement since. It further states that;

> "The main guidelines to feature design are that features should be invariant to (or at least insensitive to) irrelevant variations, such as limited amounts of translation, rotation, scale change, etc., while emphasising differences that are important for distinguishing between patterns of different types." [50, p23]

Since then many techniques of automatic feature selection have been developed. These include the use of a feature selection lattice to minimise the overall error of a specific problem [51] [52] and dimension reduction using techniques such as principal components analysis.

A more recent study into "what makes a good feature?" examines a Bayesian or equivalent framework for ascertaining the importance of image features. However, the results suggest that "non-accidental" or "suspicious" characteristics (such as parallel lines) cannot be located by likelihood ratios since extremely low *a priori* probabilities are unlikely to produce a sufficiently high *a posteriori* probability. They hence conclude for this framework to be useful it requires "specifications of the current content". [53]

As suggested above features should ideally have desirable invariances or tolerance and be general, based on image content rather than external assumptions. A thesis by Kovesi [54] directly addresses the need for invariant quantities in images, making use of phase congruence to extract unit-less edge like features without the need for masks, neighbourhoods or any direct pixel or intensity measurements. This is a good example of a general feature, although it is very time consuming and has huge memory requirements compared to conventional edge detectors. Consequently the current implementation is unattractive.[4]

In general feature extraction techniques can be categorised into local and global techniques. Local features are more common in segmentation, texture analysis, edge detection and filtering [55] [56]. Examples of purely global features include distributions and histograms used in simple CBIR systems [57], global low order statistics [58], information content measures pertaining to translation, rotation and overall content [56] and ad hoc measures [59]. Attempts are also made to combine local and global features with considerations for invariances [60]. Other specific methods of feature extraction are outlined as they arise through the remainder of this chapter.

Regardless of the problem, feature vectors should be general both in form and what they measure. Ideally, context or problem specific information should not be included in the feature extraction which, similar to shape representation, should be robust and support invariances. Hence, a point in a Euclidean space ($\mathbb{R}^{\mathbf{N}}$) is an appropriate feature representation. From this problem specific decisions can be made without compromising the generality of the feature extraction.

2.1.5 Clustering

Clustering, often referred to as unsupervised learning, is the process of identifying *clusters* or natural groupings in the feature space. As opposed to classification, the number of classes, or clusters, is typically unknown and the data is unlabelled. Since "no single definition of a *cluster* is universally accepted" [61, p3] and the existence of clusters is often open to interpretation, *exploratory data analysis* is a common technique (i.e looking at the data). Exploratory data analysis is subjective: "The goal is to use whatever tools are available to look at the data, with the emphasis on graphs, charts, projections, and any graphical representation that assists the human visual system and nudges intuition." [61, p10]

Exploratory data analysis

Linear projections are often used to transform the $\mathbf{N}$ dimensional data into two dimensions where it can more easily be viewed. Projections can either be made arbitrarily or by decorrelating features using the eigenbasis determined from covariance matrix of the

[4] The current *Matlab* implementation was unable to process a 256×256 image on a Silicon Graphics Onyx with four R10000 processors and 128 Megabytes of memory.

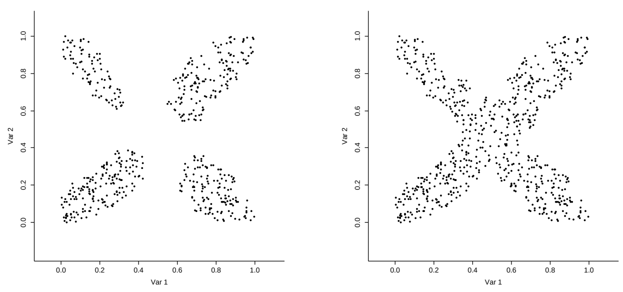
data [62, p26–31] (equivalent to the second order central geometric moments described in Section 2.1.3.) An alternative which is far more useful is to use a computer program such as *xgobi* [63] to rotate the feature space and allow further insight into the data. Feedback and the ability to label data points or regions of the feature space provide powerful tools for examining data and results from both classification and clustering. These techniques are typically not discussed in the literature.

The use of Chernoff faces provides an alternative *graphical* projection. Each face depicts one feature vector with feature elements proportional to, for example, the area of the face, the shape of the face, the length of the nose, the location of the mouth and the curve of the smile. The idea here being that a human is able to observe characteristics from all five (in this case) feature elements simultaneously and determine possible cluster criteria [61, p14–16]. Again, the use of *xgobi* would seem far more appropriate.

The thesis by Roy [64] provides a practical application of exploratory data analysis applied in cluster analysis.

How many clusters?

This is the fundamental question in cluster analysis which cannot be quantitatively answered since there is no accepted definition of a cluster. Figure 2.7(b) contains a number of clusters, although ascertaining how many is both non-trivial and debatable.



(a) 4 clusters (b) 1, 2 or 4 clusters

FIGURE 2.7: An ambiguous number of clusters.

Figure 2.8, adapted from [62, p2], outlines the problem of scale. Debatably this image consists of 4 clusters at the global or higher level or 9 at a lower similarity threshold. Ideally the data should be clustered hierarchically. Again, the problem is ambiguous with no obvious answer and in many cases is influenced by application-specific knowledge.

A "theoretical justified guide" for determining the number of clusters in both Gaussian mixture model and the k-means algorithm is presented in [65] which claims to solve this general problem[5]. A well established and better accepted theoretical method is to use the

[5]The author of this paper made an enlightening and enthusiastic presentation, although many members of the audience remained sceptical. No extensions or applications to the work have been seen.

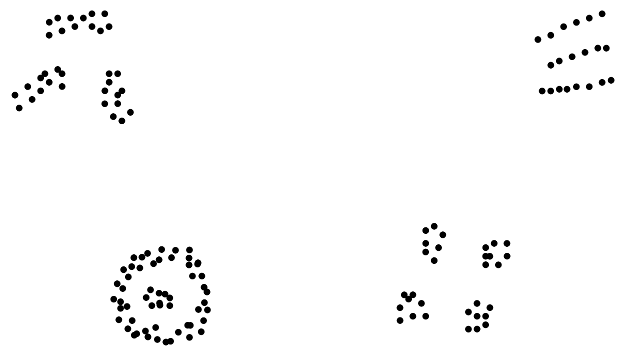


FIGURE 2.8: The dependency of scale on the number of clusters.

Akaike Information Criterion based on minimum description length (MDL) models [14, p158–187] [66]. Other methods include determining the validity of clusters using techniques such as cluster tendency [67] and finding clusters via bootstrapping [68].

Metrics and normalisation

A metric is a distance measurement $\mathbf{d}$ on a set X satisfying the properties:

1. $\mathbf{d}(\mathbf{x}, \mathbf{y}) \geq \mathbf{0} \;\; \forall \mathbf{x}, \mathbf{y} \in X$
2. $\mathbf{d}(\mathbf{x}, \mathbf{y}) = \mathbf{0} \Leftrightarrow \mathbf{x} = \mathbf{y} \;\; \forall \mathbf{x}, \mathbf{y} \in X$
3. $\mathbf{d}(\mathbf{x}, \mathbf{y}) = \mathbf{d}(\mathbf{y}, \mathbf{x}) \;\; \forall \mathbf{x}, \mathbf{y} \in X$
4. $\mathbf{d}(\mathbf{x}, \mathbf{z}) \leq \mathbf{d}(\mathbf{x}, \mathbf{y}) + \mathbf{d}(\mathbf{y}, \mathbf{z}) \;\; \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \in X$

The most common and general unnormalised metric is the Minkowsky metric. The limiting Minkowsky metric is equivalent to the Chebychev metric or maximum component, the second is equivalent to the geometric or Euclidean distance, and the first is equivalent to the *city block* metric or sum of vector elements. [69, p52–53]

$$\lambda^{th} \textit{Minkowsky metric}: \qquad \mathbf{d}_\lambda(x, y) = \left(\sum_{i=1}^{N} |x_i - y_i|^\lambda \right)^{\frac{1}{\lambda}}$$

To account for different measurements and units within a feature vector, normalisation is often used. For example, *z-score* normalisation [61, p9] [62, p23-25] independently scales elements to a mean of 0 and standard deviation of 1. An alternative is to use the quadratic metric [69] based on the covariance matrix which models the distribution of the entire feature space. This metric, which is equivalent to the Euclidean distance under a change of basis by scaling and rotation, brings rise to the Mahalanobis distance between a data point and a premodelled set of feature vectors or cluster. The Mahalanobis distance from a set of features with mean μ and covariance matrix $\mathbf{C}$ is defined as:

$$\textit{Mahalanobis distance}: \qquad \mathbf{d_m}(\mathbf{x}) = \sqrt{(\mathbf{x}-\mu_{\mathbf{x}})^T \mathbf{C}^{-1}(\mathbf{x}-\mu_{\mathbf{x}})}$$

Dendrograms and cluster trees

A dendrogram is a binary tree that depicts hierarchical clustering. A class of algorithms known as *SANH* (sequential, agglomerative, hierarchical, non-overlapping) are generally used to form this type of tree structure [61, p17–21]. A simple example of a cluster tree algorithm is to combine the two *closest* feature vectors into a vector according to the centroid of the original contributing vectors. This process is repeated until a single vector results. This algorithm allows the final tree to be cut at any level, hence resulting in an arbitrary number of clusters. Although this distinction can be made very quickly once the tree has been formed, this type of algorithm is very computationally expensive (order N^4) and hence infeasible for large datasets.

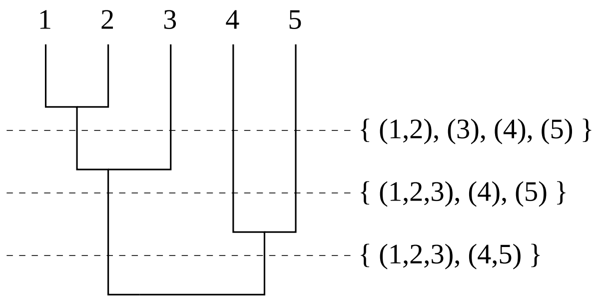


FIGURE 2.9: Dendrogram for five patterns. Reproduced from [61, p17].

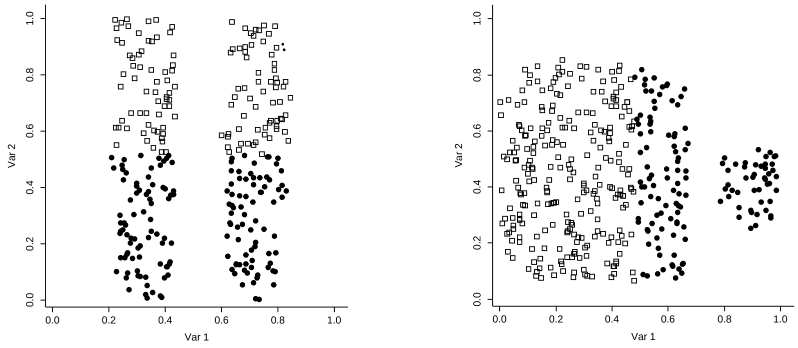
Similar results to dendrograms can be obtained using mathematical morphology by performing a dilation operation over the feature space [20, p184]. Although seemingly faster for simple cases this is less appropriate for feature spaces of more than two dimensions and infeasible for sparse datasets. Where cutting the cluster tree allows the proximity of cluster means to be controlled, the size and shape of the dilation mask allows control over the proximity of individual vectors from neighbouring clusters.

The k-means and CLUSTER algorithm

The k-means algorithm is the most popular of a suite of algorithms which attempt to minimise the squared error for a predetermined number of clusters. As outlined in [61, p21–23] it uses an iterative heuristic splitting and lumping criteria to create clusters until stabilisation occurs.

A more sophisticated algorithm called CLUSTER is proposed by Jain [62, p101–117] which "attempts to find the *best* clusterings containing 1, 2, ..., K clusters". It uses a two phase algorithm, creating a sequence of clusterings containing 2 to K clusters and then combines clusters two at a time to “see if a better clustering can be achieved.” These phases are repeated until the squared error no longer decreases.

As illustrated in Figure 2.10 the square error, which produces a linear cluster boundary using a Euclidean distance, is not very good criterion for the clustering problem.



(a) two elongated clusters (b) disjoint large and small clusters

FIGURE 2.10: The inadequacies of square-error clustering. Based on [62, Fig 3.27].

Gaussian mixture models using the EM algorithm

A mixture of Gaussian models provides an alternative which naturally and quite easily solves the problems in Figure 2.10. These models in combination with an expectation maximisation (EM) algorithm are commonly used [10] [70] [71] and outlined more completely in [10] and [9].

The EM algorithm when used for Gaussian mixture modelling tries to find the parameters of the mixture of distributions to maximise the likelihood estimation for the data. Conceptually in 2 dimensions one can consider it to model the data with Gaussian hills which are iteratively refined to fit the data.

In general the multivariate Gaussian with $\mathbf{K}$ components in $\mathbf{N}$ dimensions is defined [10] as:

$$g(x|\theta) = \sum_{i=0}^{\mathbf{K}} \mathbf{w}_i f(x|\mu_i, \mathbf{C}_i)$$

$$\text{where} \qquad f(x|\mu_i, \mathbf{C}_i) = \frac{1}{(2\pi)^{1/\mathbf{N}}|C_i|} e^{i\frac{1}{2}(x-\mu_i)^T \mathbf{C}^{-1}(x-\mu_i)}$$

Here $\mathbf{w}_i$ is the relative weight of Gaussian i which has associated mean μ_i and covariance matrix $\mathbf{C}_i$ and θ is a vector representing the current model parameters.

The problem is to find the model parameters θ which maximise the total likelihood or equivalently the log likelihood $\mathbf{L}$ of the n data points.

$$\mathbf{L} = \log\left(\prod_{j=1}^{n} g(x_j|\theta)\right) = \sum_{j=1}^{n} \log g(x_j|\theta)$$

The EM algorithm uses iterations of estimation and maximisation steps which estimate the log likelihood of the current model θ according to the data and refine θ to maximise it. Each iteration monotonically increases the likelihood to a (not necessarily global) maximum, although "convergence of the EM algorithm is very slow" [9, p17].

In summary, the algorithm consists of the following steps:

1. For a predetermined number of clusters, establish K Gaussian models with initial parameters μ_i, $\mathbf{C}_i$, $\mathbf{w}_i$ for i=1..K and such that $\sum \mathbf{w}_i = 1$. Initialisation may be randomly based on the data or use a strategy such as *Dog Rabbit* [10]. An example is to set μ_i to a data point, $\mathbf{C}_i = I$ (the identity matrix) and $\mathbf{w}_i = \frac{1}{\mathbf{K}}$.

2. For all Gaussians calculate the Eigenbasis for $\mathbf{C}_i$ and hence the values of $\mathbf{C}_i^{-1}$ and $|\mathbf{C}_i|$.

3. For every data point calculate its weighted *distance* or likelihood from each mixture and the sum of the distances. Use this to assign a relative proportion of the data point to each recalculated mixture, storing running weighted sums for the number of data points, vector elements and element products, i.e. for data point j, current mixture i and measurements for new mixture i',

$$\text{weight of j to i} = \mathbf{w}_{(j \text{ to } i)} = \frac{\mathbf{w}_i f(x_j|\mu_i, \mathbf{C}_i)}{\sum_{l=1}^{\mathbf{K}} \mathbf{w}_l f(x_j|\mu_l, \mathbf{C}_l)}$$

$$\text{number of points in i}' = \text{ndp}'_i = \sum_{j=1}^{n} \mathbf{w}_{(j \text{ to } i)} i$$

$$\text{sums of points for i}' = \text{sumx}'_i[l] = \sum_{j=1}^{n} \mathbf{w}_{(j \text{ to } i)} x[l]$$

$$\text{sums of products for i}' = \text{sumxy}'_i[l][m] = \sum_{j=1}^{n} \mathbf{w}_{(j \text{ to } i)} x[l]x[m]$$

4. Recalculate the values for the parameters of each mixture, removing mixtures with less than 1.0 of weighted points. Hence, the new mixture i parameters from the

temporary sums above are:

$$\mathbf{w}_i = \mathrm{ndp}'_i \qquad \mu_i[l] = \frac{\mathrm{sumx}'_i[l]}{\mathbf{w}_i}$$

$$\begin{aligned}\mathbf{C}_i[l][m] = \quad & [(\mathrm{sumxy}'_i[l][m]) - (\mu_i[l])(\mathrm{sumx}'_i[m]) \\ & -(\mu_i[m])(\mathrm{sumx}'_i[l]) + (\mu_i[l])(\mu_i[m])(\mathrm{ndp}'_i)]/(\mathbf{w}_i - 1.0)\end{aligned}$$

5. Repeat from step 2 until the algorithm converges to a minimum or is sufficiently stable.

The algorithm results in a number of weighted Gaussians which model clusters in the data.

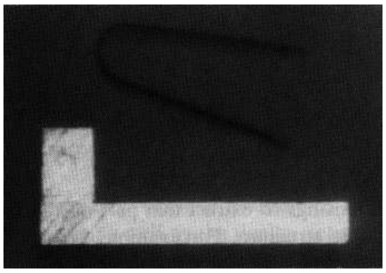
2.2 Locating defects

Is the industrial world ready for machine vision? As discussed by a panel of university professors in [72], although the unanimous answer is yes, there is "still a gap between the available technology and successful applications."

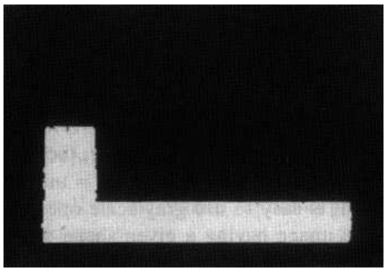
There are numerous applications of optical industrial inspection in the "metallurgical industry, metal fabricating and automotive industry, pulp and paper, chemical process industries, plastic forming and in electronic components" [73, p17–21]. For example, in 1980 a small British petrochemical company employed almost 100 on-line process analysers to perform highly repetitive and potentially automated inspection. Other applications outlined in this section include the detection and classification of leather defects, food screening and the analysis of wood in the timber industry.

Although automated systems currently exist, most are control systems with specific real-time hardware components. They often make use of computers, but rarely computer vision. Mechanical systems, such as the measurement of surface roughness for metallic surfaces, are successful but often harm the final product. Hence an emphasis is placed on non-destructive inspection, such as those based on light scattering theories or other optical technologies and techniques [73, p79–531]. However, the use of digital image processing techniques for surface inspection has (as recently as 1988) been concluded to be infeasible even for restricted applications with simple and highly repetitive objects [73, p288].

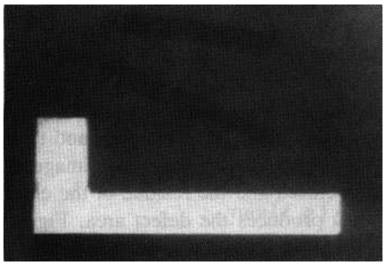
This conclusion is perhaps based on a combination of lack of knowledge and either the absence or simplicity of approaches outlined in the more common textbooks. These [20] [35] [74] [73] concentrate on morphological operations, simple template matching, thresholding and histogram techniques, such as that depicted in Figure 2.11. This example of defect detection is highly specific and sensitive to thresholds and mask characteristics. A similar specifically designed example which used histograms is presented



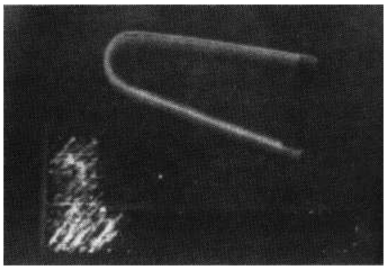
(a) original image f_0



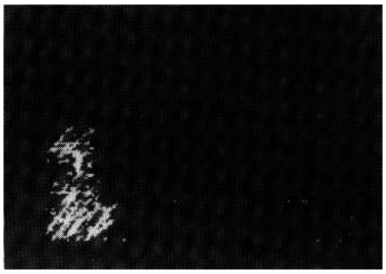
(b) threshold



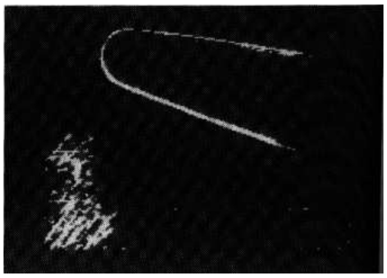
(c) original masked with threshold and then closed with a rod structuring element R



(d) closing residue



(e) threshold of (d)



(f) ANDing of two thresholded images

The algorithm can be compactly written by
$[f_0 \wedge (f_0 > t_1)] \bullet R - f_0 \wedge (f_0 > t_2) > t_2 \wedge (f_0 \wedge t_1)$.

FIGURE 2.11: Detection of a polished surface. Taken directly from [20, p214].

in [74, p352–354].

Despite these inadequacies, visual inspection can be categorised into the classes *geometric*, *structural* and *superficial* [35, p411-434]. Geometric tasks, for example the screening of food products such as biscuits, are well suited to template matching although shape detection is often difficult due to perspective even for the human eye (see Sections 2.1.2 and 2.1.3). Structural problems look for "missing items or foreign bodies" considering topological features such as holes in mechanical components. Both geometric and structural inspection concentrate on locating and dealing with objects. Superficial types examine surface quality.

Surface quality and surface defects are highly subjective and require the appropriate training. Examples include steel, wood, glass or leather. For steel and wood, high production line speeds are desirable, however it is virtually impossible for a human to focus on objects at rates above 20m/s [35, p429]. On the other hand automated systems which can cope with such speeds are unable to inspect wood, which is extremely subjective and "something of an art" [35]. Both applications are made even more difficult by the different interpretation of *texture* from a line inspector and vision researcher viewpoint.

One approach to detecting leather defects [26] indicates that constant illumination is a problem, suggesting the use of an oblique incident light to highlight detects and the need for constant uniform illumination across the image. Consequently, a specifically design machine vision system is used. Despite this tightly controlled environment, it is concluded that, "after careful examination, we found that it was not possible to use any method to detect all types of defects." The final methods adopted used a combination of edge detectors and histogram statistics for texture to locate defects. Defects are classified based solely on their size and severity, with morphological operators used to create a final defect map. In this work more emphasis is placed on cutting the leather given a predetermined defect map.

A second approach to leather [75] treats it as a multi-scale edge detection problem using pyramids and examining a variety of edge features. To avoid image acquisition problems taking images directly from photographs is proposed. The final system uses the mean values of the two highest edges in a neighbourhood at each pyramid level and concludes by outlining that "there are some difficulties for parameter tuning" with respect to root selection and edge thresholding for noise reduction. In summary, the system assumes defects consist of edges and requires several static parameters.

Aspects relating to the detection of glass defects are discussed, but not solved in [74, p354–360]. In a similar vein to Figure 2.11 an example solution is illustrated for locating a crack in patterned glass. The technique takes the threshold of the difference between the original image and a smoothed image. This specifically designed technique relies on the relative thickness of the pattern and crack and a careful choice of threshold. It perhaps reinforces a naive and pessimistic statement: "Inspection always requires comparison to an ideal model of some kind."

A fully designed and implemented glass defect detection scheme presented in [76] locates, but does not classify, major defect classes including *blister*, *knot* and *stone*. By making use of an industrial standard which defines defects based purely on size and accepts those with width less than 0.005 inch, the processing is able to be defined by simple low level operations. The final implementation uses a 1×5 median filter, a gradient operator for edge detection and a threshold component all implemented in hardware. Concentration is placed on the integration of optical, mechanical and electronic components as opposed to image processing or recognition techniques. However, it is suggested that morphological operations could be used improve performance, although components would be too costly.

There are countless other examples of systems designed specifically based on the properties of an individual problem. Several other systems deal with printed circuit boards

(PCBs) and low level electronic devices [77] [78] as discussed further in [35, p427–429] [74, p360–364]. Attempts have also been made to combine the recognition and inspection process using template matching [79].

A more general, but less common, technique involves the location of defects by distinguishing foreground objects or anomalies from background homogeneity. In [59] Gabor filters are used on "*Dinophysis* and *Ceratia* images" with the final correct identification of 77.5% of background pixels (a misclassification of 22.5%) and 98.7% of foreground pixels. The problem is basically reduced to a clustering problem with 2 clusters (see Section 2.1.5). With the same problem [80], the two illustrated examples claim classification rates of 80% and 78.44% although results are subjective. The final results claim to produce 73.4% "acceptable images" as opposed to the author's previous results of 54%. In both papers, although the approach aims to locate anomalies on an otherwise homogeneous image, the techniques and results are less than impressive.

In summary, there are many systems which lack generality. Current implementations apply assumptions or knowledge specific to one particular application and hence are not reusable. With these systems there is also a re-occurring emphasis on the choice and sensitivity of lighting conditions [74, p352] [26]. This is exacerbated by the need for predefined thresholds or masks in current standard techniques. Davies makes the general conclusion that "pattern recognition techniques are highly relevant to the process of inspection." [35, p432]

2.3 Frequency methods

This section provides a basic overview of the Fourier transform and its properties and hence a background for the introduction of Gabor filters. Throughout, common thoughts, definitions and theories are outlined or illustrated from an image processing viewpoint. This provides the relevant information for several methods in texture analysis and segmentation, as presented in Sections 2.4 and 2.5.

For an elementary and very friendly mathematical introduction to Fourier analysis refer to [81, p113–135]. A more typical and thorough coverage appears in [82, p117–197] and a summary of transform pairs and properties with emphasis on two dimensional images in [37, p15–21].

2.3.1 The Fourier transform and its useful properties

Although not usually presented as such, the Fourier transform is a change of basis obtained by projecting a signal onto the set of orthonormal sinusoid functions $e^{-\mathbf{j}x} = \cos(x) - \mathbf{j}\sin(x)$.[6] This corresponds to the more common definition, outlined for the N dimensional case below [82].

[6] Since $\cos x = \sum_{n=0}^{\infty} \frac{(-x)^{2n}}{(2n)!} = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \ldots$, $\sin x = \sum_{n=0}^{\infty} \frac{(-x)^{2n-1}}{(2n-1)!} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \ldots$, and $e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$.

For functions defined as mappings from Euclidean space into the complex plane, i.e. $f : \mathbb{R}^N \to \mathbb{C}$ and i.e. $F : \mathbb{R}^N \to \mathbb{C}$, the Fourier transform $\mathbf{F}$ is defined as

$$\mathbf{F}(f(x)) = F(\omega) = \int_{\mathbb{R}^N} f(x) e^{-\mathbf{j}\omega^T x} d^n x$$

Conversely, the inverse Fourier transform is defined as

$$\mathbf{F}^{-1}(F(\omega)) = f(x) = \frac{1}{(2\pi)^N} \int_{\mathbb{R}^N} F(\omega) e^{\mathbf{j}\omega^T x} d^n \omega$$

One of the common functions or characteristics of the Fourier transform (which is typically assumed rather than defined) is the power spectrum or power spectral density of P is given by the square of its magnitude [41, p82].

$$P(\omega) = |F(\omega)|^2 = \Re^2(\omega) + \Im^2(\omega)$$

Some of the properties and theorems of the Fourier transform are:

- The transform is unique and without loss of information. "There is no loss of information if instead of preserving the image, its Fourier transform is preserved." [37, p16]

- The transform is linear, hence for constants $\alpha, \beta \in \mathbb{C}$, $\mathbf{F}(\alpha f + \beta g) = \alpha \mathbf{F}(f) + \beta \mathbf{F}(g)$. [82] Consequently, the sum of the Fourier transforms of three images corresponds to the spectrum of the sum of the original images and hence the inverse transform will produce this sum as illustrated in Figure 2.12. Furthermore, a signal or image can be decomposed into parts based the frequency spectrum produced from its Fourier transform. For example, in Figure 2.13 the flower picture has been decomposed into its low and high frequency components using a simple low and high pass filter pair. In this case the sum of the two resulting images is equivalent to the original.

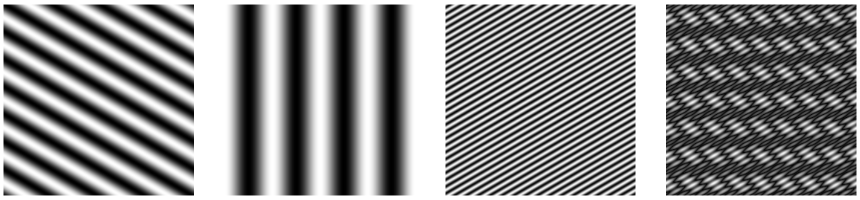
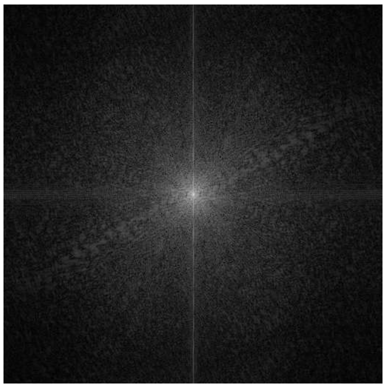


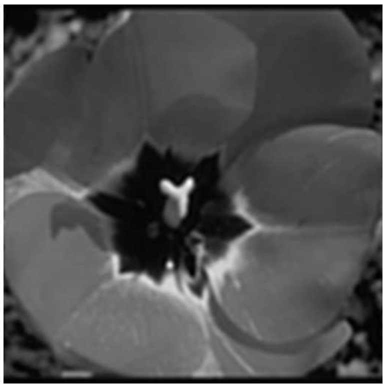
FIGURE 2.12: The linearity property of the Fourier transform. It shows three sine waves (s_1, s_2, s_3) along with the inverse of their summed frequency spectra $(\mathbf{F}^{-1}(\mathbf{F}(s_1) + \mathbf{F}(s_2) + \mathbf{F}(s_3)))$.

- The transform is separable as is clear from the definition. Hence a two dimensional transform can be implemented using successive one dimensional transforms. This is

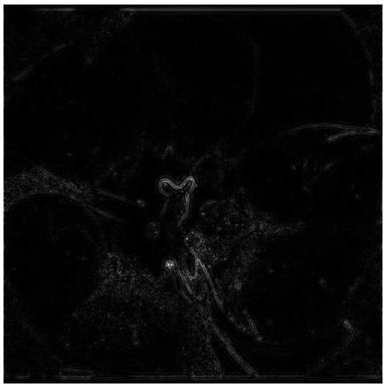
(a) flower, average=60.38



(b) FFT of flower; $(1 + \log(FFT)$ scaled.



(c) low-pass filter response -5.27 to 205.41, average=60.38



(d) high-pass filter response -66.94 to 103.43, average=$-4.48 \times 10^{-17} \approx 0$

FIGURE 2.13: The fast Fourier transform of a digital image and the result of applying a low and high pass filter. The magnitude of the transform is shown and all other images are purely real.

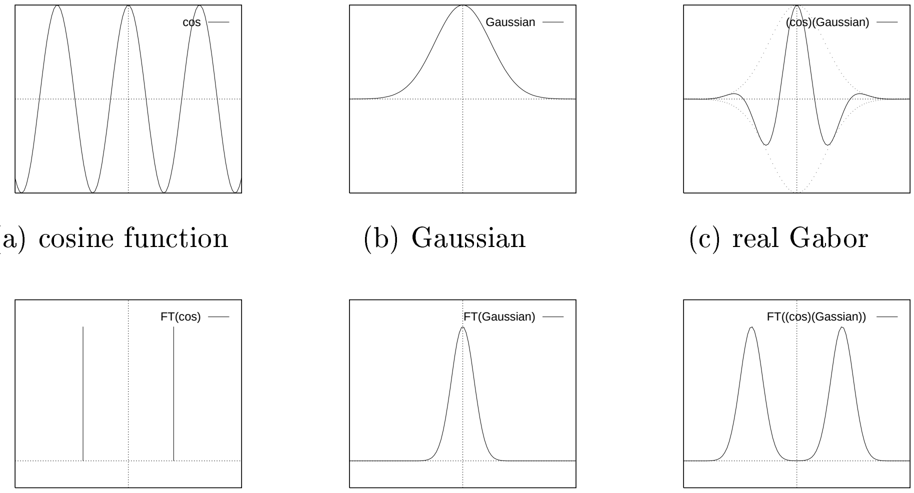
important when considering algorithm complexity and implementation overheads.

- The convolution theorem provides a relationship between convolution in time and multiplication in frequency and vice versa. It states that the "Fourier transform of the convolution of two functions and the product of their Fourier transforms" [37, p18]. The converse is also true, hence for an arbitrary dimension and the convolution operator $*$

$$h(x) = f(x) * g(x) \Leftrightarrow H(\omega) = F(\omega)G(\omega)$$

$$h(x) = f(x)g(x) \Leftrightarrow H(\omega) = F(\omega) * G(\omega)$$

This is an extremely powerful theory allowing convolution with an arbitrary size mask to be achieved using one or two Fourier transforms and an inverse transform.



(a) cosine function (b) Gaussian (c) real Gabor

(d) two impulses (e) Gaussian (f) shifted Gaussians

FIGURE 2.14: Functions and their Fourier transforms illustrating the convolution theorem.

It has also led to encouraging the design of filters with respect to spatial frequency[7] rather than masks in terms of image pixels. Figure 2.14 provides an example resulting from this theory. Refer to [41, p100–111] for further discussion and the similar relationship for correlation.

- A change of basis in the image space corresponds to the same change of basis in frequency. This is particularly relevant to rotation in the two dimensional Fourier transform where the rotation of the transform corresponds to the same rotation of the image. For further discussion on affine coordinate changes refer to [82, p136-137]. Although this holds for continuous signals it is not globally appropriate for the discrete transform due to the additional periodicity constraint. This is illustrated in Figure 2.15.

- Properties exist for signals or images with special characteristics as shown in Table 2.1. These are very useful for filter design.

function	Fourier transform
real and even	real and even
real and odd	imaginary and odd
imaginary and even	imaginary and even
imaginary and odd	real and odd
imaginary	hermitian, i.e. $f(-x) = f^*(x)$

TABLE 2.1: Fourier transform characteristics for functions with special characteristics.

[7]This is often colloquially referred to as "the Frequency domain". This term will generally be avoided.

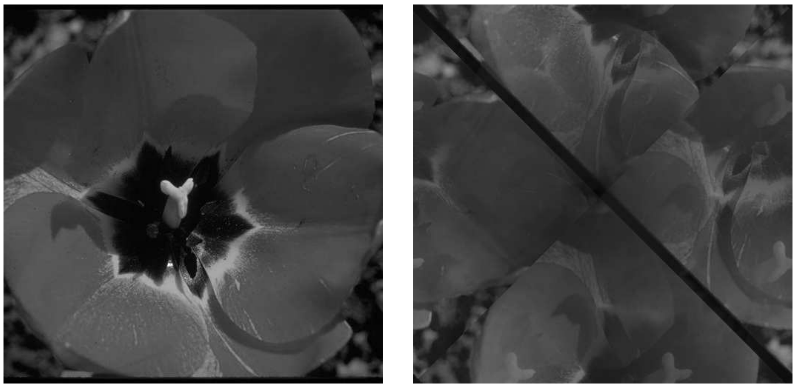


FIGURE 2.15: The result of rotating the FFT by 45 degrees and taking the inverse. The final image lacks the desirable phase or positional information.

- The autocorrelation function can easily be obtained using the Fourier transform [83, p249–250] [84, p13]. The autocorrelation function of an image $\mathbf{I}(x, y)$ is defined below along with its equivalent according to the autocorrelation theorem.

$$\rho(x,y) = \frac{\sum_i \sum_j \mathbf{I}(i,j)\mathbf{I}(i+x,j+y)}{\sum_i \sum_j \mathbf{I}^2(i,j)} = \mathbf{F}^{-1}(|\mathbf{F}(\mathbf{I}(x,y))|^2)$$

- Spatial derivatives and absolute geometric moments can be determined using the Fourier transform. However, the methods for calculating these are not particularly useful in digital image processing. The relationships for the N^{th} derivative, and one dimensional case for moments[8] are:

$$g(x) = (\frac{d}{dx})^N f(x) \Leftrightarrow G(\omega) = (\mathbf{j}\omega)^N F(\omega)$$

$$\mathbf{M}_p = \int_{-\infty}^{\infty} x^p f(x) dx = \mathbf{j}^p [\frac{d^p}{d\omega^p} F(\omega)]_{\omega=0} \quad \forall\ p \in \mathbb{Z}^+$$

  The only point to note here is than $\mathbf{M}_0 = F(0)$. Hence the signal average is stored at zero frequency. This is not surprising since at this point the transform breaks down to a projection onto the basis function $\cos 0 + \mathbf{j} \sin 0 = 1$. This zeroth frequency component is usually called the DC component.

Although there are many more properties and considerations, the above summarises the more useful ones frequently used and misused by image researchers.

[8] These are extended to an arbitrary dimension in [82, p139–140].

The discrete Fourier transform (DFT)

For digital images the continuous Fourier transform cannot be applied because the image or signal is discrete and furthermore bounded. To handle this boundedness constraint the image is considered to be periodic, hence the left side of the image wraps around to the right side. This is necessary to preserve a 1 to 1 correspondence between spatial discrete functions and its periodic Fourier transform as proved in [82, p182–184]. Consequently the frequency spectrum for an image is defined for $\omega \in [-\pi, \pi)$[9]. For two dimensions,

$$F(\omega_x, \omega_y) = \sum_{m=0}^{N_x-1} \sum_{n=0}^{N_y-1} f(m,n) e^{-\mathbf{j}(m\omega_x + n\omega_y)} \qquad \omega_x, \omega_y \in [-\pi, \pi)$$

An example of the DFT performed on the signal (or image) $(0,0,1,0)$ is shown in Figure 2.16. The frequency representation of the signal correspond to sine and cosine waves of frequencies ($-\pi$,$-\pi/2$, 0, $\pi/2$) which produce a nice smooth function when drawn in continuous form. This continuous function passes through the discrete points of the original signal and hence can be used to produce the original data points, i.e. as with the continuous Fourier transform no information is lost. This also provides justification for sub-pixel measurements in images, for example locating an edge to a precision of 0.2 of a pixel [35], although such claims are often dubious resulting from fitting sinusoids rather than actual image values.

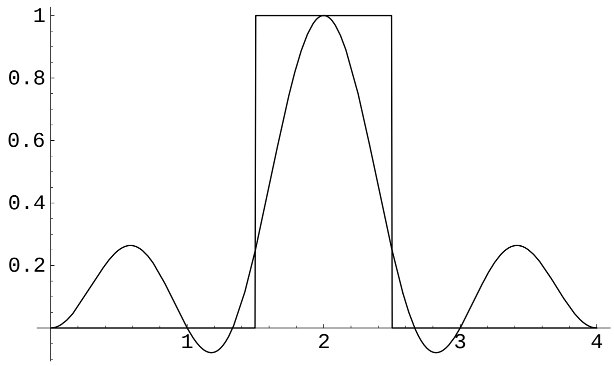


FIGURE 2.16: A discrete signal and the function corresponding to the Fourier co-efficients of the DFT.

For a one dimensional signal the DFT requires a number of multiplications and additions proportional to N^2, or $O(N^2)$. However with the proper decomposition it can be performed in $O(N \log_2 N)$. Furthermore, since the transform is separable the two dimensional transform requires $O(N^2 \log_2 N)$ operations. This fast Fourier transform (FFT) offers a more than 3000 times improvement in speed for a 512×512 images. The majority

[9] All illustrations, such as Figure 2.13, 2.17, 2.18 and 2.23, range from $-\pi$ to π on both axis with the origin is the centre.

of algorithms do however require an image to have dimensions of $2^N \times 2^M$ for any positive integers N and M. An FFT algorithm is presented and discussed in [41, p119–128].

2.3.2 Gabor filters

A Gabor function is basically a Gaussian modulated by a sinusoid as shown in Figure 2.14. It was, in effect, an early non-orthogonal wavelet. It can be specified by the frequency of the sinusoid ω and the standard deviation of the Gaussian σ.

$$\mathbf{h}(x) = g_\sigma(x)e^{\mathbf{j}2\pi\omega x} = \frac{1}{\sqrt{2\pi\sigma}}\, e^{\left(\frac{x^2}{2\sigma^2}+\mathbf{j}2\pi\omega x\right)}$$

Despite this simple form, there is no standard and precise definition of a two dimensional Gabor function, with several variations appearing in the literature [85] [86] [87] [88]. The history of 2D Gabor functions dates back to a 1980 paper by Daugman [89] which examines a spectral analysis of cortical receptive field profiles. Although this paper is often associated with the development of the 2D Gabor function, it makes no mention of Gabor functions nor does it provide any mathematical definition equivalent to, or in the same form as, the original one dimensional Gabor function. The paper does however propose a spectral model for the receptive field profile which consists of two real Gaussians located symmetrically around the origin. The general rationale of the model is based on the appearance of other receptive field models. This simple model is depicted using a figure which is subsequently included and referred to by Daugman in 1985 [85] with the caption "An even-symmetric member of the family of 2D Gabor filters, ... (From Daugman 1980)" and also reprinted with citation in [90] and [91]. Although these references provide a misconception that the 2D Gabor functions were first proposed by Daugman in 1980,[10] in 1985 [85] he is careful to reference Marcelja (1980) [94] who proposed that one dimensional Gabor functions be used to model receptive field profiles. Regardless of these developments, [85] is perhaps the most influential paper on 2D Gabor filters. It defines the form of the spatial filters and frequency response as well as proving many important properties. This paper has been referenced extensively since.

Definition

The following, more general, definition of the 2D Gabor function is based on the one outlined in [95]. This definition does not place any constraints on the orientation or aspect ratio of the Gaussian with respect to the sinusoid, but does assume that the sinusoid has one particular predefined orientation. The 2D Gabor function $\mathbf{h}(x, y)$ and its Fourier

[10] Lee [92] following a citation to [89] states, "The 2D Gabor functions proposed by Daugman ..." and Ravichandran [93] directly states that 2D Gabor functions were first proposed by Daugman [89].

transform $\mathbf{H}(x,y)$ are defined by:

$$
\begin{aligned}
\mathbf{h}(x,y) \quad &= g(x',y')e^{\mathbf{j}2\pi(Ux+Vy)} \\
&\text{where } (x',y') = \quad (x\cos\phi + y\sin\phi, -x\sin\phi + y\cos\phi) \\
&\qquad g(x,y) = \quad \frac{1}{2\pi\lambda\sigma^2}e^{-\frac{(x/\lambda)^2+y^2}{2\sigma^2}} \\
\mathbf{H}(x,y) \quad &= e^{-2\pi^2\sigma^2[(u'-U)^2\lambda^2+(v'-V)^2]} \\
&\text{where } (u',v') = \quad (u\cos\phi + v\sin\phi, -u\sin\phi + v\cos\phi)
\end{aligned}
$$

Using this definition, $\mathbf{h}(x,y)$ is a complex sinusoid of radial frequency $F = \sqrt{U^2+V^2}$ and orientation $\theta = \tan^{-1}(V/U)$ modulated by a 2D Gaussian with aspect ratio λ and orientation ϕ. The Fourier transform $\mathbf{H}(x,y)$ consists of a single real Gaussian function translated from the origin (similar to Figure 2.14). The orientation of the Gaussian remains unchanged, but the aspect ratio becomes $1/\lambda$.

Typically a circular Gaussian is used ($\lambda = 1$) and the orientation chosen (or designed) to match that of the sinusoid ($\theta = \phi$). Consequently the specification of the Gabor function is reduced to (U, V, σ) or radial frequency (F), orientation (θ) and bandwidth (σ). Other restrictions are presented in the next section.

Properties and rationale for use

A common rationale for using 2D Gabor filters stems from its origins of modelling cortical receptive fields in the visual cortex of cats and monkeys [89]. However, even though Gabor filters provide a reasonable model, there is a significant residual, or difference, between the two as indicated in [96] [97] [85] and [91]. This is not sufficient justification since other filters such as difference of Gaussians (DOG) and difference of offset Gaussians (DOOG) used by Malik and Perona [98] [99] have similar properties.

The second property dwelt upon is the achievement of optimal resolution in time and space [100]. As proved in [85] the joint resolution is limited by $\frac{1}{16\pi^2}$. This theory ensures an inevitable tradeoff between time resolution and frequency resolution "analogous to Hiesenberg's uncertainty principle in physics" [92].

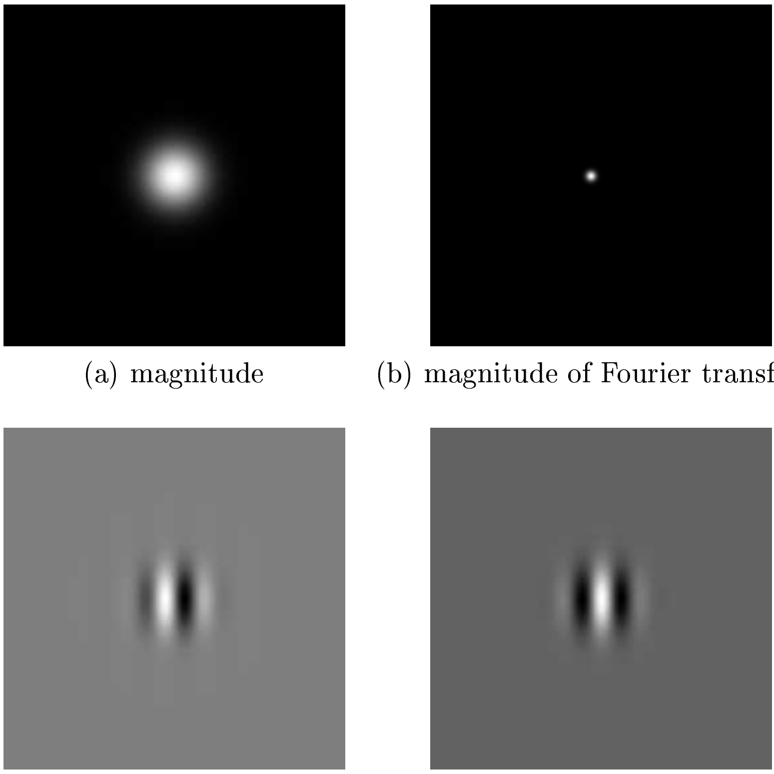
Almost all work which uses 2D Gabor filters justifies their use with these two properties, for example the abstract of [101] includes "The rationale presented for the use of the Gabor filters is their relation to models for the early vision of mammals as well as their joint optimum resolution in time and frequency."[11]

A well used property as indicated in the definition above is the ability to tune filters to specific centre frequencies, orientations and radial frequency bandwidths. Consequently the "filter masks can be considered as orientation and scale tunable edge and line (bar) detectors." [107] Unfortunately this claim is only partially true. The resolution and general ability is limited by the joint resolution constraint, otherwise these filters would have single

[11] A handful of papers with similar justifications include [92], [102], [103], [104], [87], [88], [105] and [106].

handedly solved many computer vision problems such as edge detection.

Finally, Gabor filters can quite easily be implemented using the convolution theorem and the FFT. Figure 2.17 shows the magnitude, real symmetric and imaginary asymmetric components as well as the FFT of an arbitrary Gabor filter at an orientation of 0 degrees with frequency $\omega = \frac{\pi}{16}$ Figure 2.18 shows Gabor filters modulated by the same Gaussian at frequencies of $\frac{\pi}{8}$, $\frac{\pi}{4}$, $\frac{\pi}{2}$ and π. Note the interference caused at $\omega = \pi$ is due to the violation of the Nyquist sampling theorem. The final two filters show the effects of variation in spread and orientation (Figure 2.19).

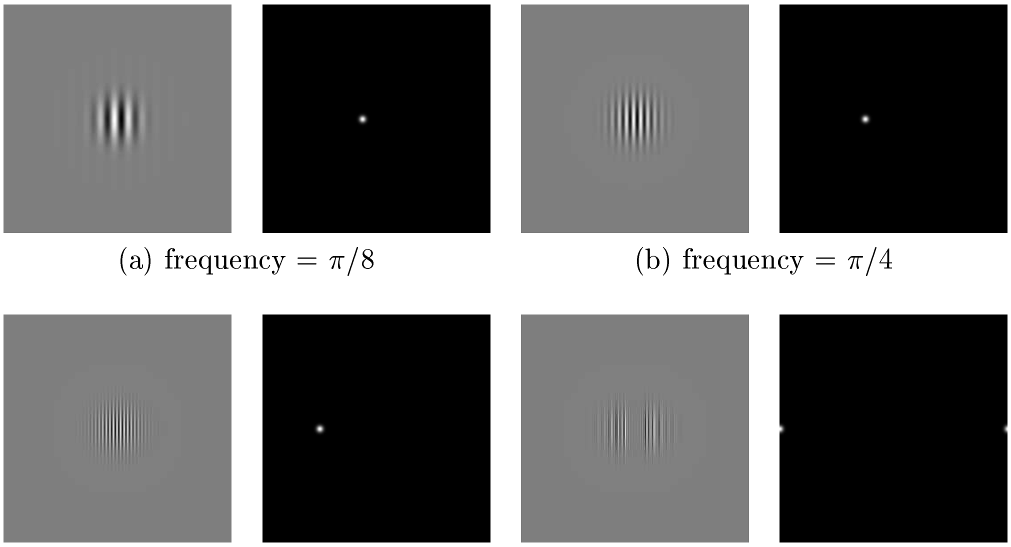


(a) magnitude (b) magnitude of Fourier transform

(c) real component (d) imaginary component

FIGURE 2.17: An example of a 2D Gabor function.

Each of these filters was created in spatial coordinates with other images produced using an FFT. This took about 0.35 ± 0.01 seconds for each 256×256 pixel example using an R10000 processor. By specifying the frequency spectrum the calculation of a Gabor filter response requires an image FFT, a complex image multiplication and an inverse FFT, taking 0.81 ± 0.01 seconds. Testing these process on a *Linux* workstation with a 200MHz Pentium processor took 0.86 ± 0.01 seconds. Alternatively the Gabor response can be calculated directly using spatial convolution as in [108]. In this extreme case the filter is reduced to a resolution of 16×16 and images reduced by a factor of 4. This improved the speed of calculating a single Gabor filter response from about 3 days to 50



(a) frequency = $\pi/8$ (b) frequency = $\pi/4$

(c) frequency = $\pi/2$ (d) frequency = π

FIGURE 2.18: A series of 2D Gabor functions illustrating the effect of changing the frequency for the same modulating Gaussian. Real component and transform.

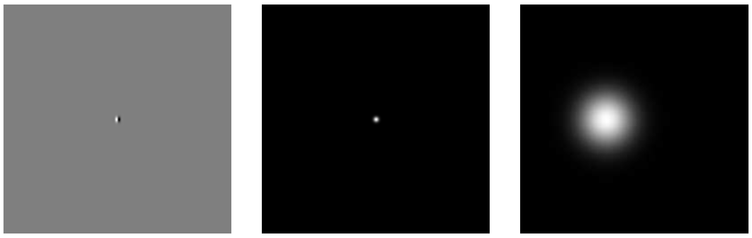
minutes (in 1994). The author suggests a hardware implementation!

2.3.3 Frequency issues, filter design and related characteristics

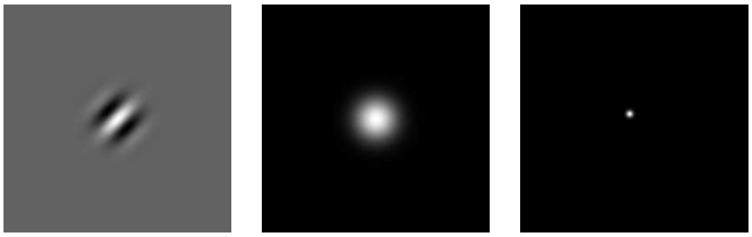
The main frequency issues which are sometimes considered but often overlooked include filter orientation, radial frequency bandwidth, general filter design, linear verses logarithmic frequency scale and phase considerations. These are discussed in this section.

Another issue not commonly addressed involves the automatic determination of image scale or resolution for window size parameter estimation. In [109] it is suggested that the Fourier spectrum be used to automatically select a local window size, although no method or analysis is presented (rather it is left as future work). The approach is appropriate for methods used in Chapters 4 to 6 of this thesis.

Filter orientation is now considered. As clear from its definition, the Gabor filter consists of an odd and even symmetric component and hence the frequency response from a filter at orientations of 0 and 180 degrees is not the same. This issue is typically not raised, however most authors use filters oriented over a range of 180 degrees, e.g. [103] [86] [87] [88] [105] [106] [101], as opposed to the full 360 degrees [54] [92]. Since Gabor filters are generally used to extract meaningful features from real images (and hence with a hermitian frequency response) the response to the even symmetric filter component will remain unchanged for filters oriented 180 degrees out of phase and the odd symmetric (or asymmetric) component negated. Consequently there is some justification for ignoring half of the orientations as is typically done. Alternatively, if phase is considered particularly important the missing orientation can be calculated by modifying the phase of the opposing



(e) modulating with a Gaussian with larger deviation.



(f) orienting the filter at 45 degrees.

FIGURE 2.19: Gabor functions with a different spatial spread and orientation than Figures 2.18 and 2.17 respectively. Real component, magnitude and transform.

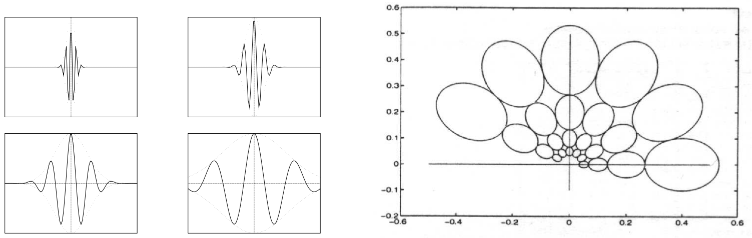
orientation. This method saves about half the storage and computational requirements, although it is not considered in [54].

Orientation spacing varying by 30, 45 or 60 degrees for this 180 degree range is typically used with the choice made to sufficiently cover the frequency spectrum. Jain [105] uses 4 orientations at spacings of 45 degrees, but notes that: "Psychophysical experiments show that the resolution of the orientation tuning ability of the human visual system is as high as 5 degrees." Hence it is suggested that finer quantisation of orientation is necessary with current restrictions made for computational efficiency.

The next issue is radial frequency bandwidth.

> "... it is interesting to note here that the Gabor transform is not really in agreement with psychophysical finding: the Gabor transform has a constant resolution in the space and frequencies [sic] domains (constant shape of its resolution cell in a phase-space description) while the psychophysical experiments have showed that the frequency bands have varied widths of approximately an octave." [106]

The Gabor filters in Figure 2.18 go half way to solving this problem. The frequency responses of these filters are centred one octave apart ($\pi/2, \pi/4, \pi/8, \pi/16$), however the frequency spread is constant due to the constant shaped Gaussian used. Octave band decomposition solves this problem. It is achieved by varying the standard deviation of the modulating Gaussian proportional to the wavelength of the sinusoid as illustrated in Figure 2.20. Such a decomposition provides a resolution inversely proportional to the frequency.



(a) Gabor wavelets (b) Gabor filter design.

FIGURE 2.20: Self similar Gabor *wavelets* and a typical Gabor filter design in the frequency spectrum.

An example of a filter design using octave band decomposition is given in [101]. This feature extraction scheme uses the even symmetric part of the Gabor filter at orientations of 0, 45, 90 and 135 degrees. The filters are designed in the frequency domain by Gaussians with deviations $\sigma_u = \frac{f_0}{3\sqrt{2\ln 2}}$ and $\sigma_v = \frac{\tan 22.5 f_0}{\sqrt{2\ln 2}}$ at frequencies of $\frac{1}{\sqrt{2aN}}, \frac{2}{\sqrt{2aN}}, \frac{4}{\sqrt{2aN}} \cdots \frac{N/4}{\sqrt{2aN}}$ where N is the number of pixels in the image (equivalent to 2π). This design is similar to [110] [105] [111] and many others. The justification for using even filter components at these frequencies is based on psychophysical ground or models of the human visual system.

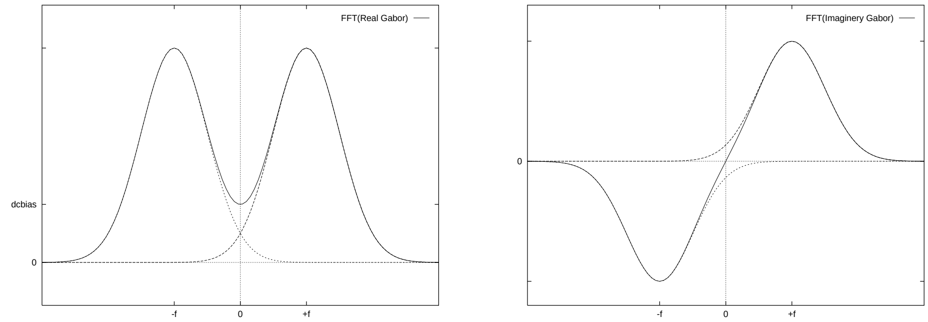
The *octave band decomposition* outlined above is alternatively viewed as a wavelet decomposition using the Gabor function as the mother wavelet.

> "The basic idea behind wavelet analysis is that one uses a bank of filters to analyse the signal. The filters are created from re-scalings of the one wave shape, each scaling designed to pick out particular frequencies of the signal being analysed. An important feature is that the scales of the filters vary geometrically, giving rise to a logarithmic frequency scale." [54, p34]

Despite being referred to as a wavelet, the Gabor functions are not orthogonal and hence cannot benefit from a large proportion of theoretical progress in this area. Consequently it is more difficult to recreate an image from the response to a set of Gabor filters than an orthogonal set such as the discrete cosine transform (DCT), although lossy Gabor based compression systems have been formulated and implemented [92] [104]. Unfortunately there do not exist any orthogonal wavelets consisting of symmetric and asymmetric filters other than the Haar wavelets as proved in [112]. The Haar wavelets are obviously inappropriate.

Although such octave band decomposition does take into consideration a logarithmic response to frequency, filters still suffer from DC biasing at low frequencies. This is illustrated in Figure 2.21.

As outlined in [54, p80] the *log Gabor* function first proposed by Field [113] solves this DC bias problem. Furthermore, as seen in Figure 2.22(a), the frequency response of the filter drops off more slowly at higher frequencies than the original Gabor filter.

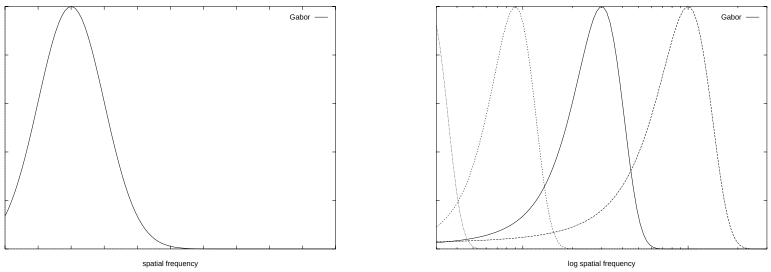


(a) DC bias caused by even component

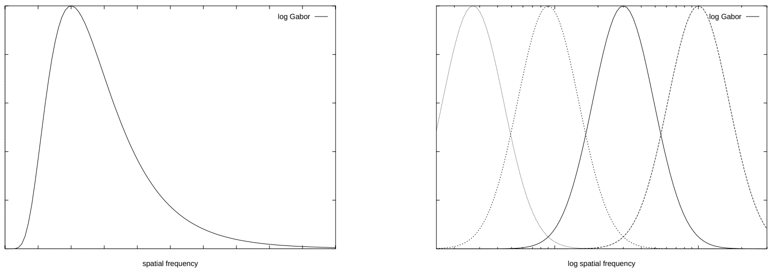
(b) destructive interference of odd component

FIGURE 2.21: DC bias caused by the even symmetric component of the Gabor filters at low frequencies.

This is an important characteristic. As pointed out by Kovesi [54] "Field's studies of the statistics of natural images indicate that natural images have amplitude spectra that fall off at approximately $1/\omega$." Field not only suggests that this representation is more compact, but suggests that it may "help to explain why the frequency-response functions of many cortical cells are symmetric on a log axis." [113] Perhaps more interesting is that the form of this *log Gabor* function matches very well with alternative models discussed by Daugman in 1980 [89, Fig 5]. Considerations with this particular model have not been clearly made.



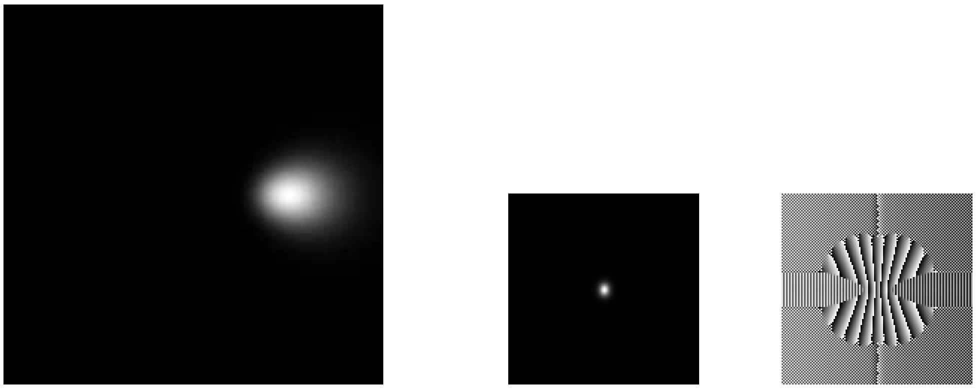
(a) Gabor function at linear and logarithmic frequency scales.



(b) *log Gabor* function at linear and logarithmic frequency scales.

FIGURE 2.22: Characteristics of the Gabor and *log Gabor* frequency spectra.

Intuitively, and from an image recognition viewpoint, it would appear that the general spread of the function into higher frequencies is disadvantageous, causing *less independent* measurements within the frequency spectrum. Figure 2.23 shows a *log Gabor* function as used by Kovesi [54] with its corresponding inverse FFT. For this illustration zero is used to cover the undefined value at the frequency origin.



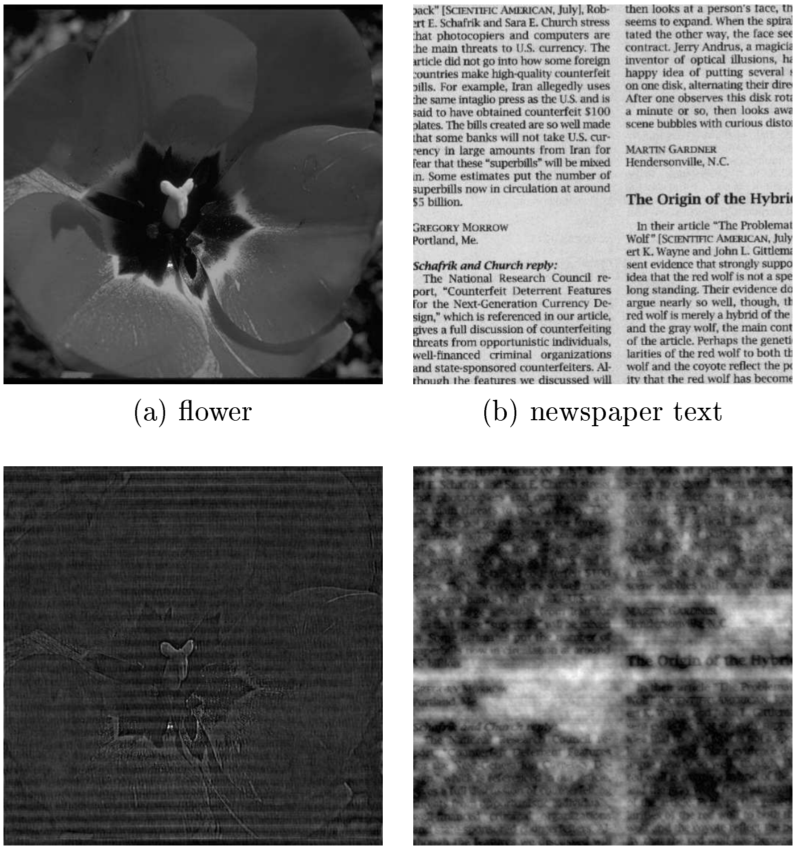
(a) *log Gabor* function (b) magnitude and (c) phase of spatial filter

FIGURE 2.23: An example of the *log Gabor* filter used by Kovesi [54]. Generated for a spatial wavelength of 4 pixels and appropriate angular and frequency deviations.

Circular-Mellin features provide an alternative spectral decomposition in the log-polar coordinate system. These orthogonal operators can be used for scale and rotational invariant feature extraction and it is suggested that while "these feature extractors have similar functional form to the Gabor operators, the distortion-invariant characteristics of circular-Mellin operators make them preferable for texture segmentation." [93] Similar biological studies are used to justify the use of the log polar coordinate system. For further detail refer to [114].

The final issue considered in this section is the importance of image phase. In [88] phase is not considered at all important: "The resultant transform is a complex image. However, we use only the magnitude of the transform because any extra information in the phase appears to be not worth the storage costs required to save it." On the other extreme Kovesi bases his entire thesis around the importance of phase, developing numerous methods for measuring phase congruency, an invariant and unit-less low level image feature. He claims that: "When phase information from one image is combined with magnitude information of another it is the phase information that prevails." [54, p20].

The first statement is naive and the second debatable and perhaps even provocative as outlined by the illustrative example in Figure 2.24. Figure 2.24(c) is the result of combining the phase information from (a) with the magnitude information from (b). It is not surprising that a human is able to observe the flower from just the phase information since by definition the phase in the frequency spectrum corresponds to the offset or position of the spatial signal. However, for this example, the texture from (b) is apparent as a global



(a) flower (b) newspaper text

(c) phase from flower (d) magnitude from flower

FIGURE 2.24: The result of swapping the phase and magnitude components of two images.

feature in (c) clearly indicating the importance of the information provided by magnitude. Similar remarks can be made for Figure 2.24(d).

2.3.4 Summary

In summary there are many issues surrounding the Fourier transform and the frequency spectrum it produces. From the point of view of image analysis, feature extraction and recognition it important to know about these and to understand the underlying assumptions. The FFT provides a tool for calculating autocorrelation and performing convolution quickly using arbitrary sized convolution masks. It has also led to the design of local spatial filters in terms of frequency, such as the Gabor filters and its variations. These are justified by their joint optimal resolution in time and frequency and modelling of "cortical simple cell receptive-field profiles" [85].

In general, considering orientations of Gabor filters up to 180 degrees and performing intelligent filter design such as octave band decomposition is appropriate, or at least standard practice, for most recognition applications. Variations, such as the *log Gabor* and circular-Mellin features, further consider psychophysical studies in biological vision systems. Finally, the information contained in the phase of the frequency spectrum should in general not be discarded.

2.4 Texture and texture discrimination

This section provides a brief overview of texture and texture recognition or classification, but not texture segmentation. Texture classification involves describing and distinguishing between textures in a predefined region based on properties modelled or described by feature extraction. One of the fastest growing associated applications is the retrieval of textures, image regions or images from a database based on perceived textural characteristics or content. This section is further restricted to grey scale images from a two dimensional viewpoint. It does not consider three dimensional scenes or the extraction of shape from surface texture.

2.4.1 Introduction, texture theories and properties

There is no precise definition for 'texture'. The Oxford dictionary defines it as "*n.* Arrangement of threads etc. in textile fabric, characteristic feel due to this; arrangement of small constituent parts, perceived structure, (*of* skin, rock, soil, organic tissue, literary work....". As outlined by six different definitions [83], texture is considered as having constant local statistics, being periodically varying, consisting of tonal primitives with spatial organisation, a field with a defined gradient, the repetitive placement of patterns or elements, or a combination of these and other properties. In summary, "the 'definition' of texture is formulated by different people depending upon the particular application and that there is no generally agreed upon definition" [83]. Such varying and even adaptable definitions of texture are apparent throughout the literature.

Qualitative texture properties used such as fineness, coarseness, smoothness, granulation, randomness, irregularity, linearity, periodicity and hummocky are often discussed but, as pointed out by Haralick [20, p455], rarely defined "into precise properties of grey

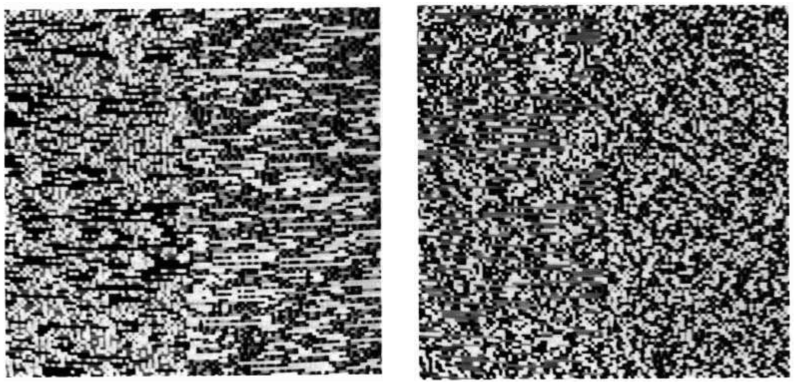


FIGURE 2.25: Images each containing two visually disjoint textures [116, Figures 1 and 26].

level primitives and their spatial distribution." Furthermore, texture analysis often deals with *textures* that have subtle variations in attempts to show the superiority of one method over another. Such illustrations do not usually considering what a human perceives as significantly different texture as shown in Figure 2.25. These two example taken directly from [116] supposedly each consist of "two visually distinct textures".

To cater for the different definitions and approaches to texture, most researchers consider psychophysical findings and texture theories. The classical model of the human visual system suggests texture perception occurs in two distinct modes. The first is known as *preattentive texture discrimination* (or effortless discrimination) occurs within the first 160 milliseconds of viewing an image [117]. The second involves higher level cognitive processing and close scrutiny or focal attention. Julesz, a pioneer and well renowned figure in this field, provides numerous examples of such distinctions [117] [117] [118], concisely illustrated in Figure 2.26. This contains three textures, two of which can be distinguished immediately and one which requires more careful examination.

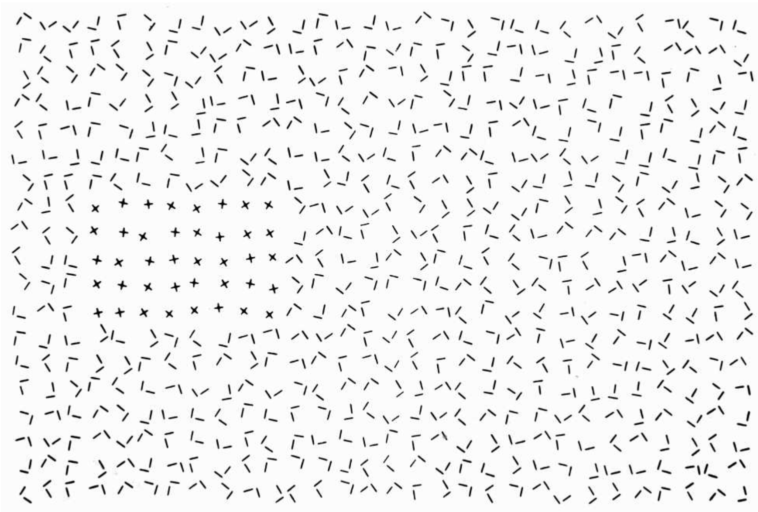
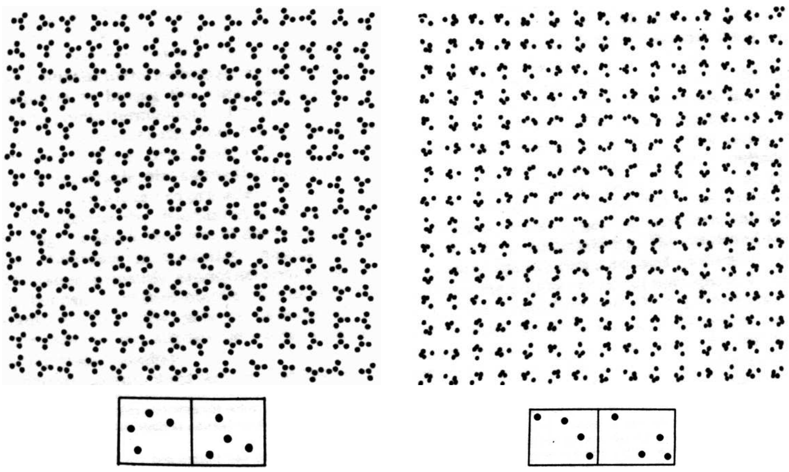


FIGURE 2.26: Two textures immediately identified by preattentive vision, from Julesz [118, p48].

Julesz introduced the theory of textons. Textons are separated or disjoint elements, each consisting of non-overlapping binary line segments with specified length, width, velocity, binocular disparity and flicker rate. [100]

In the development of the texton theories, Julesz's original conjecture that "iso-second order textures are indistinguishable" was later modified to state that "the preattentive textural system cannot globally compute third- or higher-order statistics" [117]. The reason for this modification is apparent from Figures 2.27. This example disproves the original conjecture since the textures in Figures 2.27(b) are preattentively distinguishable based on the number of terminations (three in the top half and four in the bottom). Figure 2.28(b) is based on local conspicuous features of quasi-collinearity of disks. It is



(a) indistinguishable (b) distinguishable

FIGURE 2.27: Preattentive texture discrimination example contradicting Julesz's original conjecture [117, p93].

suggested that this "local feature might be visually extracted by local detectors tuned to elongated blobs (or given width, orientation and aspect ratios)" [117].[12]

As a consequence of these examples and others[13] texton theories also provide considerations for local conspicuous features, connectivity, line crossings and terminations.

Malik and Perona [99] suggests that models for preattentive texture discrimination should have the criteria outlined below. Using these, an alternative model is proposed using DOG and DOOG filters (mentioned in Section 2.3.2).

- Biological plausibility. Julesz work and the Gabor filters (see Section 2.3.2) account for this.

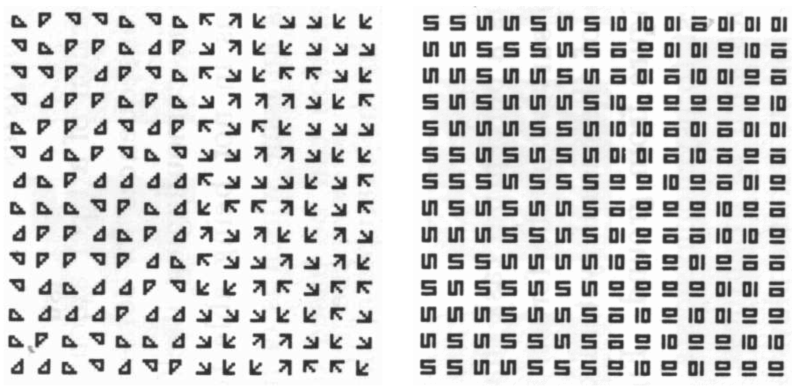
- Generality. The model should be able to be "tested on any arbitrary grey-scale image". Although commonly considered Julesz texton theories are restricted to binary images.

- Quantitative match with psychophysical data. Discrimination should be consistent with human experiments.

Other models consider psychophysically observed frequency characteristics and neurons sensitive to observed blobs, bars and edges. These typically justify the use of Gabor filters. As pointed out by Malik and Perona [98] "no scheme in this framework has been fully specified, implemented and successfully tested."

[12]This claim is possibly the origin for additional justification for *tunable* Gabor filters discussed in Section 2.3.3.

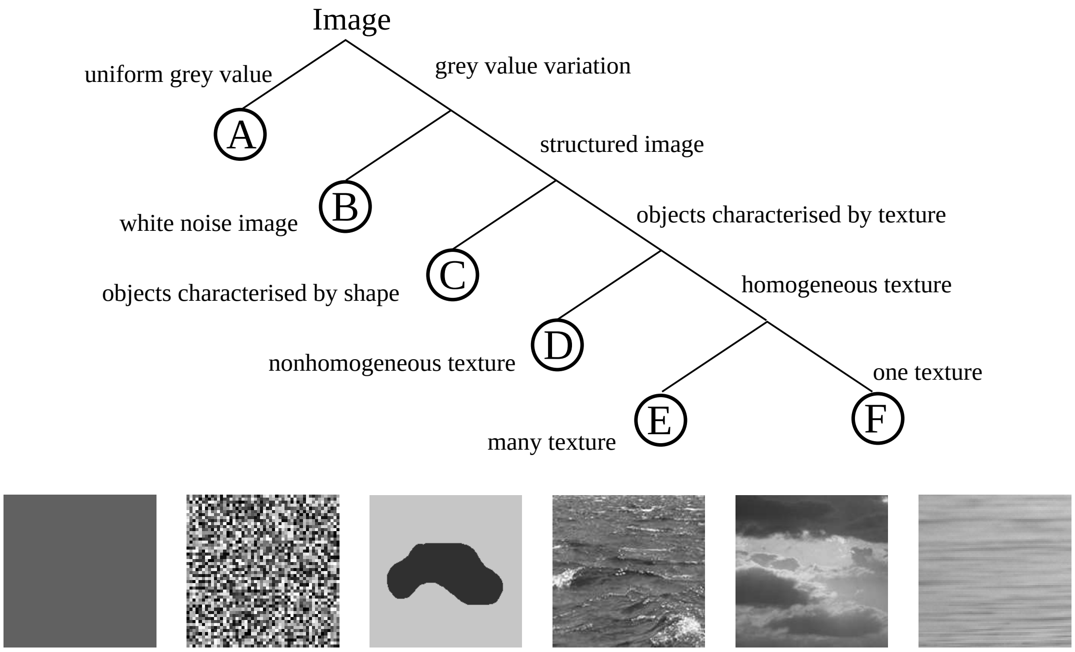
[13]Refer to [117], [100] and [118] for numerous other fascinating examples of artificial textures based on textons. It is also interesting to note that in some examples of texton gradients [100] incidental features other than those being illustrated are observed. Furthermore, supposed preattentive discrimination is not necessarily experienced with response time varying up to 20 or 30 seconds.



(a) triangle-arrow (b) S-IO textures

FIGURE 2.28: Texture pairs used in psychophysical studies of texture perception. Outlined in [117] and scanned from [83, p240].

Figure 2.29 outlines the proposed classification of textured and non-textured images from attempts to determine the presence of texture in images [119]. This system uses the density of local maxima and minima to determine texture coarseness, providing a good illustration of texture resolution considerations. This is also illustrated by observing a newspaper at different resolutions or from different distances. When viewed from a distance the print appears well textured, however from close up the structure of the individual text elements dominates. In general texture is very dependent on the scale at which it is viewed and is typically only observed over a certain resolution range.



(a) (b) (c) (d) (e) (f)

FIGURE 2.29: A taxonomy of textured and non-textured images suggested in [119].

Other texture theories and ideas, such as viewing texture as local line segments with N-folded symmetries [120] [121], are discussed throughout this section.

In summary, there is no definition of texture, and its consideration is based loosely on application and theories of human perception. Although theories of preattentive texture discrimination are widely accepted it should be pointed out that these only consider extremely low level processing and do not cater for focal attention and higher level cognitive processing.

2.4.2 The Brodatz textures

The Brodatz textures, originating from a photographic album for artists and designers [122],[14] are widely considered in texture analysis.

Although often used as a benchmark to compare results, images used differ in resolution, quality, acquisition methods and even subjective individual selection. This is basically due to the lack of standard distribution of Brodatz textures caused mainly by the copyright imposed on the estate of the author. Alternative texture databases, such as MIT Media Laboratories *VisTEX* [123], are freely available but have not yet gained the popularity to become an alternative standard.

Table 2.2 provides reference to numerous papers which use the Brodatz textures. As indicated, the number of textures used for classification, database retrieval or segmentation differs remarkably, with some opting to use a mixture of Brodatz and other selected textures (i.e. 19 *of* 30 *selected*). The classification rates are also not comparable. For example, [129] achieves a rate or traditional accuracy of between 95% and 97% using 30 textures, whereas [130] achieves subjective results by considering the matches that occur in the closest N textures using 111 textures. These results cannot be compared and the latter is very subjective and open to interpretation.

2.4.3 Statistical and simple texture measures

There is an abundance of primitive texture measurements of simple features. For example, the position (x, y), size (x, y), regularity (x, y), brightness, directionality, orientation, roundness, brightness of background, average intensity, standard deviation, contrast and angular second moment. As outlined in [142] these can be effectively used for texture classification by reducing the distance between primitives in the same texture class while increasing it between different classes. Clustering is suggested.

Perhaps the simplest, but most effective, texture measurement is the local standard deviation of pixel intensity, although this is often hidden within complex measurements or presented as a new measurement. For example in [143] a structural uniformity measurement is proposed using "an average gray level $\overline{f}(i, j)$ in a 3×3 window centred around the pixel (i, j) such that: $U = \sum_i \sum_j [f(i, j) - \overline{f}(i, j)]^2$. It measures the consistency of a

[14] This book has not been directly seen.

Citation	A	Features	Approach	Images used	Results
[124]	c	Gabor filters	20 and 50 features	50 selected	70 and 90%
[124]	c	Gabor filters	20 and 50 features	remainder	40–70%
[125]	c	local pixel values	RANK order statistics	10 selected	98.125%
[111]	c	Gabor filters	first order statistics	14 selected	95.7–100%
[111]	c	co-occurrence matrices	40 feature vectors	14 selected	about 60%
[126]	c	co-occurrence features	histogram quantisation	9 selected	87.5%
[127]	c	Gabor filters	Kullback Leibler and quadratic metric	12 selected	98.7 and 95.1%
[128]	c	Gabor filters	Bayesian classifier	10 selected	94.7%
[129]	c	Gabor filters	intelligent decomposition	30 selected	95–97%
[129]	ic	Gabor filters	intelligent decomposition with additional DFT	19 of 30 selected	85–92%
[130]	c	SAR features	principal components	all 111	45–92%[a]
[130]	c	variance features	principal components	all 111	45–60%
[131]	r	Gabor filters	mean and deviation based	109 of 116, some omitted	74.37%
[131]	r	MR-SAR[b] filters	mean and deviation based	109, some omitted	73.18%
[107]	r	Gabor filters	heuristically trained neural network	109 of 116	illustrated
[132]	r	MRF Models	features from clique importance	30 selected	88.8%
[86]	s	Gabor filters	demodulation based	examples	subjective
[119]	s	coarseness	thresholding techniques	3 selected	illustrated
[105]	s	Gabor filters	clustering of local energy	2, 5 and 16 selected	illustrated
[133] [134] [135] [136] [95]	s	Gabor filters	threshold of optimal selection	pairs chosen	illustrated
others	s	Gabor filters	see [106] [137] [101] [138] [109] [139] [140]	selected	illustrated

The application class '**A**' is either classification (c), rotation invariant classification (ic), image retrieval (r) or segmentation (s). Also see [121] [103] [130] [107] [141].

TABLE 2.2: A sample of the different treatments of the Brodatz textures.

[a]Results are based on the number of correct matches in the first N nearest patterns and hence very subjective (a result of 70% is perhaps justified.) Although this is considered texture classification or recognition it is presented from the image retrieval point of view.

[b]Multi resolution simultaneous autoregressive (MR-SAR) features, not to be confused with synthetic aperture radar (SAR) photographs.

pixel with respect to its neighbours and reveals the homogeneity of the regions." This is obviously a scaled variance.

Sanderson [144] uses the autocorrelation of adjacent pixels to describe and reconstruct texture in already segmented areas for image compression. The reconstructed region agrees well with the human observer suggesting that simple natural textures can be reasonably modelled with just three numbers. These parameters ($h_{x,r}(1)$, $h_{y,r}(1)$, σ_r^2) are defined below ($\rho(x,y)$ is defined on page 43).

$$h_{x,r}(1) = \frac{\rho(-1,0)}{\rho(0,0)} \qquad h_{y,r}(1) = \frac{\rho(-1,0)}{\rho(0,0)}$$

$$\sigma_r^2 = (1 - h_{x,r}^2(1))(1 - h_{y,r}^2(1))\rho(0,0)$$

An alternative measurement also based on the autocorrelation is presented in [145]. By considering the fractal dimension of a texture as "a simple simple log ratio of the variance of the surface and the autocorrelation at a distance of 1", a classification rate of 94.6% is achieved for 7 textures. This rate is impressive considering that only a single feature is used. The fractal dimension (H) is defined in one dimensional form as

$$H = \frac{1}{2}[\frac{ln[\frac{R_{I_m I_m}(1)}{\sigma^2} + 1]}{ln(2)} + 1]$$

A measurement of image coarseness offers another simple yet effective texture feature. As outlined in [119]: "Coarseness of an image is not an absolute measure, but depends on the scale at which the image is processed. Reducing the image size makes a textured surface seem smoother, while magnifying the image brings forward the rough structure of the surface." Hence, as with many image recognition tasks, it depends on the scale at which the image is processed. A simple measure of coarseness is defined by the density of extrema (minima and maxima) within a local region. Results in [119] show the effectiveness of locating regions with densities between predefined thresholds (t_L and t_H) on images containing non-textured and Brodatz texture regions. To achieve statistically meaningful or valid results the local windowsize (s) is chosen to produce approximately n local extrema, i.e. $s = \sqrt{(2n)/(t_L + t_H)}$. For example locating textures between $t_L = 0.04$ and $t_H = 0.16$, the window size used is $s = \sqrt{(2)(32)/(0.04 + 0.16)} \approx 18$. Overall the method is very promising as a preprocessing tool for determining the existence of texture in images at various scales.

The final two simple texture techniques considered examine local characteristics. The first [138] uses the pixel values in an $n \times n$ window as the n^2 inputs into a three layer feed forward neural network using backpropagation for training. This thoughtless approach uses 3×3 and 8×8 window sizes, although it is suggested that "a series of different window sizes be used; the one which produces the best output should be selected." Despite the simplicity and computational expense of this method the results are reasonably good, suggesting the effectiveness of using raw pixel values without spatial ordering.

The second local technique [125] considers ordering pixel values in the horizontal, vertical and diagonal directions. In each case, the two pixels either side of the central pixel are examined. The order of these four pixels determines one of $4! = 24$ possible *rank* values. Statistics of these rank values are used with k-nearest neighbours to classify 10 selected Brodatz textures with an accuracy of 98.1%. Interestingly and counter intuitively, the orientation with the least variance is weighted twice as heavily for each image location (no reason or justification is presented). This novel approach illustrates the effectiveness of statistics based on adjacent or spatially defined pixel values. A more standard approach is to use features from co-occurrence matrices.

2.4.4 Co-occurrence matrices

Co-occurrence matrices were first proposed by Julesz in 1962 for texture discrimination experiments [20] [146]. Since then they have become one of the standard tools for texture classification.

The grey scale co-occurrence is defined by $P_{(d_x,d_y)}$, a square matrix of relative frequencies for *neighbouring* pixels with relative offsets (d_x, d_y). In other words, the matrix provides a histogram for the occurrence of grey levels at a specified displacement. These matrices are symmetric as illustrated in Figure 2.30. The matrix $P_{(1,0)}$ (equivalent to P_H at 0 degrees in [20]) is the co-occurrence matrix for pixel values at a horizontal distance of 1. In this example, a 5×5 pixel image with 4 grey scales $(0, 1, 2, 3)$ produces a matrix with elements summing to the number of horizontally adjacent pixels $(2N_y(N_x - 1) = (2)(5)(5-1) = 40)$.

$$
Image = \begin{bmatrix} 0 & 0 & 2 & 3 & 1 \\ 0 & 0 & 3 & 1 & 2 \\ 1 & 2 & 3 & 2 & 1 \\ 3 & 3 & 3 & 2 & 0 \\ 3 & 2 & 2 & 0 & 3 \end{bmatrix} \qquad P_{(1,0)} = \begin{bmatrix} 4 & 0 & 3 & 2 \\ 0 & 0 & 3 & 2 \\ 3 & 3 & 2 & 5 \\ 2 & 2 & 5 & 4 \end{bmatrix}
$$

FIGURE 2.30: A co-occurrence matrix for a synthetic image with 4 grey scales.

In [147] Haralick defines a class of quickly computable textural features from the co-occurrence matrix. The 14 textural features presented are known as angular second moment, contrast, correlation, sum of squares, inverse difference moment, sum average, sum variance, sum entropy, entropy, difference variance, difference entropy, information measures of correlation (2 of these) and maximal correlation coefficient. In general, the meaning of these features is unclear, although from intuitive inspection [147] it is deemed that the angular second moment, also referred to as energy, measures the degree of image homogeneity and the entropy measures image complexity.

Texture feature	Formula
Contrast	$\sum_i \sum_j (i-j)^2 P_{(d_x,d_y)}(i,j)$
Correlation	$\frac{\sum_i \sum_j (i-\mu_x)(i-\mu_y) P_{(d_x,d_y)}(i,j)}{\sigma_x \sigma_y}$
Energy	$\sum_i \sum_j P^2_{(d_x,d_y)}(i,j)$
Entropy	$-\sum_i \sum_j P_{(d_x,d_y)}(i,j) \log P_{(d_x,d_y)}(i,j)$
Homogeneity	$\sum_i \sum_j \frac{P_{(d_x,d_y)}(i,j)}{1+\lvert i-j\rvert}$

TABLE 2.3: Common features extracted from grey level co-occurrence matrices.[15]

The five most commonly used features [83] [111] [110] [148] are defined in Table 2.3.[15] Performance evaluation [148] suggests that *contrast* is the most important co-occurrence feature. Interestingly this measure is omitted in [111] which suggests Gabor filter responses are superior to co-occurrence features.

An alternative to using these features is shown in [149]. In this thoughtless example three co-occurrence matrices ($P_{(1,1)}$, $P_{(2,2)}$ and $P_{(4,4)}$) quantised to 16 grey levels are concatenated to form a 768 element feature vector with no consideration for symmetries, invariances or other image properties. The success of this "new unsupervised method" is concluded, although the illustrated results are very subjective.

Intelligent variations include generalised co-occurrence matrices which are well suited to macro textures [150] and multidimensional texture histograms which are simply extended to include more than two displaced image positions [126].

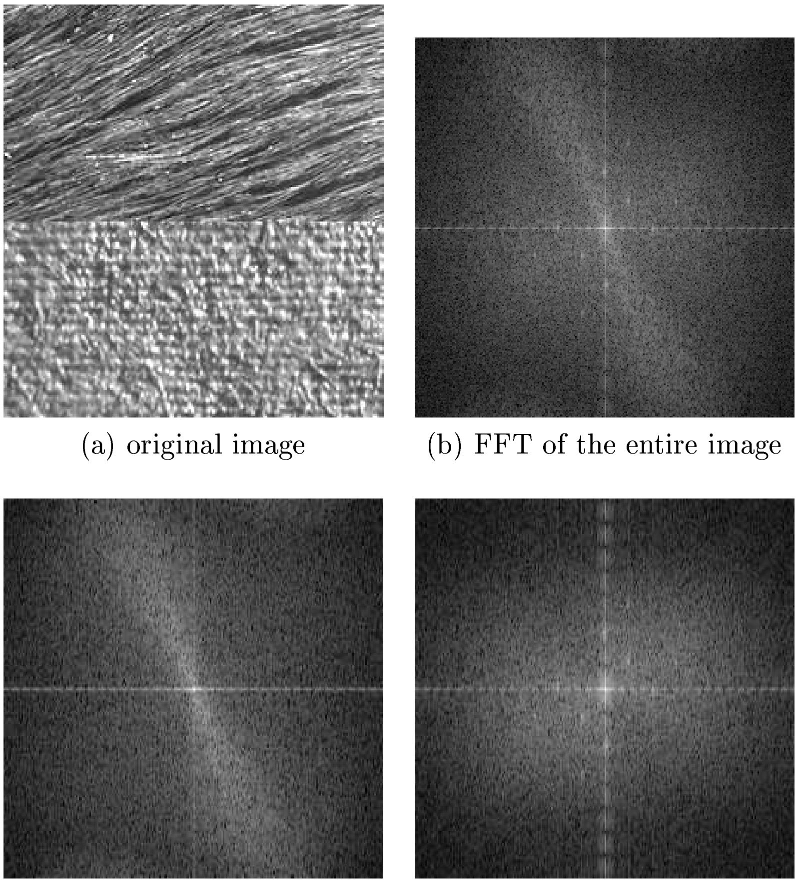
In conclusion co-occurrence matrices provide a benchmark for texture features which can be computed very quickly. Consequently they offer an attractive solution for texture database retrieval. Classification rates are shown in Table 2.2.

2.4.5 The frequency spectra of textures

As illustrated in Figure 2.31, the magnitude of the two dimensional FFT highlights characteristics such as texture orientation. The response of the top texture shows a strong orientation at around 120 degrees extending into the higher frequencies, thus also indicating a fine scale or micro texture. The bottom texture forms bands along the axes and a more heavily concentrated low frequency response corresponding to a blotchy although reasonably square oriented texture.

Such analysis is only applicable for textures with well defined orientations or periods and hence homogeneous frequency characteristics across the entire image (or region). It

[15] In all cases $P_{(d_x,d_y)}[i,j]$ represents a proportion, not an absolute histogram count. Also, μ_x and μ_y are the means and σ_x and σ_y are the standard deviations of $\sum_j P_{(d_x,d_y)}[x,j]$ and $\sum_i P_{(d_x,d_y)}[i,y]$ respectively.



(a) original image (b) FFT of the entire image

(c) FFT of the top region (d) FFT of the bottom region

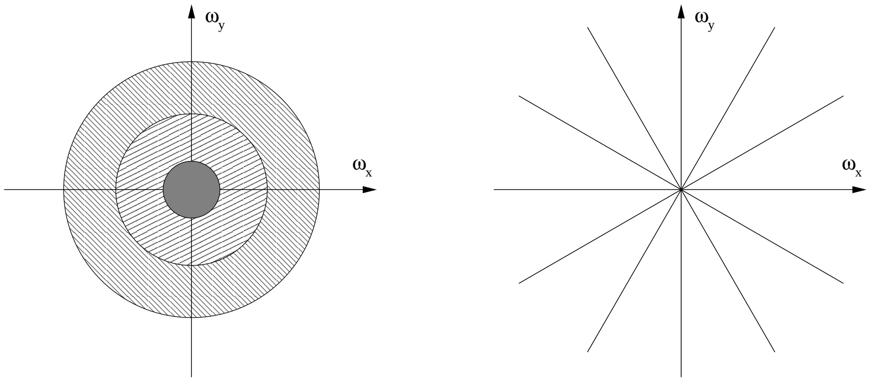
FIGURE 2.31: The frequency spectrum of textured images obtained using the two dimension FFT.

is suggested [142] that phase is not important for such texture features, although work by Daugman [151] on amplitude and phase demodulation indicates otherwise. Illustrations [151, Figures 1–7 and 15–17] show the effectiveness of describing the general underlying characteristics or background texture, but at the cost of the more subtle surface and positional detail. Such textural details may be encoded using phase.

A simple set of features based on the distribution of energy in the power spectrum is shown in Figure 2.32. These are adequate for some classes of texture, although lack phase information. Other variations include the use of autocorrelation functions from the power spectrum (see page 43).

Decomposition of the power spectrum allows a measure of texture coarseness and orientation [127]. In [152] the periodicity and orientation of the texture are determined by locating the maximum in the power spectral density. This allows a more intelligent selection of the displacement vector for the co-occurrence matrix.

A variation [153] measures the entropy in square frequency bands (similar to Figure 2.32(a)), using these as features. This work shows distinct clusters using three Bro-



$$f_{r_1,r_2} = \int_0^{2\pi} \int_{r_1}^{r_2} |F(\omega_x, \omega_y)|^2 dr d\theta \qquad f_{\theta_1,\theta_2} = \int_{\theta_1}^{\theta_2} \int_0^{\infty} |F(\omega_x, \omega_y)|^2 dr d\theta$$

$$r = \sqrt{\omega_x + \omega_y} \quad \theta = \arctan(\omega_y/\omega_x) \qquad r = \sqrt{\omega_x + \omega_y} \quad \theta = \arctan(\omega_y/\omega_x)$$

(a) texture feature indicating fineness/coarseness (b) texture feature indicating directionality

FIGURE 2.32: Texture features extracted from the power spectrum or power spectral density. From [83, Figure 12] with modifications.

datz textures and one random white noise image. Another variation involves fitting radial curves to the power spectral density, assuming an isotropic fractal Brownian surface [110].

In general, much information is held within the frequency spectrum. However, direct use of this relies on global characteristics which cannot easily be related to spatial position. It is more common and usually more appropriate to use joint spatial and frequency techniques such as Gabor filters or wavelets.

2.4.6 Gabor filters

"Daugman [89] proposed the use of Gabor filters in the modeling of the receptive fields of simple cells in the visual cortex of some mammals." [83, p263] This interpretation of the literature is perhaps misleading, as outlined in Section 2.3.2 along with a two dimensional definition of the Gabor function and related discussion. The similarity to receptive fields and the optimal joint space and frequency resolution make banks of non-orthogonal Gabor filters a popular choice for texture analysis.

It is clear from Table 2.2 that the use of Gabor filters is common for texture classification or retrieval. However, the number of filters and design varies considerably. For example [87] uses self similar Gabor filters (Gabor wavelets) with a wavelength of the sinusoid twice the standard deviation of the modulating Gaussian. Six orientations (0, 30, 60, 90, 120 and 150 degrees) and four linearly spaced frequency scales are used with the thresholded filters trained to distinguish target tanks from non-target vehicles. The results indicate a real danger of manning a non-target vehicle, although the illustrated data is extremely difficult even for a human expert. In [103] 4 orientations at 45 degree increments and 5 linear scales are used. Neither considers filter design.

Filter design is considered in [124] which utilises 4 orientations and 5 octave spaced scales. The filters are designed with respect to the frequency spectrum to obtain "the half-peak magnitudes of the frequency bandwidth, B_r, (in octave [sic]) and the orientation bandwidth B_θ (in degrees)" similar to the ones defined below for centre frequencies ω_0.[16]

$$\sigma_{radial} = \frac{\omega_0}{\sqrt{2}} \frac{2^{B_r} - 1}{2^{B_r} + 1} \qquad \sigma_{angular} = \frac{\omega_0}{\sqrt{2}} \tan(\frac{B_\theta}{2})$$

A minimum distance classifier is used on features from the average of each filter's magnitude response, with an accuracy of about 70% on 16 selected Brodatz textures. This is increased to 90% by using 10 orientations and 5 scales. [124]

A more sophisticated design approach provides a set of filters based on the desired frequency range (ω_L to ω_U), number of scales (s) and number of orientations (k) [131]. This design, which ensures "that the half peak magnitude support of the filters responses in the frequency spectrum touch each other", results in the following relationships.

$$a = (\omega_H/\omega_L)^{-\frac{1}{s-1}} \qquad \sigma_{radial} = \frac{(a-1)\omega_H}{(a+1)\sqrt{2\ln 2}}$$

$$\sigma_{angular} = \tan(\frac{\pi}{2k})[\omega_H - 2\ln(\frac{\sigma^2_{radial}}{\omega_H})][2\ln 2 - \frac{(2\ln 2)^2 \sigma^2_{radial}}{\omega_H^2}]^{\frac{1}{2}}$$

In addition to this, the real even symmetric component of the filter is modified by a constant to remove DC bias (refer to Figure 2.21 on page 51).

With this design method Manjunath and Ma [131] use the magnitude of Gabor filters at 4 scales and 6 orientations. The means and standard deviations form 48 features $f = [\mu_{00}, \sigma_{00}, \mu_{01}, \sigma_{01}, \ldots \mu_{35}, \sigma_{35}]$ with the distance between textures defined as

$$\mathbf{d}(i,j) = \sum_m \sum_n (|\frac{\mu^{(i)}_{mn} - \mu^{(j)}_{mn}}{\alpha(\mu_{mn})}| + |\frac{\sigma^{(i)}_{mn} - \sigma^{(j)}_{mn}}{\alpha(\sigma_{mn})}|)$$

The values $\alpha(\mu)$ and $\alpha(\sigma)$ are used to normalise individual feature components. Since texture classification (in this case database retrieval) is used, "$\alpha(\mu_{mn})$ and $\alpha(\sigma_{mn})$ are the standard deviations of the respective features over the entire database..."

Results use 116 different texture classes, 109 of which are Brodatz textures (labelled D1 to D112 with D31, D32 and D99 omitted for some reason). The database consists of

[16] The Gaussians are actually defined by standard deviations in the ω_x and ω_y directions, but this is only applicable for an orientation of 0 (or 180) degrees.

1856 128×128 images created from 16 non-overlapping 512×512 images. Performance is measured by the number of correctly classed images in the top 15 and top 100 retrieved images. Average results obtain performance of 74.37% and 92% respectively. The feature extraction for these relatively small images takes 9.3 seconds on a SUN-Sparc20 (1996).

Overall this demonstrates the effectiveness of power spectral density measurements from well designed Gabor filters for texture image retrieval. A further extension is to adaptively select distinct features by examining the power spectrum (see Section 2.4.5). As presented in [131] this reduces the processing time to 2.3 seconds and degrades retrieval performance to "about 50% of the correct patterns in the top 15 retrieved patterns." Again a simple distance measure is used.

$$\mathbf{d}(\omega_x, \omega_y) = \frac{[|F(\omega_x, \omega_y)| - |F_{mean}(\omega_x, \omega_y)|]^2}{(F_{var}(\omega_x, \omega_y))}$$

A further extension [107] uses texture classes identified by humans to improve the visual similarity of image retrieval (e.g. D8, D56, D64 and D65 are considered similar). The system, which uses self organising feature maps and vector quantisation for post feature learning, is illustrated on the same data. More quantitatively, it close to doubles the average retrieval rate of numerous regions from aerial photographs consisting of over 250000 texture objects.

Gabor filters are also used for rotationally invariant texture classification [129]. In this case 3 octave spaced scales and 4 orientations are used. Rotating the input texture by the orientation spacing of filters produces the same filter response by the adjacent filter. Hence, the feature at each scale can be considered as a periodic signal with the rotation of the input image equivalent to a rotation or phase change of this signal. Using this type of viewpoint the feature vector is further encoded using the discrete Fourier transform to separate it into rotationally variant and invariant components. With this approach classification rates for 30 textures (including 19 Brodatz textures) are determined using k-nearest neighbours, backpropagation and rules based classifiers. Each 256×256 image is decomposed into 16 non-overlapping regions and used at 5 degree rotational increments. Results of 95–97.5% and 84.5–91.5% are obtained for non-rotated and rotated textures respectively.

There are numerous other approaches to texture which use Gabor filters. Texture discrimination work in [154] compares the even symmetric component of the Gabor filter to the difference of Gaussians (DOG) filter at locating texture gradient. However there is no obvious conclusion as to which is better. In [111] features similar to those used by Manjunath and Ma [131] [107] are compared to 5 common co-occurrence features, suggesting Gabor filters are far superior and less sensitive to noise. In [127] a minimum distance classifier based on the Kullback Leibler distance of filter response distributions is used to obtain a classification rate of 98.7% with 16 textures.

In conclusion, Gabor filters provide an attractive approach which is well suited to texture classification and database retrieval. However, as illustrated above these functions have great scope for either mistreatment or adaptation to suit specific data. Methods demonstrated by Manjunath and Ma [131] [107] are extremely effective for database retrieval, although unfortunately are not generally applicable for segmentation or intra-image analysis.

2.4.7 Markov random fields

Markov random field (MRF) models can be applied to texture modelling and texture generation or synthesis. The concept of MRFs originated from a model to describe ferromagnetic materials proposed by German physicist Ernst Ising in the 1920s [155]. The *Ising model* provides a basic introduction to MRFs in a simple one dimensional case consisting of two states, or spins $s_i(\omega) = \pm 1$, for a sample sequence (or signal) $\omega = (\omega_0, \omega_1, \ldots \omega_n)$. For each configuration an energy $U(\omega)$ is assigned as

$$U(\omega) = -J \sum_i \sum_j s_i(\omega) s_j(\omega) - mH \sum_i s_i(\omega)$$

where J, m and H are constants based on the properties of the material and magnetic field intensity.

For such an energy, Ising assigned probabilities to configurations ω by

$$P(\omega) = \frac{1}{Z} e^{-\frac{1}{kT} U(\omega)} \qquad Z = \sum_i e^{-\frac{1}{kT} U(\omega_i)}$$

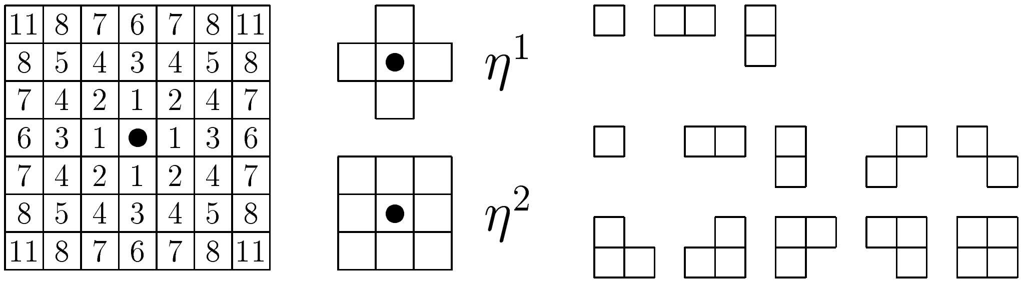
where T is the temperature and k is a universal constant. This probability measure is called Gibbs measure, or Gibbs probability distribution [156].

The Gibbs distribution is chosen because it maximises the uncertainty in terms of entropy and has an important nearest neighbours interaction property [155]. The latter, more important characteristic, provides a Markov type property for a set of neighbours N_j and an expected energy a. These probabilities represent local characteristics and define a Markov random field.

$$P(s_j = a | s_k, k \neq j) = P(s_j = a | s_k, k \in N_j)$$

Extending this basic theoretical framework to grey scale images requires a defined neighbourhood system and an increased number of states to cater for a defined number of grey levels. Adopting a common notation [142] [157] [158], a neighbourhood system η

$$\eta = \{\eta_{ij} : (i,j) \in L, \eta_{ij} \subseteq L\}$$



(a) expansion of neighbourhood orders (b) first and second neighbourhoods (c) associated cliques

FIGURE 2.33: Neighbourhood configurations used to define cliques for Markov random fields. Illustrations with similar content are presented in [159] [157] [160] [161] [158] [83] and [162] to name a few.

is defined on a square region or lattice L by the relationships,

$$(i,j) \notin \eta_{ij} \text{and} (k,l) \in \eta_{ij} \Leftrightarrow (i,j) \in \eta_{kl} \forall (i,j) \in L$$

Figure 2.33(a) shows the neighbourhood systems typically used for image processing, referred to as η^i, where η^1 denotes the system of first order neighbourhoods consisting of the four nearest neighbours of each pixel and so on. Figure 2.33(c) shows the set of *cliques* (c) for the first two ordered neighbourhoods (η^1 and η^2). A clique is defined to consist of groups of pixels such that "if $(i,j) \neq (k,l)$, then $(i,j) \in c$ and $(k,l) \in c$ together imply that $(i,j) \in \eta_{kl}$; C is the all cliques for the neighbourhood system η over the lattice L;" [157].

Given a lattice, neighbourhood system and set of cliques, a random field $X = \{X_{IJ}\}$ is defined with a Gibbs distribution according to the Ising model as

$$P(X = x) = \frac{1}{Z} e^{-U(x)} \qquad \text{where } U(x) = \sum_{c \in C} V_c(x) \text{ and } Z = \sum_i e^{-U(x_i)}$$

Once again Z is a normalisation factor and $V_c(x)$ defines the potential (or energy) associated with clique c.

As outlined in [83] there is a unique Markov random field for every Gibbs random field with respect to a given neighbourhood and vice versa. Consequently, texture can be modelled globally by the total energy of the lattice or locally by specifying local clique interactions in terms of conditional probabilities.

A variety of potential functions are used. Geman et al. [163] suggest that the function chosen should be monotonic to selectively encourage similarity or dissimilarity and be bounded to accommodate for occasional random clique relationships. In [157], the potential function is parameterised by simple pixel clique values trivially and for all other cliques by the following regardless of size.

$$V_{ck}(x) = \begin{cases} B_k & \text{if all } x_{ij} \in c_k \text{ are equal} \\ -B_k & \text{otherwise} \end{cases}$$

Other work tends to restrict the model to cliques containing at most two pixels, hence the energy function is simplified to the following form,

$$U(x) = \sum_{i \in L} V_i(x_i) + \sum_{i \in L} \sum_{n \in \eta} V_{in}(x_i, x_n)$$

where the V_i's are the single site potentials and the V_{in}'s are the two site potentials [160, p29]. Other variations of this form and generalisations of the Ising model for first and second order neighbourhoods are presented as such in [162] [159] and [163].

Two more common texture models which only consider single pixel and pairwise pixel cliques of η^2 are the auto binomial model and Derin-Elliot model [158]. In these models, parameterised by $\theta = [\theta_1, \theta_2, \theta_3, \theta_4]$, the energy is given by

$$U(x) = \frac{1}{2} \sum_{l \in L} [w_1(x_i), w_2(x_i), w_3(x_i), w_4(x_i)]^T [\theta_1, \theta_2, \theta_3, \theta_4]$$

where $w_r(x_i)$ is defined in the following form in [83].

Auto-binomial model:	$w_r(x_i) = x_i(x_{i-r} + x_{i+r})$	$1 \leq r \leq 4$
Derin-Elliot model:	$w_r(x_i) = I(x_t, x_{i+r}) + I(x_i, x_{i+r})$	$1 \leq r \leq 4$

"Here r is the index that defines the set of neighbouring pixels or a site i, and $I(a, b)$ is an indicator function as follows:"

$$I(a, b) = \begin{cases} -1 & \text{if } a = b \\ 1 & \text{otherwise} \end{cases}$$

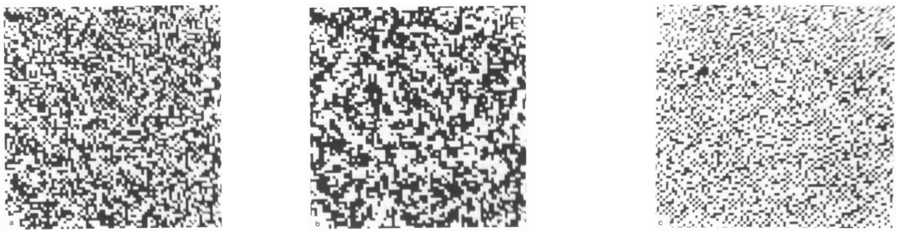
Implementations using these two models appear in [110] and [158] respectively.

An example of texture synthesis is presented in [162]. A first order neighbourhood is used with the energy function defined as

$$U(x) = a \sum x(i, j) + b_1 \sum x(i, j-1) x(i, j) + b_1 \sum x(i-1, j) x(i, j)$$

Furthermore, the MRF is assumed to be isotropic (rotationally invariant), hence $b = b_1 = b_2$. Figure 2.34(b) and (c) shows texture produced using specified values for a and b.

MRFs are not as common as Gabor or co-occurrence matrix features for use in texture classification and retrieval. In [132] an MRF provides a measure of importance for pairwise clique families of a large range of displacements. These are used along with grey level



(a) initial 80x80 random field (b) isotropic random field $a = -6$, $b = 1.5$ (c) isotropic random field $a = 6$, $b = -3$

FIGURE 2.34: Texture synthesised using an MRF model. Taken directly from [162, Figure 9].

difference histograms to extract texture features for database retrieval. This modelling assumes the image database only contains homogeneous images (presumably with respect to local textural characteristics). The scheme is illustrated on 30 selected Brodatz textures with retrieval rates favourably presented according to the existence of at least *one match in the first N* retrieved images. Catering for small image rotations, results yield 88.8% of correct first choices and 96.7% among the top two choices. Results on 12 *VisTEX* texture classes yields 76.8% and 84.9% although *inappropriate* classes and inhomogeneous textures within other classes are excluded.

The simultaneous autoregressive (SAR) model also offers a neighbourhood based approach to texture analysis. These models, which are an instance of MRFs [141], provide rotationally variant texture features for arbitrary sized neighbourhoods. A rotationally invariant version is proposed in [141]. SAR models are often used for multiple neighbourhoods, referred to as multiresolution SARs (MR-SARs) [130] [131] [141].

The main relevant application for MRFs is in texture segmentation as presented in Section 2.5.6. Other applications include use in magnetics, physical systems (e.g. gases and binary alloys), biological systems (e.g. cell structures, motion of birds, etc.) and higher level more abstract areas such as sociological and political models and models to stabilise economies [155]. In such models energy is often analogous to, or replaced by, tension.

2.4.8 Closing comments

Throughout this section a number of major texture analysis methods have been presented. Those not covered include Laws' *texture energy measures* [142] [148], fractal models, tessellation techniques [83] and syntactic or grammatical tree representations [164]. Laws' measures use convolution masks to measure local average, derivatives and, on a second pass, deviation. Tessellation and syntactic techniques attempt to model geometric structure.

Studies and statements comparing different techniques are subjective and implementation dependent. In [110] four classes of texture features are compared, namely co-occurrence matrices, second-order auto-binomial MRF, Gabor filter and fractal features. It is suggested that "co-occurrence features performed best in our experiments and should certainly be tried first when working with small images" although reservations are made due to the lack of theoretical foundation behind co-occurrence matrices. MRFs are suggested to be only applicable for large images or regions and the Gabor filters are badly designed with features based on average absolute deviation (AAD) and energy for each filter. "The Gabor features are not recommended in the form we used" [110]. Other performance evaluations [148] also conclude co-occurrence matrices, in particular the contrast feature, to be favourable.

Generalised co-occurrence matrices, MRFs and Gabor filters all cater for Julesz texture theories in terms of neighbourhood relationships and second order statistics. This justifies the use of these methods for modelling preattentive texture discrimination (or effortless discrimination) in biological vision systems. However, it is unclear whether this in itself is sufficient for intelligent image analysis or higher level object classification.

In summary, co-occurrence matrices and MRF models use individual pixel based measurements and are hence not easily flexible to changes in image resolution. On the other hand Gabor filters can trivially be scaled, or designed in the frequency spectrum to account for resolution changes possibly incurred by changes in image acquisition. Similar properties apply for simple spatial window based measurements. For applications in texture classification and image retrieval, results by Manjunath and Ma [131] [107] indicate the power of well designed Gabor filters using almost the entire Brodatz texture set. Other simple texture measurements, such as coarseness and local standard deviation, also cater for resolution changes and are a simple yet effective form of texture analysis well suited for preprocessing.

2.5 Segmentation

Image segmentation methods are directly applicable to the problem of decomposing images into sub-images. By a glossary definition, "*Image Segmentation* is a process that typically partitions the spatial domain of an image into mutually exclusive subsets called regions. Each region is uniform and homogeneous with respect to some property, such as tone or texture, and its property value differs in some way from that of each neighbouring region." [40, p593] This is clearly a subproblem of the decomposition problem outlined in Chapter 1.

As is apparent in this section there are numerous general approaches to image segmentation which usually combine *standard* methods with *ad hoc* and often application specific ones. Many segmentation categories are suggested, for example the principal approaches can be categorised into region based schemes, edge based methods and clustering [125]. There is much overlap in these particular categories which fail to cover all approaches.

Morel [165] addresses the question, "Is there a general theory for segmentation?", concluding that there are several theories in surrounding techniques and in components of specific segmentation algorithms, but no general theory. Furthermore, there are inevitable tradeoffs, for example, it is desirable to smooth homogeneous regions for noise elimination, but on the other hand keep accurate location of region boundaries. Haralick [166] also raises this problem:

> "What should a good segmentation be? Regions of an image segmentation should be uniform and homogeneous with respect to some characteristics such as gray tone or texture. Region interiors should be simple and without many small holes. Adjacent regions of a segmentation should have significantly different values with respect to the characteristics on which they are uniform. Boundaries of each segment should be simple, not ragged, and must be spatially accurate.
>
> Achieving all these desired properties is difficult because strictly uniform and homogeneous regions are typically full of small holes and have ragged boundaries. Insisting that adjacent regions have large differences in values can cause regions to merge and boundaries to be lost."

In view of this, the following assertion is made.

> "As there is no theory to clustering, there is no theory of image segmentation. Image segmentation techniques are basically ad hoc and differ precisely in the way they emphasise one or more of the desired properties and in the way they balance and compromise one desired property against another." [166]

As a result there are many approaches to image segmentation using different methods and implicit or explicit assumptions. Some schemes are very general while others irrationally assume prior knowledge. Important segmentation principles such as proximity, similarity, common fate, connectedness and good continuation [167] tend to be qualitative in nature. In general techniques are typically justified by results or loose correspondence to biological vision systems.

2.5.1 Scope and types of images

Throughout this thesis the restriction is made to static grey scale images, although multi dimensional images are considered in places. In these cases, each dimension is assumed to be an independent measure and hence no consideration is made for colour representations or transformations. Colour and colour specific image segmentation are covered in Sections 2.6 and 2.6.5.

Three dimensional image segmentation, e.g. [168], and optical flow techniques on consecutive images or on an individual image [169] [170] [28] are out of scope (see [40] for a general but comprehensive coverage). Hence techniques to obtain three dimensional shape from surface texture and to reconstruct scenes from stereo images are not covered.

Furthermore, contrived segmentations for which *correct results* are not obvious by a human observer will be overlooked. These are typically artificial images designed with considerations for the properties of the algorithm and used for comparison against other algorithms. See [148, Figure 10] or Figure 2.31 for good and borderline examples respectively.

The types of images within scope include:

- Tone images. Images where the intensity value corresponds to a physical value for which segmentation is applicable. MRI and X-ray images are specific examples of tone images since the image value directly corresponds to the physical properties of the material requiring segmentation.
- Textured images. Refer to Section 2.4.
- Natural scenes. Images containing a single view of a natural or real-world scene. This huge class of images includes analysis of photographs, synthetic aerial radar (SAR) images and those captured from digital cameras which are often used by robots.
- Images specific to medical applications. Although these are within scope, less emphasis is placed here. These images benefit from heavily knowledge dependent segmentation schemes and have been studied extensively [5]. They are less applicable to a generalised image decomposition.
- Text document images. Images containing a mixture of text, pictures and possible background. These include work station screens and printed media such as newspapers. These are addressed specifically.

2.5.2 Histogramming and thresholding

Thresholding is the simplest form of image segmentation, in which image values above a certain value, or threshold (t), are assigned the value 1 and other pixels the value 0, or vice versa. Essentially this is a simplified clustering problem in one dimension with two clusters (or more clusters for multiple thresholds) (see Section 2.1.5). Often a predefined, or static threshold is used, but it is more general to choose single or multiple thresholds dynamically. This choice is typically based on a histogram ($P(x)$) which provides a probability density function (p.d.f.) of the pixel values of an image. In principle thresholding is well suited for tone images.

Standard techniques

One of the first dynamic thresholding techniques is to minimise the 'within group' variance $\sigma_w^2(t)$. This measure, proposed by Otsu in 1979, is defined according to the weighted sum of variances

$$\sigma_w^2(t) = \alpha(t)\sigma_1^2(t) + (1 - w(t))\sigma_2^2(t) = (\sum_{x=0}^{t} P(x))\sigma_1^2(t) + (\sum_{x=t+1} P(x))\sigma_2^2(t)$$

The $\sigma_i^2(t)$ terms correspond to the variance of pixel intensities above and below the threshold, which along with the relative weight $\alpha(t)$ can be formulated directly from the histogram and easily rearranged into a recursive relationship as shown in [20].

Otsu's measure is well suited to bimodal histograms where approximately the same number of pixels are above and below the threshold, but tends to fail to locate relatively small objects and is susceptible to over saturated images. For example, a relatively small dark object on a mostly white and heavily saturated image will tend to produce a higher than ideal threshold than would be expected intuitively.

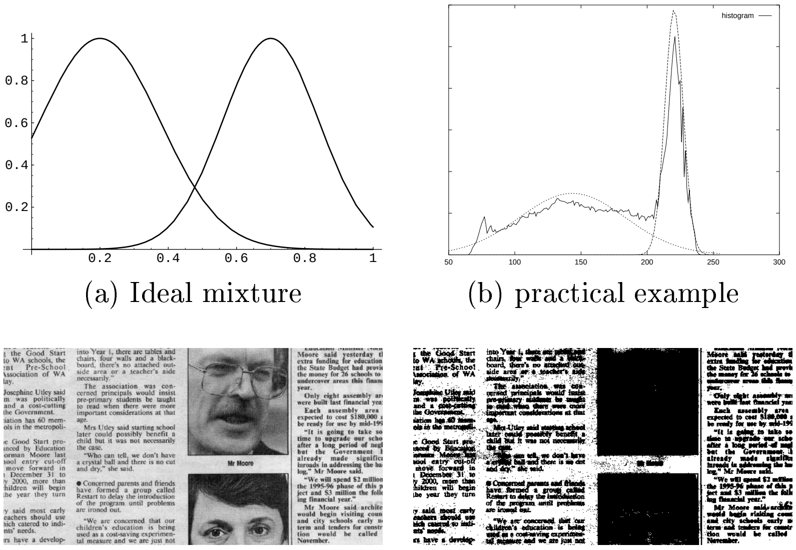
An alternative set of algorithms presented and compared in [171] assume the histogram is a mixture of two Gaussian distributions, determining the corresponding weight (α), means (μ_1 and μ_2) and standard deviations (σ_1 and σ_2). Hence, the p.d.f. is a mixture of two one dimensional Gaussians, of the general form

$$P(x) = \alpha g_0(x) + (1 - \alpha)g_1(x), 0 < \alpha < 1 \qquad \text{where } g(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{1\frac{x-\mu}{2\sigma^2}}$$

With this framework a threshold can be determined quite easily be assuming that the weight and standard deviation of each Gaussian is the same (i.e. $\alpha = 0.5$ and $\sigma_1 = \sigma_2$) as proposed by Ridler and Calvard in 1979 [168]. Kittler and Illingworth provide a more general solution which minimises the Kullback Leibler distance between the histogram and the Gaussian model [20] and is claimed to produce the minimum error [171] [17]. A more theoretical approach which minimises the probability of misclassification given two model Gaussians is depicted by the valley in Figure 2.35. When formulated as such, the thresholding problem is a simplified version of the Gaussian mixture model (presented in Section 2.1.5) restricted to two mixtures and one dimensional data.

The performance of these Gaussian mixture methods depends on how well the histogram data fits the assumed model. Evaluation on synthetic images in which classes are predefined suggests the simplest method proposed by Ridler and Calvard is superior [171]. Performance is based on accuracy, stability, speed and consistency, and is a little subjective.

In [172] a threshold selection scheme is proposed based on minimising the cross entropy from an information theory viewpoint. The formulation uses the Kullback Leibler distance between the candidate thresholded histograms and a distribution for which no prior information is available. Hence it can be seen as an extension to the standard maximum entropy method. This corresponds to the following criterion function which is simplified



(a) Ideal mixture (b) practical example

(c) corresponding image (d) threshold using (b)

FIGURE 2.35: A mixture of two Gaussians used to model image histograms. The EM algorithm forms two stable Gaussians with weights 0.59 and 0.41, means 143.5 and 220.3, and standard deviations 40.06 and 6.95, respectively.

based on the mean and number of points in each distribution.[17]

$$e(t) = \sum_{x=0}^{t} P(x)\log(\frac{P(x)}{\mu_1(t)}) + \sum_{x=t+1} P(x)\log(\frac{P(x)}{\mu_2(t)}) = -\mu_1 \log\frac{\mu_1}{n_1} - \mu_2 \log\frac{\mu_2}{n_2}$$

Comparison on scanned images between this, Otsu's, and Kittler and Illingworth's methods are illustrated, suggesting the cross entropy measure to be superior and more tolerant to real distributions. When applied to a mixture of two Gaussians it outperforms Otsu's method, although is obviously inferior to Kittler and Illingworth's as illustrated in [172].

Another common global thresholding technique is to use image gradients to determine the relevance of pixels, omitting those with strong gradients from the histogram. The idea is to eliminate the biasing effect of pixels at the edge of objects. Scattergrams [35] [166] provide a tool for such analysis as seen in Figure 2.36. A thresholding algorithm which uses image gradient is presented in [173, p78]. An alternative [84, p599] is to form a histogram using only the pixels with a large positive or negative response to the Laplacian operator (second partial derivative). This tends to produce a well defined bimodal distribution with a distinct valley but is very susceptible to noise.

Adaptive and local thresholding techniques are well presented and discussed in [35, p77-99]. These handle factors such as global lighting variations and will not be discussed.

[17] This equation is loosely based on that derived in [172], although has been rearranged somewhat to maintain consistency in variable names and general form.

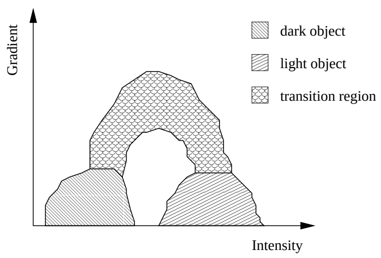


FIGURE 2.36: The ideal scattergram allows for basic image segmentation of tone images. [35]

Specific examples

In practice application specific approaches are often chosen in preference to the more general methods outlined above. The segmentation examples here are based on local or global histogram techniques.

The first example is both very application specific, poorly designed and typical of a threshold implementation. In [17] a histogram is taken along the principal eigenvector intensity and colour saturation plane in order to locate sections of road for autonomous road navigation. This histogram is smoothed and analysed to locate either a bimodal or trimodal distribution (to account for shadowed road) through an ad hoc algorithm which assumes ranges for both the threshold and the proportion of road versus non-road in the image. The results are far from impressive.

An alternative approach for thresholding involves decomposing the co-occurrence matrix[18] of an image [174]. The approach uses the co-occurrence entropy measure to maximise the information between regions by decomposing the matrix into square sections down its diagonal. Each of multiple thresholds is selected by maintaining the highest block entropy since the last chosen threshold and selecting a new threshold when the block entropy is reduced to a fraction of this given that this local maximum is large enough. More precisely, for histogram $H_t(s)$ consisting of block entropy measurements, H_{max} is initialised to 0 with a chosen threshold and H_{max} reset to 0 whenever $(H_{max} - H_t(s) > C_1 H_{max})$ and $H_{max} > 1$. The histogram position s is incremented from 0, C_1 is a constant to handle histogram noise and H_{max} updated each iteration using $H_t(s)$. The condition $H_{max} > 1$ imposes a restriction on the lowest local maximum after which a threshold can be chosen. The results of this global thresholding are impressive indicating the effectiveness of using intuitive methods on histograms, or in this case, co-occurrence measurements.

In [175] a three dimensional clustering problem (using colour image data) is reduced to a series of one dimensional histogram decompositions. By thresholding the most suitable or well behaved histogram h_1, pixels are classed to allow the construction of multiple histograms from h_2 and h_3. These are in turn thresholded where appropriate to locate regions of the space which loosely correspond to clusters. The process is extremely fast

[18] Presented in Section 2.4.4.

and results are subjectively good, seeming no worse than more computational standard clustering techniques. A more conventional cluster based segmentation is presented in [109] in which local geometric moments are fed into the CLUSTER algorithm (see Section 2.1.5) to successfully identify different textures within an image.

For medical applications thresholding is appropriate for classifying X-ray images [176] and MRI images [177]. In [176] histograms with a bin size of 5 are employed with a custom-designed neural network to locate 3 peaks in the inverted histogram and hence threshold X-ray images into four classes. Results on two selected images are reasonable although inferior to manual selection. In [177] it is noted that MR images are tone images with short term correlation and hence well described by local histograms. Furthermore, it is assumed that local blocks chosen small enough to contain one tissue type are well modelled by a Gaussian distribution. The global histogram is approximated by fitting a predefined number of Gaussians which are used to initialise class templates. Local histograms along with the Kullback Leibler distance then provide an iterative algorithm to refine these templates producing a segmentation based on image intensity.

A more sophisticated threshold based segmentation considers background, object and noise intensity components, using morphological operations to help distinguish between these [178]. It assumes that the noise is the primitive of smallest size and the background the largest. Hence, by constructing a set of images using morphological opening operations and forming a histogram-like plot of the mean intensities of this set of images, the sizes of each primitive is determined by locating the three dominant peaks. Peaks with the largest difference to immediate neighbours are selected. With the size determined, other morphological operations allow the background and noise to be removed, reducing the problem to one of global thresholding. This is tackled using Otsu's measure, locating a smooth well defined noise-free object as opposed to results obtained by directly using Otsu's method or using alternative background removal techniques. The results on synthetic and real images illustrate the effectiveness of using spatial relationships rather than simply locating a noise corrupted valley in a bimodal distribution. [178]

As clear from above, thresholding techniques are well established but only generally applicable for images with well defined characteristics, such as modal histograms. In practice thresholding is often only applicable as a pre- or post-processing tool.

2.5.3 Edge detection

Detecting edges of regions or objects provides a means for image segmentation. Canny [179] suggests the three performance criteria for edge detection are good detection, good localisation and only one response to a single edge. For segmentation however, it is also necessary either to obtain closed edges or to derive an algorithm or form of post-processing to complete region boundaries.

The two main methods of edge detection are differential gradient and template matching techniques. Differential gradient operators are typically performed in x and y directions with the gradient obtained by $g = \sqrt{g_x^2 + g_y^2}$, or the zeroth or limiting Minkowsky metric (see page 32) to save computation. Table 2.4 shows convolution masks for numerous differential gradient operators presented in [84]. Canny [179] provides a means for maximising the signal to noise ratio and localisation constrained to a single response factor, resulting in a mask which varies from a boxcar function to the derivative of a Gaussian based on this factor.

Operator name	Horizontal mask	Vertical mask
Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Separated pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Frei-Chen	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$

Table 2.4: Convolution masks used for differential gradient edge detection. [84] [35]

Template matching edge detectors utilise eight convolution masks at 45 degree increments, as illustrated for 0 and 45 degrees in Table 2.5. These, which have originated on an intuitive basis (e.g. Kirsch masks) to compensate for human eye characteristics (e.g. Robinson masks) [35], are more computationally intensive requiring 4 or 8 convolution operations depending on mask symmetry. Larger templates can be used to provide finer quantisation of the edge orientation angle and greater noise immunity at the cost of localisation and computation.

Second order derivative edge detectors, such as the Laplacian operator ∇^2 in Figure 2.37, offer an alternative. The zero crossing of ∇^2 indicates the presence of an edge. For the continuous case or in images with certain smoothness constraints the zero crossings

Operator name	**Horizontal mask**	**Mask at 45 degrees**
Prewitt masks	$\begin{bmatrix} 1 & 1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & -1 & -1 \\ 1 & -2 & -1 \\ 1 & 1 & 1 \end{bmatrix}$
Kirsch masks	$\begin{bmatrix} 5 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & -3 & -3 \end{bmatrix}$	$\begin{bmatrix} -3 & -3 & -3 \\ 5 & 0 & -3 \\ 5 & 5 & -3 \end{bmatrix}$
Robinson *3 level* masks	$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & -1 \\ 1 & 0 & -1 \\ 1 & 1 & 0 \end{bmatrix}$
Robinson *5 level* masks	$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & -2 \\ 1 & 0 & -1 \\ 2 & 1 & 0 \end{bmatrix}$

TABLE 2.5: Convolution masks used for template matching edge detection. [35]

are closed curves [84].

For a more thorough general overview of edge detection techniques, theory and operator design refer to [35, p100–127], [84, p491–556] [20, p337–351] or [37, p347–357]. A novel orientation independent approach incorporating edge thresholding into the algorithm at each pixel location is presented in [180]. A fuzzy edge detector and relevant comparisons is proposed by Bezdek [181]. Other variations appear in [182], [55], [137], [183] and [184].

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

(a) Laplacian operator

$$\frac{1}{4}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

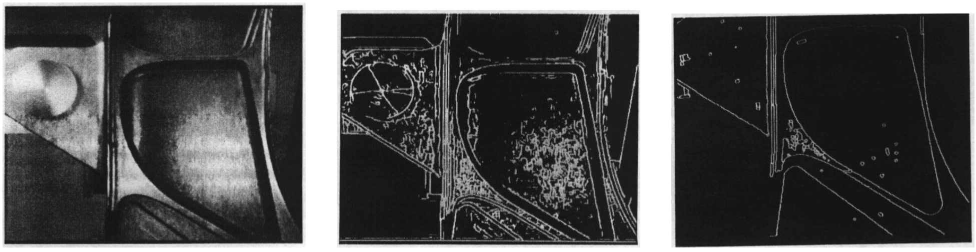
(b) 4 neighbour approximation

$$\frac{1}{8}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

(c) 8 neighbour approximation

FIGURE 2.37: The Laplacian operator and two commonly used convolution mask approximations.

Although edge detection results are often presented as segmentations, strictly speaking segmentation requires the identification of regions or location of bounded edges. While the Laplacian operator does provide this in theory, results often contain unobservable edges, and closed curves are insignificant or unattainable due to image quantisation effects. Haralick [20] suggests modelling neighbourhoods of second derivative edge operators using cubic polynomials to determine the edge direction and change in edge direction. A segmentation based on this method is shown in Figure 2.38.



(a) *bulkhead* image

(b) edges detected by locating Laplacian zero crossings

(c) segmentation using neighbouring edges

FIGURE 2.38: Segmentation using edge detection from Haralick [20, Figures 10.10, 10.27 and 10.28].

Many other edge linking methods are suggested for various template and differential edge operators [84, p610–622]. For example, a heuristic approach by Roberts uses thresholds for edge magnitude and correlation between edge directions to link neighbouring edges within 23 degrees of one another. Segmentation boundaries are tentatively marked and disregarded if they fail to meet the linking criteria.

Zhou *et al.* [182] tackles the line linking problem by heuristically defining clique sets for each line direction. These cliques provide tolerance to noise, slight directional changes, edge gaps and other common variations at a scale of two to three pixels. The results are impressive in finding line segments and locating corners, but do not provide segmentation as such.

In [183] segmentation is achieved by locating change in the local or instantaneous variance. This simple process, which is performed by convolution masks, variance estimation and one dimensional change detection row- and column-wise, produces a typical unconnected edge map. However, by iteratively smoothing the edge map by combining local edge (or in this case variance), measures connected edges and hence disjoint bounded regions are formed. Results on images containing several Brodatz textures indicate the effectiveness of edge based segmentation using local variance to account for texture.

Similar results for Brodatz texture segmentation are produced in [137]. In this case it is assumed that a low resolution segmentation has already been performed. Robinson edge masks (Table 2.5) are used with a linking method to combine edges in local neighbourhoods and refine the boundaries between segments. The located connected boundary is further refined to maximise a heuristically derived merit function favouring smooth boundaries. The complete algorithm and implementation are well presented in [137].

In general, while edge detection methods are useful for segmentation they typically require other techniques to consider image regions. Region growing using edge detection and local edge histograms provide useful tools for image segmentation.

2.5.4 Regions growing and split merge techniques

The general idea of region growing methods is to partition an image into homogeneous regions by starting with small regions or region seeds and growing them according to some homogeneous criterion. Such techniques suffer from similar problems to edge detectors:

> "If we could do region analysis, then edges could be declared at the borders of all regions. If we could do edge detection, regions would be the areas surrounded by the edges. Unfortunately, we tend to have trouble doing either: edge detectors are undoubtedly noisy and region growers often grow too far." [185, p171].

Region growing schemes can be classified into three basic types, namely single linkage, hybrid linkage and centroid linkage [20]. For these, regions are grown according to adjacent pixels, local neighbourhoods and entire region properties, respectively. Hybrid combinations and variations of these basic classes are common. Furthermore, as outlined in [186], the criteria used to accept new elements can vary as the region grows.

The result of a basic region growing algorithm is shown in Figure 2.39[19]. The same data produces very similar segmentation by applying the following extremely simple region growing algorithm [188]. Take the first pixel (or next unassigned pixel) as the seed pixel and grow the region including all connected pixels whose grey level is within a predefined threshold, i.e. $|G - G_s| \leq T$. As outlined in [188]. "A low T would result in an image that had a large number of regions and a high T would result in the opposite." This trivial algorithm is suggested to be very effective, although the illustrated images do not appear to be very useful even for the chosen values for T. Using this algorithm with $T = 6$ on the data below forms two major regions (with intensity ranges 142 ± 6 and 130 ± 6) and 5 single isolated pixels. While these regions are similar to those shown below (from [187]), changing the value of T to *any other value*, or changing the seed pixel (to 135 for example) illustrates the ineffectiveness of this scheme.

In [189] a region growing technique with considerations for edge detection is presented. The scheme is simple but effective, using an adaptive Gaussian smoothing filter with a variable standard deviation according to the local image gradient ($\sigma^2(i,j)e^{-\frac{G^2(i,j)}{K}}$, for constant smoothing parameter K). The algorithm, most easily explained by the equations below, considers each pixel $f(i,j)$ connected to the current region, examining a weighted function $z(i,j)$ consisting of the local gradient $G(i,j)$ and relative contrast $x(i,j)$. If $z(i,j)$ is less than a given threshold β then $f(i,j)$ is added to the region and the average intensity X_{av}^{N+1} updating accordingly. The parameters ϕ and β control the weight of the local gradient and the general contrast measure respectively. This algorithm is obviously only applicable to tone images as illustrated for MRI images in [189].

$$x(i,j) = |X_{av}^N - f(i,j)| \qquad z(i,j) = (1-\phi)x(i,j) + \phi G(i,j)$$

[19] Result taken directly from [187] in which a framework, but no algorithm, is presented.

(region 1: average=140, standard deviation=1.6, minimum=136, maximum=144)

142	141	140	139	138	140	140	140	143	142	139	139	142	140	138	144	139	142	140	140	141	142	139	143	139
139	140	139	140	141	140	144	140	142	138	141	139	140	138	141	139	138	140	138	139	137	142	136	140	140
141	141	140	139	139	138	139	140	138	141	140	143	140	141	138	138	140	138	138	140	140	142	142	140	138
130	132	130	126	134	132	135	133	133	132	129	129	123	131	129	129	132	131	137	137	130	124	131	138	136
129	135	132	134	135	132	127	138	132	129	125	138	138	136	135	129	138	130	139	126	134	136	126	131	132
137	132	130	134	142	134	134	138	126	132	132	130	131	127	126	131	124	133	133	132	132	126	134	132	134

(region 2: average=132, standard deviation=4.2, minimum=123, maximum=142)

FIGURE 2.39: Example result from a simple region growing method. This result is very subjective with the values 138 and 139 immediately below the boundary debatably misclassified. The numerical data is from [187] with the image constructed accordingly.

$$X_{av}^{N+1} = \frac{N}{N+1} X_{av}^{N} + \frac{1}{N+1} f(i,j)$$

The *Fisher distance* offers a more general criterion and is a popular choice for merging regions [190] [191]. The distance is defined below for two regions with known numbers of data points (n_1,n_2), means (μ_1,μ_2) and standard deviations (σ_1,σ_2). A major drawback of the Fisher distance is that it cannot distinguish between two distributions with the same mean but different variances where it clearly reduces to a value of 0.

$$\text{Fisher distance} = \frac{(n_1 + n_2)(\mu_1 - \mu_2)^2}{n_1 \sigma_1^2 + n_2 \sigma_2^2}$$

Split-merge algorithms offer a more general approach to region identification by allowing regions to be separated or merged together. These are more powerful, require less computation, allow more general initialisation and are more reliable than algorithms with only a split or merge component as discussed in [173, p90–93]. One of the most common split-merge frameworks uses a quad-tree decomposition to repeatedly divide an image region into 4 subregions until the resulting regions obey a defined homogeneous criterion. Neighbouring regions with appropriate characteristics are then merged.

An alternative is to split the image to a certain resolution regardless of region characteristics before performing merging. This is basically a region growing algorithm which combines or grows small regions into larger segments. An example is presented in [192], in which each region is modelled by a one dimensional Gaussian distribution using estimates for the mean and variance. Regions are merged on the basis of these models. This is successfully illustrated on synthetic images containing two random fields of different means or different variances. The question as to how well this model and segmentation approach fits real images is "still open because of the wide variety of natural images." [192].

Similar region modelling along with the more standard quad-tree decomposition is used in [193], [194] and [195].[20] The image is split into regions down to a limiting minimum size of 4×4 pixels for which the Gaussian model is considered a reliable estimate. Adjacent regions, which may vary in size, are conservatively merged using thresholds between region means and variances until no new merges are possible. A final phase uses the Euclidean distance of RGB colour to perform additional merging. The final results are reasonable, although subjective and only illustrated on the same sample natural scenes in all papers.

The minimum description length (MDL) principle [66] offers a common and more theoretical formulation for region growing or merging [196] [190] [191]. The MDL principle states that the best estimate is the one which results in the most compact encoding. For example, if a set of 1000 points can effectively be modelled by a mixture of two weighted Gaussians, it is more compact to store the means, standard deviations and weight parameters than to explicitly define the points.

In [196] each region is modelled by the p.d.f. of a Gaussian, with the MDL criteria formulated based on the number of regions, the dimension of the data, the length of the regions' boundaries and the determinant of the co-variance matrices. The algorithm begins by considering each pixel as a separate region, combining the adjacent pixels or regions which reduce the MDL cost by the largest amount. This process is repeated until no more adjacent regions can be combined to reduce the cost, at which time spatially disjoint regions are considered. This results in a (possibly local) minimum configuration according to the MDL based objective function. The results indicate the effectiveness of the method which maintains smooth region boundaries due to the boundary length consideration. The formulation presented in [196] is applicable to tone images and seems extremely well suited to two band MRI images, although is not applied to these.

Several other variations and applications to region growing and split merge algorithms exist. For example, region merging using a genetic algorithm to improve an initial fuzzy c-means segmentation [143], artificial intelligence or knowledge based systems applied to merge and locate regions aided by quad-tree decomposition [197], the use of local histogram features for region merging [198] and applications in image compression [144]. In each case neighbouring regions are considered as candidates for merging based on some predetermined criteria. As seen throughout this section, such criteria and associated parameters or thresholds govern the success of these algorithms. The common goal is to locate regions with homogeneous characteristics.

2.5.5 Segmentation using Gabor filters

Gabor filters are defined in Section 2.3.2 and discussed with respect to texture in Section 2.4. There are many Gabor filters used for image segmentation. This section concentrates on a few which appear to be well designed, are well cited or are generally superior.

[20] The Gaussian is taken over an unordered neighbourhood in 3 dimensions to account for colour. This model, which is introduced as a Gaussian Markov random field, bears no resemblance to the standard MRF models presented in Section 2.4.7.

One underlying hypothesis to texture segmentation suggests that "the perception of texture is due to 'regularity' in local perceptual properties over an image region" and in general has no intrinsic local meaning [106]. This appears to be the case for Gabor filters as suggested by Figure 2.40(b) and 2.40(c).

Consequently Perry [106] suggests the need to perform phase merging or image blurring on the response from Gabor filters. Filters are extracted at 3 scales and 8 orientations. Using the mean and standard deviation of all filter responses in each local neighbourhood as initial features, two final feature vectors are formed. These consider the scale at which the filters provide a maximum response, the associated orientation and a measure referred to as "*significant peak measure (SPM)*". This SPM has the form $SPM = (x - \mu_{\overline{x}})/\sigma_{\overline{x}}$, where x is the filter response with the largest absolute value and $\mu_{\overline{x}}, \sigma_{\overline{x}}$ are calculated over all other filters. The idea is that this feature vector indicates the strength of the local texture at its dominant scale (period) and orientation.

Using these features a centroid linkage type region growing algorithm is applied. Seed regions are chosen at locations with 4 *close* neighbours, with the addition of features within a threshold of the region's mean value which is recalculated each step. A final region merge method assigns small loose regions to the appropriate neighbour or labels them as untextured.

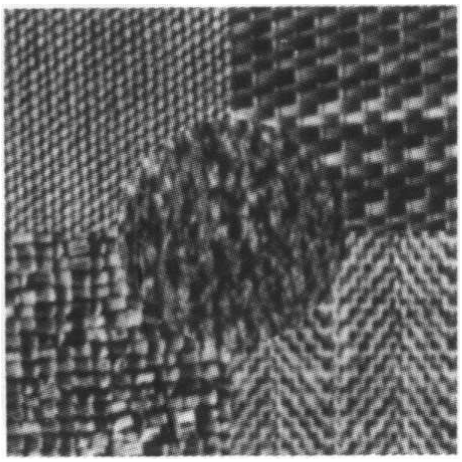
Overall the algorithm design incorporates many human-specific assumptions about the Gabor filters and what is important in order to distinguish textures. Although the results are at first impressive, illustrations are presented on selected Brodatz textures with well defined orientations. In general the reduction from 48 features to just a few is not well justified and furthermore sensitive to noise or subtle variations present in real-world data. The approach is unsupervised (although requires predefined thresholds) and suited to well behaved artificial textures.

Jain [140] [105] [83] proposes an alternative more general algorithm for unsupervised texture segmentation. The method involves the following steps.

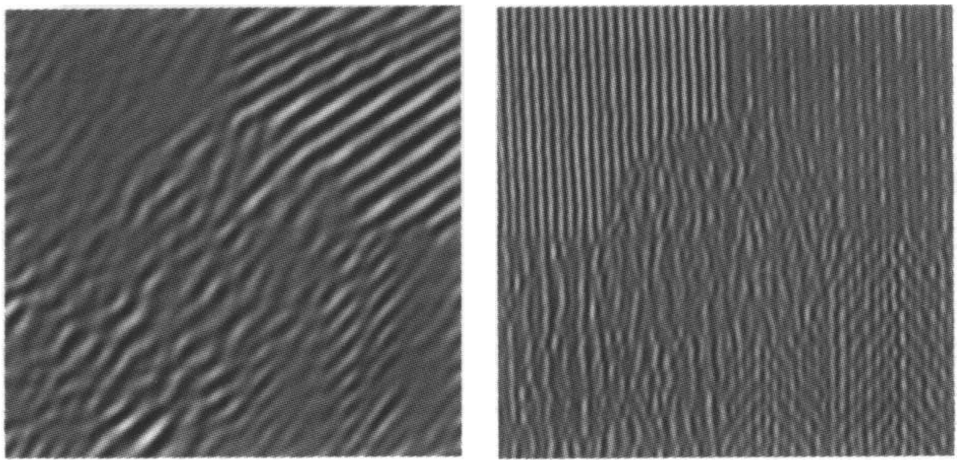
- Use a bank of real symmetric Gabor filters at 4 orientations and octave spaced scales designed with half-peak magnitude bandwidths (see Sections 2.3.3 and 2.4.6). This corresponds to octave spaced scales of all available resolutions up to $\pi/\sqrt{2}$. For a 256×256 image this corresponds to 28 filters at scales $1\sqrt{2}, 2\sqrt{2}, 4\sqrt{2}, \ldots 64\sqrt{2}$ cycles per image width.

- Reduce the number of filters by examining the power spectrum of the image being processed (see Figure 2.31, page 63). Filters are chosen to encompass 95% of the intensity variation in the image.

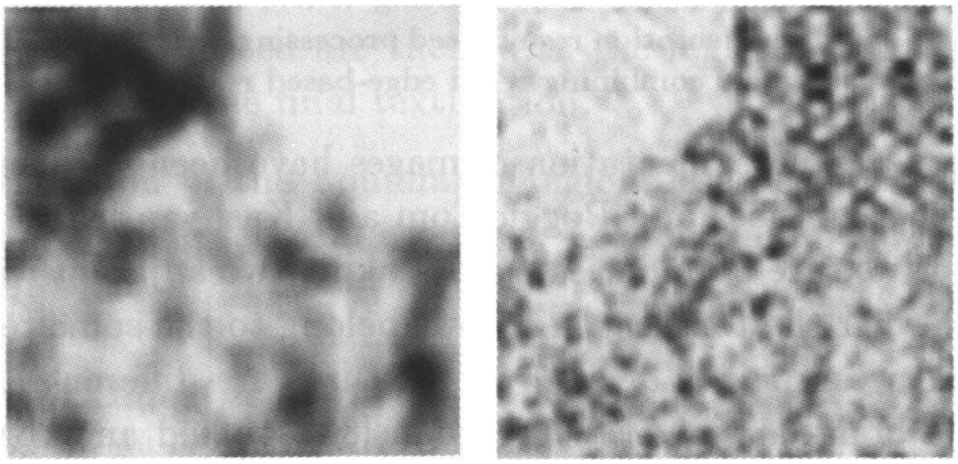
- Process dominant filter outputs $r(x, y)$ with the nonlinear function $\phi(x)$ shown below. The justification is that this "nonlinearity bears similarity to the sigmoidal activation



(a) original image



(b) Gabor filter responses



(c) responses from (b) after post-processing

FIGURE 2.40: Gabor filter response on a textured image with nonlinear post-processing. [140] [83]

function used in artificial neural networks." A value of $\alpha = 0.25$ is used.

$$\phi(x) = \tanh(\alpha x) = \frac{1 + e^{-2\alpha x}}{1 - e^{-2\alpha x}}$$

- Extract features by taking the average absolute windowed response of $\phi(r(x, y))$ using a Gaussian filter with a standard deviation proportional to the period of the associated Gabor filter. An example of these features is shown in Figure 2.40 (d) and (e).

- Use the CLUSTER algorithm on a randomly selected 6% of the data to reduce computational overheads.

The above techniques [140] produce subjectively acceptable results on Brodatz textures and the triangle-arrow and S-IO textures used in psychophysical studies of texture perception (Figure 2.28, page 57). However, while the algorithm is reasonable for images contain a few textures, the results decline as the number and variation of textures within the same image increases. This is mainly a result of the clustering algorithm which dynamically chooses the number of textures and lacks spatial representation or post feature neighbourhood considerations during clustering. It also ignores phase information. This algorithm does out perform other cluster based algorithms which use Gabor filters [106] [199] [88].

A more recent and less standard approach by du Buf [120] [121] suggests that textures have local N-folded symmetry, treating the problem as an abstract line detection problem as depicted by Figure 2.41. The procedure involves computing the power spectrum using 30 filters (5 octave spaced frequencies and 6 orientations), taking complex moments, reducing the number of features using the Karhunen-Loeve transformation (principal components analysis) and performing a quad-tree based split merge algorithm. The algorithm is reasonably simple, but effective and well suited to textures with even fold symmetries due to the properties of the complex moments [121]. Hence, it effectively discriminated textures with linear, rectangular and hexagonal geometric primitives but is less effective on random fields.

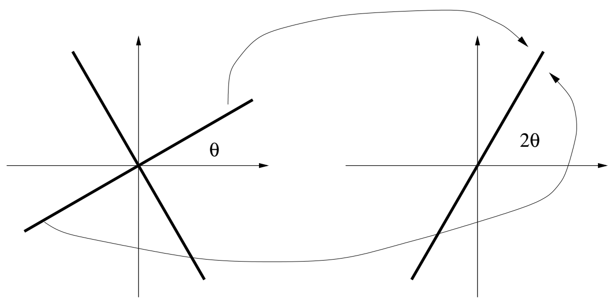


FIGURE 2.41: An illustration of the cross-fitting problem by performing line fitting in a transformed co-ordinate system. [121]

Significant efforts have been made by Dunn [139] [133] [134] [95] to design the optimal Gabor filter to segment images contains two predefined textures. More details on the methods are outlined below. Bovik, et al. [86] also examines this problem by adopting a simple peak finding scheme to the global power spectrum to select filters. Both of these, along with methods by du Buf and Jain are discussed in a review of texture segmentation techniques [142].

In general, Gabor filters provide useful features for texture segmentation, although some claim that other similar multichannel filters are superior for texture segmentation [101]. However, for practical applications, either the textures need to be known *a priori* or a large number of filters with post-processing is required. Hence it is unclear how effective Gabor functions are for segmenting real-world images in which perceived textures are not well defined, irregular and not homogeneous within meaningful regions.

Dunn's optimal filter design

Dunn's PhD research is well summarised by two papers [95] [134][21].

The first [95] examines the theoretical aspects of filter responses considering the effects of *texel spacing* on these responses. It is suggested that different filters result in four different distinct signature types, namely a step, valley, ridge and step in local output variation. With consideration for these it is suggested in [134] that segmentation can be performed by locating discontinuities in filter output given suitable Gabor filter parameters.

Rather than using traditional ad hoc methods, a specific filter is designed to optimally distinguish between two textures for which prototype samples are given *a priori*. It is further suggested that filter output for each prototype form Rician random variables which are "often approximated well as Gaussian rv's" [134].

Given these findings the following algorithm identifies the *optimal Gabor filter* for distinguishing between two pre-known textures.

1. "Select a value for σ, based on the structure of the textures" [134]. This values specifies the bandwidth or spatial spread of the filter. There is a tradeoff between sensitivity to local variation and spatial precision as implied in [134] which suggests that "a good compromise is to choose σ approximately equal to the texel spacing". A further suggestion made is to run the algorithm over a range of values and compare results.

2. Apply several candidate Gabor filters of different orientations, frequencies and bandwidths to randomly selected points within a texture. In [134] the orientation and frequency are considered by Cartesian coordinates in frequency (U, V) and the bandwidth as above.

[21] A written request for the actual thesis was replied with "These 2 papers cover the bulk of my thesis." Although differing in layout these papers contain virtually the identical information to articles [95] and [134]

3. For each (U, V) model each texture by a sample mean and standard deviation. Using the model for each texture calculate the minimum probability of classification error for this filter.

4. Rank these errors to indicate the "most effective Gabor filter for segmenting images containing textures A and B" [134].

In summary, this algorithm produces good results for a wide variety of textures [133] [134] and is applicable for additional images presented in [95]. Unfortunately it is quite computationally expensive, leading to an alternative sub-optimal method which examines the local power spectrum of an image [139]. This basically determines the minimum Gaussian smoothed ratio between the local power spectrum of two images, hence providing (U, V, σ) according to the frequency location and smoothing deviation. Raw results are reasonable and much improved by heavily smoothing the filter output. However this additionally assumes well defined spatially disjoint texture regions at a macro scale and is not applicable for real scenes.

2.5.6 Markov random fields

These fall into the category of "stochastic models and decision-theoretic techniques" reviewed in [142]. Several approaches to supervised maximum *a posterior* probability (MAP) segmentation using MRFs are based on the methodology introduced by Geman and Geman [159]. These use the MRF model outlined in Section 2.4.7 exploiting the equivalence between modelling the total energy of a lattice and specifying clique interactions in terms of conditional probabilities.

A hierarchical model proposed by Derin and Elliot [158] uses a high level Gibbs distribution to model image regions and a low level one to model textural properties within regions. Segmentation using the MAP criterion with a second order neighbourhood system η^2 is successful although requires high level distribution parameters to be known. This implementation is performed on strips of the image to reduce the computation, providing a suboptimal solution using a few minutes of CPU time (1987). The algorithm is nicely presented in [200]. Similar algorithms using dynamic programming are applied by Farag [201] [202].

A stochastic relaxation algorithm is usually used to perform MAP estimation, as informally described in [159]. This involves making a change based on the local neighbourhood. Changes are performed *randomly* from a local conditional probability distribution dependent on a global *temperature* parameter T. A value $T = 0$ corresponds to a greedy algorithm such as gradient descent which is bound to find a local minimum, whereas for $T \to \infty$ purely random changes are made. The stochastic relaxation operates by slowly reducing the temperature parameter.

Simulated annealing (SA) [203] [204] [205] is a commonly used stochastic relaxation algorithm applicable for NP-complete problems such as MAP estimation. It can be proved that SA converges to a global maximum for a stationary distribution [206] [203] although the required parameter choice is not usually computational viable. Simple hardware implementations of annealing algorithms may offer a computationally viable alternative [207]. Simulated annealing is used for MRF based modelling and segmentation using MAP in [159] [157].

Estimation using iterative conditional modes (ICM) which is a deterministic relaxation algorithm offers an alternative computation efficient optimisation, although it is prone to being trapped at a local minimum [208]. This method involves iteratively and deterministically changing pixel classifications to improve the probability of the segmentation field. Local clique distributions and distributions across the current image segmentation are typically used along with measurements between these histograms. Such a MAP formulation using ICM for segmentation is presented in [209].

Bouman [208] [210] suggests a multiresolution segmentation algorithm to be computationally viable and "less likely to be trapped in local minima than ICM." A pyramid decomposition is used to perform top-down MRF segmentation at gradually increasing resolutions. Classification results on synthetic images indicate performance is superior to ICM and comparable to SA but at a fraction of the computational cost [208].

All of the above image segmentation methods use a Bayesian approach and a Gibbs distribution. While MAP estimation is most common, other formulations are used, such as maximum *a posteriori* marginal (MPM) estimation which theoretically minimises the expected classification error [211] [212]. These Bayesian formulations are very attractive since they allow assumptions "to be explicitly stated in the probability distributions" [142]. An overview [142] suggests these formulations can be solved using statistical approaches such as simulated annealing or deterministic approaches categorised as "gradient descent and annealing", "path in parameter space" and "EM algorithm". A common special case of "path in parameter space" incorporates the Akaike information criterion or an MDL component (see page 31) into the energy function. This is done in [209] and [208] to allow ICM to be used without specifying the number of image segments.

Other examples and variations of MRF systems include using a top-down quad-tree based split-merge type algorithm [161], a hierarchical pyramid based segmentation with results suitable for compression applications [213] and considering relationships between MRFs and co-occurrence matrices [160]. Specific algorithms and equations have been omitted from this section, appearing within the citations quoted. For an introduction but practical overview to MRF models refer to [155] [211] and [200].

Final comments

In general, MRF based segmentations tend to be based on pre-specified parameters or a predefined number of regions. While results are often impressive on synthetic images, application to real-world scenes is rare with results subjective and less than impressive. Furthermore, while these models are theoretically very powerful, practical implementations use only first or second order neighbourhoods. This restricts the complexity or structure of the describable local characteristics as well as making these methods heavily dependent on image resolution at the lowest level. Increasing the complexity and accounting for image resolution is not considered since it is generally computational infeasible.

MRF models and Gabor filter based segmentations have remained separate fields with few references between them. Although comparisons exist they are heavily dependent on both image data and on implementation details.

2.5.7 Mixed-mode segmentation

Mixed-mode segmentation is a subset of document image analysis involving the segmentation or identification of regions corresponding to text, background, images, and so forth. Numerous techniques exist, although most make specific assumptions to take advantage of text document properties. For example, by assuming the rectangular layout of printed text and the organisation of text lines, morphological operations, projection based histogram analysis, connected components analysis and the Hough transform offer applicable tools [83]. More general techniques tend to use spatial filters assuming only that text regions have high frequency content or heavy local intensity variation.

Tang [214] [215] takes the less general view, suggesting that "knowing the structure of a document is the key to successful processing of a document" [214]. Hence, the major analysis techniques considered are the "Hough transform, projection profile cuts, run-length smoothing algorithms, connected components analysis, crossing counts and form description language" [215]. The Hough transform provides a means for skew detection, text block identification and other groupings. Skew detection involves determining the relative orientation of a text document and rotating it to a square upright position. It is typically the first form of processing, assuming a text content with performance suffering from non-text information such as embedded pictures. Text block identification assumes blocks of text have regular uniform widths and would seem to benefit by some sort of morphological smoothing prior to applying the Hough transform. Projection profiles of pixel values offer a more common text block identification tool. Locating valleys in these histograms corresponding to various orientations provides a mechanism for isolate text blocks at various scales and is also useful for skew normalisation. Refer to [215] for further details.

Baird [216] examines the problem of locating rectangular regions of text on a white background. The algorithm is claimed to be "generic and nearly parameter free", "relatively insensitive to line orientation" (for example Chinese text is vertical) and "oblivious to page orientation", although these claims are somewhat misleading. The algorithm repeatedly identifies the *best* rectangular white regions according to a measure $K(s) = \sqrt{area(s)W(alar(s))}$, where *alar* is short for the "absolute value of the base-2 logarithm of aspect ratio" [216] and the heuristic weighting function illustrated in Figure 2.42 penalises both square and extremely thin regions.

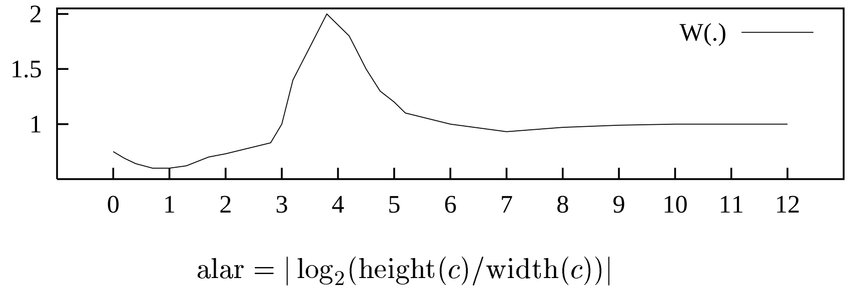


FIGURE 2.42: Heuristic dimensionless weight and *alar* functions used to provide the strength of a rectangular region in [216].

The idea is that the *alar* function prefers regions that are moderately long and thin. This heuristic is chosen over others (see [216]) with the segmentation ceasing when $K(s_j) - 42.43F(s_j) \leq 34.29$ for step j where $F(s_j)$ is the fraction of the image covered to this step. A more general method is to segment or cluster the $F(s) - K(s)$ plane.

Results are reasonable as could be expected on well oriented journal papers containing only text, although the heuristic stopping rules method often fails. Overall, the method is claimed to be generic and parameter-free since it does not require a prior layout model or a defined text size. However, the algorithm does require strict orientation (such as that attainable by skew analysis [217]), although is oblivious to 90 degree rotations.

Unsupervised clustering and supervised classification of Gabor filter features provides a more general means for segmenting text documents into regions corresponding to text and non-text. Figure 2.43 shows the result of 3 cluster segmentation using filters with 4 orientations and the 2 highest radial frequency scales [218]. These classes loosely correspond to text, uniform regions and boundaries or transitions. It is inferred that the transition region is required because Gabor filter features do not form distinct clusters.

(a) original (b) segmentation (c) text regions

FIGURE 2.43: Segmentation of a newspaper image using clustering of Gabor filters. From [83].

The supervised method assumes correct orientation and hence only uses two filter orientations. A single-layer neural network is trained on different newspaper images and produces results similar to Figure 2.43(c). Furthermore, clustering is shown superior for bar code localisation using this classifier. The clustering technique takes "about one minute of CPU time on a SPARC-Station 2, for a 512×512 image" [218].

It is suggested [218] that this proposed method is rotationally invariant, robust to variations in font size and does not require thresholds. However, it is clearly limited to font sizes with spatial frequency characteristics that are adequately described by the lowest frequency Gabor filters. Also, both text and background regions are needed to form the appropriate clusters.

Mixed-mode segmentation is also useful for compression applications.

> "For efficient storage or transmission over a band-limited channel, a viable compression algorithm should attempt to maximise compression with minimal loss in visual quality. Since text and image data have different properties, proper segmentation of the data would help in the process of compression." [219]

While popular algorithms such as *jpeg* provide high lossy compression rates, they do not perform well on mixed-mode images which have a substantial text component.

In [219] blocks of an image considered with windows measurements for absolute intensity deviation, standard deviation and Sobel edge strength are compared to DCT coefficients. Histograms illustrate the ineffectiveness of each single measurement, with results illustrated for the "best manual adjustment of threshold" on 5 test images. These images consist of hand-written text, chalk on a board, typed text with a photograph, text with a hand from an overhead projector, and a scanned magazine. A comparative evaluation of robustness uses thresholds on the *same images*. The DCT coefficients are deemed superior for classification and robustness although comparisons do not seem sound with obvious errors in the result tables and seemingly biased results.

Segmentation for compression of workspace screens is considered in [220]. Text regions are ideally compressed with lossless techniques (e.g. *gif*) or recognised using OCR, and other regions with lossy ones (e.g. *jpeg*). Two main algorithms are presented. A top-down region splitting algorithm uses a measure of region entropy using adjacent pixel *colours* similar to that used in co-occurrence matrices (see Section 2.4.4). A region growing algorithm takes advantage of the rectangular image structure using similar techniques. Connected components analysis is also considered. Results are reasonable although very application specific.

In summary, there is an abundance of document image tools based on specific assumptions. More general techniques suggest local simple measurements are not sufficient to distinguish text from non-text regions effectively, with Gabor filter features emerging the most effective general measure. All of these do however assume the existence of text with high frequency or fine scale properties.

2.5.8 Summary

In general there is no clear mechanism for segmenting images, as indicated by several surveys and evaluations of image segmentation [142] [166] [148] [221] [222]. In each case certain assumptions are made about the type or content of the images. Considering all techniques outlined in this section, region growing and split-merge algorithms appear superior to thresholding, edge detection or clustering. However such algorithms, whether they be based on random fields, energy models, top-down or bottom-up processing, require *appropriate* image features and mechanisms.

2.6 Colour image representation and analysis

This section examines colour representations from an image processing point of view. The more typically used RGB and HSI colour spaces are examined as well as some discussion on human perception and alternative colour schemes. Examples of how colour is used for image analysis are only briefly outlined, since most treatments use the methods already discussed in this chapter. Colour is only considered for the intelligent analysis of images and not for displaying, copying, reproducing or photo correction [223].

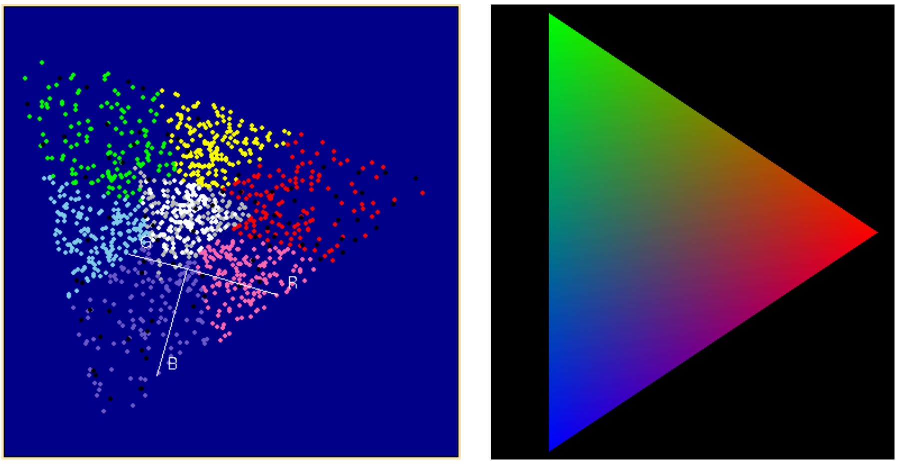
2.6.1 Red, green and blue (RGB) and normalised RGB

The essential rules of colourimetry established by Grassman in 1853 are basically (1) that any colour can be created by a unique combination of three other colours, (2) scaling the components equally produces the same colour and (3) the luminance of a mixture of colours is equal to sum of the luminances of each colour [224]. Since then, through experimental and theoretical results and based on the structure of the eye, all colours are observed as "variable combinations of three so-called *primary colours* red (R), green (G), and blue (B)" [41]. This is also referred to as the *trichromatic law* [225] [81]. Consequently, colour analysis compatible with the retina and human perception only requires three dimensions, or two if intensity or illumination is treated separately. A thorough overview of biological vision is presented in [82, p41–95].

For convenience, the *Commission Internationale de l'Eclairage* (CIE) standardised these primary colours by the wavelengths $\lambda_R = 700$nm, $\lambda_G = 346.1$nm and $\lambda_B = 435.8$nm [41]. These form a basis for computer displays corresponding to emission phosphors for mixing red, green and blue components. As a result, the RGB colour representation has become the standard method for storing digital images. Each coloured pixel is represented by the values $R \in [0, 1]$, $G \in [0, 1]$ and $B \in [0, 1]$, hence forming a colour space which is a cube of unit length. Generally an 8 bit representation is sufficient for each component and hence values are often considered to have the range $[0, 255]$.

The intensity of a coloured pixel may be considered to be proportional to the sum of the three components, i.e. $i = (R + G + B)/3$. Hence intensity normalisation is achieved by dividing each RGB component by the intensity, i.e. $r = R/i$, $g = G/i$ and $b = B/i$.

The rgb space forms a triangle embedded in three dimensional space through the points $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$ as illustrated in Figure 2.44(a). The resulting space is, of course, two dimensional.



(a) normalised RGB colour (b) *triangle colour* space

FIGURE 2.44: A projection of normalised RGB colour and the *triangle colour* space for intensity=1.0. Colours in (a) provide a rough estimate only.

One possibility is to represent colour using Cartesian coordinates in this plane. In [226] this is referred to as *triangle colour*. The two values tc_1 and tc_2 are chosen in the red and blue-green directions and scaled such that the triangle sides have unit length, i.e. $tc_1 = (2R - (G + B))\sqrt{3}/12i$ and $tc_2 = (G - B)/6i$. The space is shown for $i = 1.0$ in Figure 2.44(b). An alternative is to simply use normalised red and normalised green [227] although these are non-orthogonal. One of the main drawbacks of RGB normalisation representations is intolerance to noise especially at low intensities. For example, the colour $[0, 0, 0.01]$ is located on the extremity of the space and considered fully blue even though it is perceived as being black.

An alternative is to use the Karhunen-Loeve transformation (KLT) to obtain an RGB colour basis with decorrelated and normalised axes. This is studied by Ohta [228] using 8 different images. From results it is concluded that the principal eigenvectors in descending order are approximately $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, $(\frac{1}{2}, 0, -\frac{1}{2})$ or $(-\frac{1}{2}, 0, \frac{1}{2})$, and $(-\frac{1}{4}, \frac{1}{2}, -\frac{1}{4})$ and hence colour can be adequately described using three orthogonal colour features $ohta_1 = (R + G + B)/3$, $ohta_2 = (R - B)/2$ and $ohta_3 = (2G - R - B)/4$. These are known as *Ohta's colour features*. Although this formulation seems sound, the features are equivalent to normalised colour and the inferences made by Ohta are not necessarily justified as illustrated in Table 2.6.

Another study [229] examines the measure corresponding to the dominant eigenvector from the KLT, suggesting that "the direction of the eigenvectors in Table 2.7 are close to an intensity axis, since the intensity axis Y in YIQ is" $Y = 0.299R + 0.587G + 0.114B$. Once again this inference is subjective and heavily dependent on the images chosen. Ideally

Image	the eigenvector normalised by $G > 0$, $\lvert R\rvert + \lvert G\rvert + \lvert B\rvert = 1$								
	R_1	G_1	B_1	R_2	G_2	B_2	R_1	G_2	B_3
cylinder	0.269	0.363	0.367	0.469	0.095	-0.437	-0.308	0.461	-0.231
building	0.269	0.340	0.391	0.479	0.103	-0.418	-0.296	0.485	-0.219
seaside	0.258	0.380	0.362	-0.585	0.056	0.358	-0.176	0.464	-0.360
girl	0.336	0.354	0.309	-0.493	0.193	0.314	-0.094	0.474	-0.436
room	0.193	0.341	0.467	0.612	0.079	-0.310	-0.209	0.507	-0.284
home	0.197	0.328	0.476	0.492	0.180	-0.328	-0.313	0.484	-0.204
auto	0.304	0.317	0.378	0.239	0.309	-0.452	-0.514	0.450	0.036
face	0.175	0.411	0.414	0.523	0.128	-0.349	-0.295	0.416	-0.289
relationship	$(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$			$\pm(\frac{1}{2}, 0, -\frac{1}{2})$			$(-\frac{1}{4}, \frac{1}{2}, -\frac{1}{4})$		
feature	$(R+G+B)/3$			$(R-B)/2$			$(-R+2G-B)/4$		

TABLE 2.6: Eigenvalues from applying the KLT to the RGB space of 8 different images. Based on data from [228, p15,p20].

Image	the eigenvector for λ_1		
	R	G	B
Hada	0.547	0.571	0.612
House	0.276	0.605	0.747
Office	0.375	0.557	0.741
Peppers	0.218	0.855	0.470

TABLE 2.7: Principal eigenvector using the KLT on the RGB space with 4 different images, from [229].

several thousand images should be used rather than 4 or 8. Using Y to represent the image is also known as the the *whitening transformation* [229].

Unfortunately all RGB and normalised RGB representations and linear transformations suffer from problems. They are not transferable, producing different colours across different devices; are not perceptually uniform; are not intuitively meaningful and are incomplete with some real colours lying outside the RGB space [230] [231]. Hence other colour representations have been derived.

2.6.2 Hue, saturation and intensity: HSI Colour cones and cylinders

HSI offers a more intuitively meaningful colour representation using normalised RGB values (r,g and b). Hue defines pure colour (e.g. red, yellow or orange) and saturation is a measure of how much this colour is diluted by white light [41]. These equations define hue as an angle, saturation between 0 (grey) and 1 (full colour) and intensity for completeness.

$$\begin{aligned}
\text{hue} = \quad & h = \begin{cases} \theta & \text{if } b > g \\ 2\pi - \theta & \text{otherwise} \end{cases} \\
& \theta = \arccos \frac{2r - g - b}{\sqrt{6}[(r - \frac{1}{3})^2 + (g - \frac{1}{3})^2 + (b - \frac{1}{3})^2 +]^{\frac{1}{2}}} \\
\text{saturation} = \quad & s = 1 - \min\{r, g, b\} \\
\text{intensity} = \quad & i = (R + G + B)/3
\end{aligned}$$

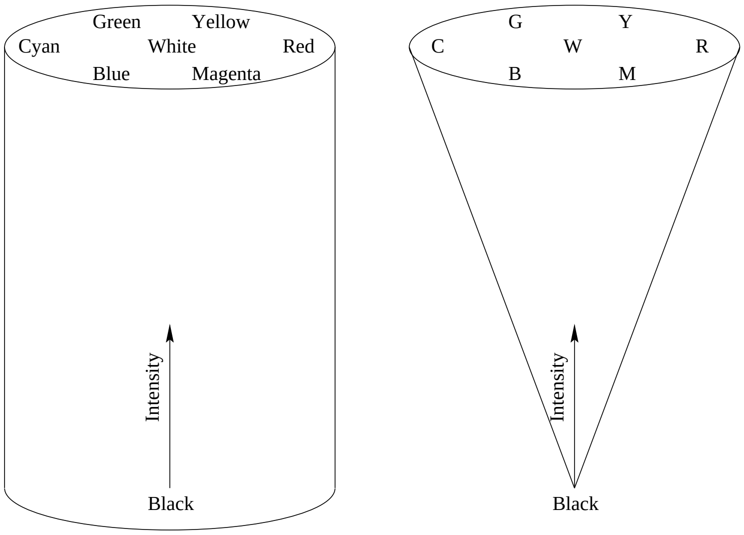
The hue-saturation (H-S) plane forms the colour wheel commonly used for selecting colours. In relation to the colour triangle in Figure 2.44(b), the hue is the angle from the triangle's centre and the saturation the ratio between the centre and edge of the triangle. Hence, the H-S plane is very similar to the triangle colour space, although it favours all saturated colours evenly rather than being biased towards the primary colours.

Although the HSI space is typically presented as a conical space, the corresponding standard equations actually form a cylindrical space. This attention to detail is overlooked in [41] [231] and [40]. As a result, the values of h and s are less meaningful at low intensities and sensitive to noise in the same way as other normalised RGB measurements. In general, "Nonlinear transformations used to obtain color features such as hue, saturation, and normalized color have non-removable singularities, near which a small perturbation of the input R, G and B can cause large jumps in the transformed values" [228, p11]. An alternative transformation is to limit the saturation proportional to intensity, hence producing the colour cone. These spaces are depicted in Figure 2.45. The colour cylinder benefits from having independent intensity and colour measurements, although it produces meaningless values for very dark colours. In contrast, the colour cone restricts the saturation at low intensities but also distinguishes between the *same colour* for medium to high range intensities.

A further concern is the angular hue value, making colour averaging difficult in this form. This is easily overcome by specifying the H-S plane using Cartesian coordinates, although this is often overlooked.

2.6.3 Perception initiated schemes

Other colour representations are based on human perception and adequate signal encoding schemes. Over 100 years ago German psychologist Ewald Hering posed a theory of six elementary colours from which all perceived colours can be formed [230]. He proposed three opponent channels, namely black-white, red-green and blue-yellow, which are intuitively sensible since reddish-green and bluish-yellow colours are not perceived. Since this time, electro-physiological readings from the ganglion cells in the retina have reinforced this theory and also suggests that the spatial sensitivity of the blue-yellow channel is much



(a) HSI colour cylinder (b) HSI colour cone

FIGURE 2.45: The two standard HSI colour spaces. These illustrations are not applicable for high intensities where the saturation is also limited.

lower than the red-green one. Consequently, contrast in the blue-yellow channel is not easily perceived, e.g. yellow text on a white background or blue text on a black background.

The natural colour system (NCS) accounts for this, using hue defined in four quadrants with opposing red-green and blue-yellow primitives. A triangular grid defines the remaining two dimensions with vertices for colour, whiteness and blackness, and colours based on whiteness/chromatic intensity, blackness/chromatic intensity and blackness/whiteness (see Figure 2.46). The NCS is used by *Dulux* for specifying paint colours and is gaining international acceptance as the best standard for specifying colour [230]. It is also known as the *Ostwald system* after Nobelist Wilhelm Ostwald [225].

The Munsell system illustrated in Figure 2.47 provides a three dimensional model for

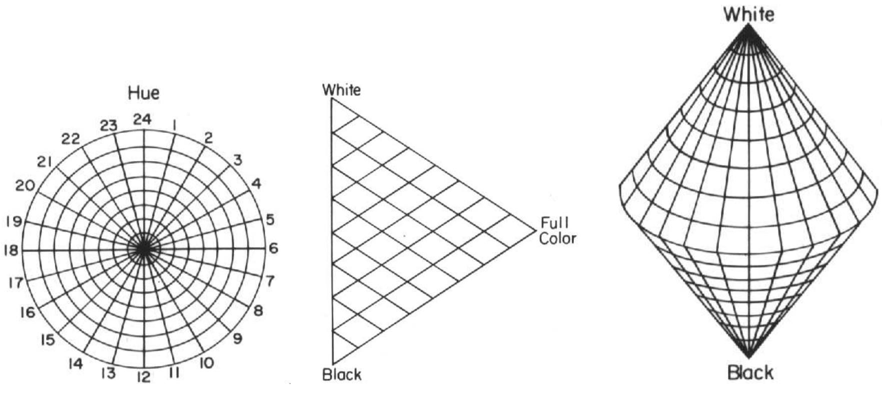


FIGURE 2.46: Natural colour system (NCS) or Ostwald system for representing and ordering colours. [225, p162]

perceptually uniform colour. Five hues (red, yellow, green, blue and purple) are used at various intensities. The important feature of the scheme is that the restriction of colour *saturation* varies both across intensity ranges and for different hues. For example, heavily saturated yellows only exist at high intensities and blues at lower intensities. Although the system models perception well, conversions to conventional colour spaces such as RGB can only be done using tables [231].

FIGURE 2.47: Arrangement of the Munsell colour system, from [225, p173].

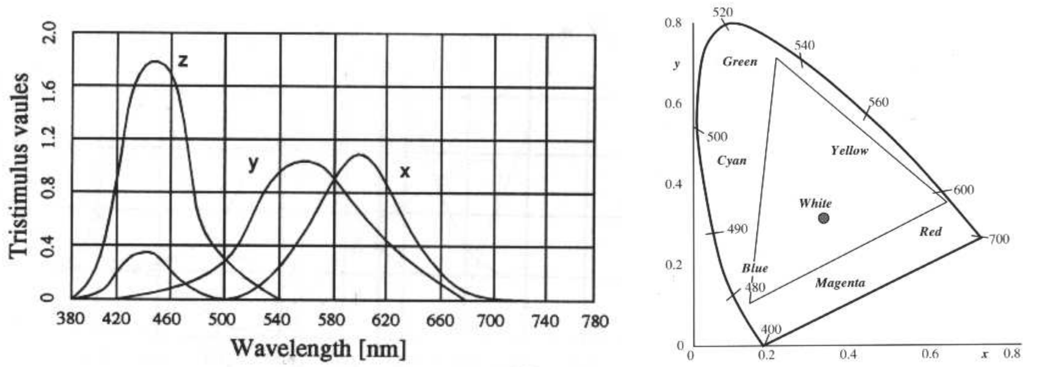
The YIQ (also called YUV) colour space also separates illumination from colour and is commonly used for the NTSC broadcasting scheme since a mathematically defined transformation to RGB exists and the separate luminance provides upward compatibility with black and white televisions. The luminance Y which is defined as being "proportional to the amount of light perceived by the eye" [41] varies from the traditional intensity measure. It considers green to provide more perceived light than blue which is consistent with the Munsell system. The I and Q colour components, which may be negative, roughly align with red-cyan and green-magenta colours respectively. The transformation [23] from RGB space is linear, one to one and reversible.

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.6.4 The CIE system of colourimetry

A *colourimeter* is an instrument which synthesises an equivalent stimulus to measure colour. The tri-stimulus values obtained with a properly designed colourimeter provide colour measurements based on an observer who determines the equivalent stimulus. Consequently values vary between two observers, even though neither has demonstrably anomalous colour vision [225].

The CIE recommended the concept of a standard observer representative of humans with normal colour vision and through experimentation proposed an international standard for the tri-stimulus value. These values, labelled X, Y and Z, are often normalised such that $x = X/(X+Y+Z)$ and $y = Y/(X+Y+Z)$. The x and y values form the CIE chromaticity diagram shown in Figure 2.48. As outlined in [82] these new non-physical primaries exist only in a mathematical sense.

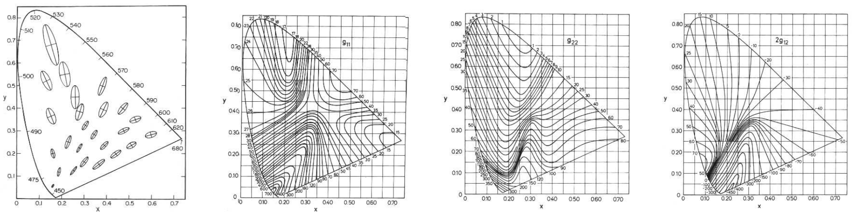


(a) Tri-stimulus values for standard observer. (b) CIE chromaticity diagram

FIGURE 2.48: The international standard colour specification system proposed by the CIE in 1931. [82, p66] and [23, p39]

The advantages of the system are that the coordinates of real colours are always positive which is not the case for RGB colours, the luminance can be separated, and white is defined by $X = Y = Z$ [231]. The major disadvantage is the lack of perceptual uniformity as illustrated in Figure 2.49. Even though the system is based on human perception, the perceived difference between two colours does not correspond to the Euclidean distance between the points. Figure 2.49(a) shows the perceived standard deviation at a few selected sites. The gradients of the corresponding variance and covariance of these are shown in 2.49(b)-(d) indicating highly irregular fields of perceptual uniformity.

To allow for perceptually meaningful colour comparison in terms of a measured difference, the CIE proposed the $CIE - L^*u^*v^*$ and $CIE - L^*a^*b^*$ colour systems. These are based on uniform sampling of the visible colour gamut and allow the Euclidean distance to determine the visible difference between two colours. The transformation from RGB colour space is highly nonlinear and "the reverse transforms are not straight forward or do not exist in closed form" [232]. Furthermore, these require a predefined white reference.



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

FIGURE 2.49: The variation of perceptual deviation in the standard CIE chromaticity diagram, from [225, 131–133].

The transformation from $CIE - XYZ$ to $CIE - L^*u^*v^*$ space is presented in [225] with $CIE - L^*a^*b^*$ defined similarly [231].

$$L^* = 116(Y/Y_0)^{1/3} - 16 \qquad u^* = 13L * (u' - u'_0) \qquad v^* = 13L * (v' - v'_0)$$

$$\text{where} \;\; u' = 4X/(X + 15Y + 3Z) \quad v' = 9Y + 15Y + 3Z$$

In this relationship u'_0 and v'_0 correspond to the white reference based on X_0, Y_0 and Z_0. "The constants X_0,Y_0 and Z_0 are the tri-stimulus values of the material adopted as the standard white reference for each application" [225, p219].

Assuming a white reference of $R = G = B = 1.0$, the relationship between RGB and XYZ is [82].

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 2.77 & 1.75 & 1.13 \\ 1.00 & 4.59 & 0.06 \\ 0.00 & 0.06 & 5.59 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

2.6.5 Example of colour image analysis

The techniques presented earlier in this chapter are often used on colour images. While many systems are general, treating colour as an additional data dimensions, some systems use specific relationships based on properties or assumed distributions in the chosen colour space. Others compare different colour spaces or justify one colour representation over another. All of these considerations are addressed in this section. Colour histograms, often used for content based image retrieval, are not discussed.

RGB

It is common to treat the separate RGB components and use pre-known properties of these with respect to the current applications. For example, "To separate the last two cell structures from each other we need only the blue component of RGB images where only the pixels belonging to red cells have bright grey levels" [21]. A more complete specific system [233] calculates the difference between planes ($R - B$, $G - B$ and $R - G$) using numerous histograms and thresholds of these to perform segmentation, such as the mean $B - G$ difference in a local neighbourhood. Others cluster data points assuming certain distributions and characteristics within two or more of these colour values. Clustering in the $R - B$ plane is quite successful for road segmentation for autonomous robot navigation [18]. All of these examples are very application specific, although they demonstrate the usefulness of simple RGB measurements.

In [234] a region colour segmentation algorithm works with raw RGB values. Seed regions are formed based on the gradient given by $grad(p) = |grad_r(p)| + |grad_g(p)| + |grad_b(p)|$ where $grad$ is a simple morphological gradient measure. Regions are grown while neighbouring pixels are below a user-defined threshold, after which unlabelled pixels are assigned according to the distance from region clusters or assigned new labels where appropriate. The algorithm is simple, but tailor-made with good results illustrated on an artificial image which possibly suits the algorithm.

The above combination of edge gradients is not necessarily the best. In [235] the application of the Canny edge detector to RGB planes is discussed. By considering a range of Minkowsky metrics on the resulting three values, the limiting metric, or maximum value, is suggested to work well as a colour edge detector. The implementation also considers connected edges and elimination of non-edges through non-maximum suppression. Despite these considerations Chang [236] effectively uses a simple intensity gradient for edge detection in colour images. Further discussion on colour edge detection appears in [231].

The split merge segmentation algorithm from [193] [194] and [195], which is briefly outlined in Section 2.5.4, uses a multivariate Gaussian to model colour in each image region. This model accounts for "not only the spatial interaction within each of the three color planes, but also the interaction between different colour planes." This is good justification for the model which is essentially the same as performing a local KLT over each region. Unfortunately later in the paper(s) the non-diagonal terms of the Gaussians covariance matrix are set to zero to improve efficiency, hence reducing the model to the means and standard deviations of the independent RGB values. This negates the main outlined strength of the model but is overlooked.

A method akin to the KLT projects colours onto the normalised colour plane, hence producing a space similar to triangle colour (Figure 2.44) [190] [191]. In addition colour values are weighted to compensate for low intensities, hence overcoming problems typically associated with normalised colour. Results are not compared with using other representations.

Issues relating to colour histograms and the use of Euclidean distance to measure colour difference are outlined in [48].

HSI

The application of the transformation equations from the standard texts are typically used forming the HSI cylinder space as outlined previously. The alternative HSI cone is considered in [237] with the distinction between mid-range colours of different intensities handled by discretising the space into predefined colour regions.

As with RGB representation it is common for algorithms to use histograms and thresholds of single colour measurements, or distributions over two measurements. For example, hue-saturation thresholding is applied to locate whales from surrounding seas in aerial photographs [11]. This assumes that water is more saturated than the whale, hence handling submerged whales. Hues of both blue and green are considered to account for variations in photographs, with unimodal histograms modeled by a Gaussian PDF. Results are very good, allowing further algorithms to identify the located whale.

Similar specific methods use hue and saturation to segment road from non-road regions and locate red and yellow traffic signs [16]. The assumptions associated with this second example are not justified due to the poorly defined environment. Consequently, they would fail in the presence of yellow flowers or cars and grey buildings. Results are illustrated in the country. A similar system [17] uses the KLT in the saturation-intensity plane to locate road regions.

Although hue and saturation measurements correspond to polar coordinates of the colour wheel, these are often treated as orthogonal Cartesian coordinates. In the worst case [188] both values are scaled between 0 and 1 (i.e. $H = H/360°$) with a Euclidean distance used to measure colour difference. An alternative colour difference considers saturation to be more important for segmentation ($|H - H_s|^2 + |S - S_s|^\alpha + |I - I_s|^2$). Results are presented using $\alpha = 6$ and further improved by limiting the hue weighting. None of the images contain red regions ($H = 0° = 360°$) or regions with dark colours where saturation is highly sensitive to noise.

Even though colour differences are not necessarily directly related to perceived differences, the HSI colour space does provide a representation which separates intensity and has intuitively meaningful components. This is often more appropriate and produces more desirable results than RGB.

Natural colour system (NCS)

The NCS is closely related to the three opposing channels of human perception (see Section 2.6.3). Hence, one possible definition of the opponent colour axis (OPP) is $red - green = R_g = R - G$, $blue - yellow = B_y = 2B - R - G$, and $white - black = W_b = R + G + B$ [238][22]. With this definition, comparisons between Euclidean difference in RGB and OPP colour space yield comparable results to reference colour table methods. The RGB results seem better although neither representation is clearly superior over a variety of images [238]. Exactly the same representation is used in [239] for colour indexing.

An alternative representation used in [240] is defined as $intensity = I = L(G)$, $R_g =$

[22] It is unclear why R_g, B_y and W_b are not scaled by $\frac{1}{2}$, $\frac{1}{4}$ and $\frac{1}{3}$ respectively. It would also appear that W_b is typically much larger than the other values hence making the Euclidean metric inappropriate.

$L(R) - L(G)$ and $B_y = L(B) - (L(G) + L(R))/2$, where $L(x) = 105\log_1 0(x + 1 + n)$ for random noise $n \in [0, 1)$. The green component is used for intensity since "the red and blue channels from some cameras have poor spatial resolution" and the noise offers a novel approach to handle low intensity artifacts. The values R_g and B_y converted to polar co-ordinates are considered hue and saturation. These are used for locating skin in images [240].

These colour schemes are less common than HSI, RGB and CIE ones with fewer comparisons or obvious advantages. In general, it involves decomposing RGB colour in a similar way to normalised triangle colour, except that the colour co-ordinates are non-orthogonal in the RGB space. While this non-orthogonality appears bad, the new colour measurements might be orthogonal with respect to human perception.

CIE representations

A variety of $CIE - L^*a^*b^*$ like representations are used for image segmentation. The TexHVC colour space, a linear transform of $CIE - L^*a^*b^*$, combines advantages from HSI making it both "a user based coordinate system and perceptually uniform" [241]. Both straight region growing and split merge algorithms are presented using TexHVC, with results showing a tradeoff between larger regions and fragmented segments. Neither is particularly impressive and are unfortunately not compared with other colour space representations.

A simpler scheme uses the $CIE - L^*a^*b^*$ colour space, representing the chromatic components (a^*b^*) in polar co-ordinates [175]. Segmentation is achieved using clustering via projected one dimensional histograms of each colour measurement. The histograms are examined for modality (unimodal, bimodal, etc.) with the entire colour space segmented accordingly. The results successfully indicate image regions of similar colour that need not be spatially connected. See [229] for an alternative cluster based segmentation using the same colour system.

Cluster based segmentation comparing a variety of colour systems is presented with respect to recognising resister bands in [242]. Digitally acquired images are used, with pixels belonging to each resister band manually selected and examined to determine the most appropriate colour space. The RGB, HSI, YUV, CIE-XYZ, $CIE - L^*a^*b^*$ and Ohta's colour features are studied to determine the existence and tightness of clusters. Results suggest that although the appropriate CIE space is perceptually uniform in principle and appropriate for tabulated Munsell values, for many digitally acquired images this is not the case. A comparison between scanned Munsell pages confirmed a discrepancy between quoted RGB values and acquired values. Although not mentioned in [242] this is probably based on an inconsistent white balance in either the tables or the digital image acquisition.

Other results show HSI to be inappropriate since "the clusters become huge at low value" [242]. This is the main disadvantage of using the HSI cylinder space as previously mentioned. The nonlinear CIE spaces produce clusters which "become very large in some regions", again probably as the result of a poorly chosen white balance. The overall conclusion suggests that the linear transforms from RGB which separate intensity are best, such as YUV and Ohta's colour features which produce readily separable clusters. Furthermore, for nonlinear spaces colour measurements are very sensitive to the characteristics of the sensor.

In general, it is unclear how well CIE perceptually uniform spaces can be applied to digital images without first knowing or estimating a value for the white reference. Since colour is a psychophysical phenomenon not yet fully understood, methods for effectively determining white balance are based on experimental and theoretical results. In general, perceived colour is based on a combination of local and global variations as depicted by [230, Figure 30–31] and is outside the scope of this thesis. Given an appropriate white reference, the CIE colour spaces are perceptually uniform and useful for chromatic edge detection [224], image indexing and image segmentation as outlined above.

2.6.6 Conclusion

The best choice of colour space for image processing is unclear. Even though perception based systems have been studied in depth no clear conclusions are available. From a mathematical or statistical viewpoint it is desirable to not only measure the distance between colours, but also calculate colour means and deviations in a simple Euclidean space. Unfortunately no *perceptually uniform* method provides this characteristic. Perhaps it is not possible to define an *average colour*. Consequently many content based image retrieval systems use other methods, such as colour distribution or histograms, to provide a measure of, and comparison between, colour regions.

For image analysis RGB and HSI colour schemes are most common. Unfortunately, RGB does not separate intensity or luminance, and normalised RGB or HSI spaces suffer from singularities or a general intensity dependency. Despite these faults the corresponding systems produce acceptable results and appear to be sufficient despite their lack of uniformity with respect to colour perception.

2.7 Content based image retrieval (CBIR)

This section covers CBIR with an emphasis on analysing natural scenes or photographs. Consequently less focus is placed on systems which use specific predefined types of image databases such as the Brodatz textures, databases of one image class, or images containing one well defined object. After a short overview three of the more general systems are outlined. The first is QBIC, one of the best known and pioneering commercial CBIR systems. The second, VisualSEEk, is a more sophisticated complete and available system which originated from work at Columbia University. The third examines techniques established by researchers at UC Berkeley and is chosen due to its generality and focus on real-world scenes.

2.7.1 Overview

> "If a picture is worth a thousand words, imagine the difficulty of cataloguing a database of up to 100,000 images by providing a written description of each one." [244]

This opening typifies research in CBIR in which it is rare to find a publication which does not contain the phase *a picture is worth a thousand words.* As for meaningfully describing images or video sequences so that intelligent queries can perform retrieval based on content, a *written description* is hard to qualify. In [244] it suggests that the cataloguer would have to "describe the color, shape and texture of every element within each picture and its relationship to every other element..." But what is an *element*, how is it represented and how does a regular user formulate an *effective* query?

Traditionally a combination of meta-data and broad image classes are used. "Metadata is the term for the 'external' data used to index and summarize content of the original visual material" [245]. This may include the image acquisition technique, encoding parameters and typically a subjective interpretation in the form of keywords provided by the initial authors or later cataloguer. Ideally the meta-data and broad image class allow for queries such as, *show me all the images containing horses from the image database of country scenes.* While useful for strictly controlled databases and labelling schemes the use of meta-data keywords has numerous disadvantages such as the following.

- Image annotation is manual and tedious.
- Human identified features are very subjective even given strict guidelines. Furthermore they are application specific and not reusable. If requirements change relabelling is very expensive.
- Annotation techniques may vary from one database to another severely degrading performance over a large network of image databases.
- Linguistic barriers make the use of keywords ineffective [48].
- Keywords are not generally unique and do not consider the hierarchy of image structure or even of individual objects. For example [48] a database with annotations for images containing people will fail given a specific query for men or women.
- Appropriate keywords may change over time according to society, language or events. As a simple example a future sporting personality will not be labelled as such.

Despite these problems standardisation of image and video meta-data and taxonomies is heavily studied with the formulation of *MPEG–7* aiming to standardise multimedia content description [245].

From an image analysis view point it is highly desirable to automatically extract features to represent the image content describing meta-data in a general and non-subjective way. The method used by Manjunath and Ma [131] outlined in Section 2.4.6 effectively does this. The system automatically extracts features sufficient for browsing and retrieving images containing a single well defined texture. Unfortunately the techniques cannot easily be extended to general scenes or even those with multiple or non-well defined textures.

Many other retrieval systems exist which are effective for other restricted problems. Techniques well suited to artificial images are applied to retrieve logos and trademarks as presented in [48]. Colour and shape features are used. Three 16 bin histograms of the individual RGB planes are concatenated to form the colour feature. Normalised histograms are compared using either histogram intersection or Euclidean distance, neither of which is clearly superior. While these features are invariant to rotation, scale and translation they fail to consider spatial relationships completely. Furthermore the coarse discretisation during histogramming makes the feature sensitive to subtle colour changes and hence less appropriate for natural scenes or full palette images. For shape, histograms of edge direction are used, treated as a one dimensional signal and smoothed to avoid coarse discretisation effects. The feature is translation and, after histogram normalisation, scale invariant, although it fails to capture or match subcomponents of the image [48]. Rotational invariance is considered by shifting the histogram. Again this feature is well suited to artificial images in which edges are well defined, however it is still limited by edge detection accuracy and is dependent on region colours (e.g. black on white and white on black.) Results are very good on a database of 400 images each containing a single trademark. Clustering images according to the mean and variance of the colour histograms (virtually equivalent to the mean and variance of the RGB planes) is performed to speed up retrievals[23]. These 6 features are illustrated using Chernoff faces (see page 31). Overall, while successful for images containing one well defined object, the techniques are not general and cannot easily be extended to natural scenes.

Photobook [246] is an example of a well designed system which is very useful for restricted predefined datasets. Considering both the work done at "input time" and "query time" the emphasis is placed on deriving a "similarity metric" for comparing image properties that "matches human judgement of similarity" and obtaining a representation that is both "perceptually complete" and "semantically meaningful." The framework uses the Karhunen-Loeve Transformation (KLT) or principal components analysis and hence has the ability to partially reconstruct an image from its features. The Photobook system uses different types of image descriptions such as the "Appearance Photobook", "Shape Photobook" and "Texture Photobook". The appearance Photobook gathers statistics from a large database to derive the mean image, e.g. mean face, and orthogonal components based on the Eigenbasis from the covariance statistics. Subsequently each image is represented by independent components in this new "face space". The method is simple, general and effective but requires images with the same basic structure and controlled positioning or effective preprocessing to provide translation invariance. The shape Pho-

[23] It is unclear why variance and not standard deviation is used. The Euclidean distance is used during clustering which is inappropriate since the units differ and variance is nonlinear.

tobook relies on preprocessing to extract silhouettes and is limited to one object in each image. It uses the Eigenbasis from a positive semi-definite "stiffness matrix" which describes how each point is connected to each other point. The texture Photobook does not use the KLT, but rather obtains texture features by decomposing the frequency spectrum into three mutually orthogonal components, namely "a harmonic field" which is periodic, "a generalised-evanescent field" with a defined orientation, "and a purely-indeterministic field." Overall Photobook provides an extremely effective set of tools for retrieval from databases of images with pre-known and strictly controlled characteristics, as illustrated on faces, tools, diagrams of fish, and Brodatz textures [246].

The systems outlined so far are not appropriate for general scenes containing numerous objects and complex spatial relationships. A more sophisticated system attempts to locate naked people in real-world scenes [247] [240]. A skin filter is used which identifies regions of appropriate colour which have no texture. Colour is represented as outlined on page 102, a preprocessing median filter is applied and located regions are expanded to fill holes given reasonably close colour and textural properties. A potential naked human is viewed as an assembly of near cylindrical parts with an appropriate relative position. Such cylinders are extracted using edges, skin regions, average cylinder width and the number of edges within each potential region. Practical assumptions are made about the range of appropriate scales. The system in [247] [240] models a human using a set of rules describing how "possible girdles and spline-thigh groups" join together. The system identifies 43% of 565 test images and correctly dismisses 96% of 4302 control images [240]. In [248] this concept of "body plans" is extended to locate horses using a model automatically inferred from training data. In general the final framework provides a means for extracting objects from complex scenes, but is highly specific to animals or other objects which comprise well structured subcomponents.

In [236] the use of refined edge thumbnail images is proposed to allow queries by visual example. The set of edge equations is equivalent to the Prewitt masks in Table 2.4 with two additional masks at $45°$ orientations. This technique forms the basis for a system which uses these edge features with additional colour features [249]. The edge gradient is taken as the maximum of the absolute response to the four convolution masks. Gradients more than one standard deviation above the mean gradient are considered as candidate edges. The windowed neighbourhood of each candidate edge is examined and labelled a true edge if "its gradient magnitude is greater than the local average plus one local standard deviation" [249]. The goal of extracting "only those edges that are *clearly perceptive*" both globally and locally proves successful in almost half of the 48 studied images. The system uses a user query (edge image) and compares this with each final edge image using numerous shifted local neighbourhood correlations to obtain a matching score. For colour the CIE-LUV space is used with $RGB = \{1, 1, 1\}$ as the white reference. A histogram with 8 bins for each colour axis is taken in each quadrant of the image. These are used as colour features, with the similarity between the each image histogram I and the query histogram Q defined as

$$S(I,Q) = \frac{\sum_{j=1}^{n} \min(I_j, Q_j)}{\sum_{j=1}^{n} Q_j}$$

Overall this is typical of standard CBIR systems. It uses a hybrid combination of global distributions and simple spatial feature, resulting in user query driven results which are extremely subjective. While illustrated results do contain similar features meaningful content is not captured.

Most other CBIR systems use similar hybrid combinations of global, local and structural features. These include QBIC, VisualSEEk, CANDID and Virage to name a few. As the growth of the world wide web and multimedia computer applications continue at a seemingly exponential rate the need and commercial benefit of such systems follows. Consequently numerous new companies, some emerging from traditional research groups, are being formed. As a result technologies are rapidly changing and the techniques and implementation details are becoming less readily available. Currently Virage appears to be attracting the most commercial and media attention [250].

2.7.2 Query by image content: IBM's QBIC

QBIC[24] is a full CBIR system supporting queries on complete scenes or individual objects based on features describing properties of colour, texture, shape, and so on [57]. Building a QBIC database involves image selection and acquisition, image clean up and image classification [251] [252]. The selection and acquisition assigns an image index and simple meta-data such as author (or artist) and title. The cleanup includes cropping, colour correction and spot toning. The classification involves computing features both globally and for individual objects which "must be identified in each scene, typically manually by outlining them prior to the queries" [251]. This is a drawback causing database information to be subjective in much the same way as meta-data keywords, with similar associated disadvantages.

> "Classification of images is more or less labour intensive, depending upon the number of visual elements that are selected for classification and their nature. Objects to outline were chosen based on the afore mentioned test subjects, thus shapes of horses, hands, etc. were always carefully outlined. Many other shapes were outlined that looked interesting, or applicable." [252]

In [57] this manual selection process is partially addressed using "semi-automated object identification". User driven programs perform an "enhanced floodfill" and "snake-based edge following" to define regions corresponding to objects. This allows the user to both identify and modify the resulting regions but still maintain precision according to

[24] Acknowledgements go to IBM's Almaden Research Center for supplying numerous unpublished technical reports, computer magazine articles and media announcements.

(a) koala bear image (b) texture

(c) colour percentage (d) colour layout

FIGURE 2.50: An example query result from QBIC. [253]

local image characteristics. It is naively suggested that "the process can be automated if the background is uniform and contrasts highly with the shape" [244].

The following features are computed by the QBIC system globally and for predefined objects.

- **Average colour**. This is based on the "average Munsell" colour and compared using Euclidean distance [57]. Although average Munsell colour is quoted, no mathematical formulation or mention of standard colour tables appears in any of [57], [254], [255], [244], [251] or [252].

- **Histogram colour**. A 64 bin colour histogram is calculated. For global features the entire histogram is used, whereas for objects or partial regions the 5 most frequent colours are stored. Histograms are compared using a predetermined distance between each colour bin according to a symmetric matrix A, i.e. $(r-q)^T A(r-q)$. Additional work [254] suggests up to third order moments in the hue-saturation plane form colour features superior to colour histograms.

- **Texture**. Measurements of coarseness, contrast and directionality form texture features which are compared using a weighted Euclidean distance [57].

- **Location**. The normalised (x,y) coordinates of the object's centroid.

- **Shape**. No significant discussion is presented in any of the papers which briefly sketch over details. For example "QBIC can also sort for more complex shapes outlined by the user" [244]. It is stated [57] that several different methods can be used suggesting a "combination of heuristic shape features (area, circularity, eccentricity, major axis) plus a set of algebraic moment invariants that together give a 20 element shape vector." Using this, shapes are compared with the Euclidean distance. No units or vector element normalisation are mentioned.

- **Sketch**. QBIC uses an unspecified Canny edge detector based algorithm similar to that used in [249] (see Section 2.7.1).

In summary the QBIC system stores the appropriate features for all legitimate manually identified regions as well as for each rectangular grid region (6 vertically and 8 horizontally) and the entire image. Queries can be made on scene text (based on metadata), scene colour from a selection of predefined Munsell colours or mixed RGB colour, scene texture, shape, or query by example [252]. A query on colour layout based on grid rectangular region features [57] and other variations are included in more recent versions.

Figure 2.50 shows results from the three available query types using an image of a koala bear. Although the features match, the results are subjective and do not really portray meaningful image content. However the practicality of the system is outlined by the following which considers a search for people in paintings.

> "Eventually, after scrolling through the selections queried on, QBIC generally finds about 16 different faces and figures with similar skin colours after about 5 queries." [252]

2.7.3 VisualSEEk

The VisualSEEk system and related research extends the basic QBIC like CBIR system by automatically extracting simple regular regions and considering spatial relationships between them. Initial techniques [256] [257] considered both texture and colour properties for primary features, examining each in depth. Numerous texture features including spatial and spatial frequency features, Gabor filters and Wavelet sub-band features are considered, but no texture features are included in the final VisualSEEk system. Colour histograms provide global features which support *query by example* queries as seen in Figure 2.51[25]. However region based *colour sets* prove superior since they provide basic size and shape information about *objects* and along with spatial relationships support queries of the form *provide images with large blue regions near the top and a small red region in the middle.* Consequently these are adopted to form the main component of VisualSEEk.

Database creation involves storing the image with associated meta-data along with a set of automatically determined regions with the parameters *imageid, regionid, colourset, position, area, height* and *width* [259]. The *colourset* parameter is a binary string defining the colours that constitute the rectangular region. The HSI colour cylinder is quantised

[25]This is a good example of a query result on a collection of distributed images. A query using a picture of a cowboy on a horse with a blue sky produced numerous light blue computer company logos, many broken links and one picture of an aeroplane. Such lack of distinction between natural scenes and artificial palette reduced images which vary significantly in simple texture is undesirable.

FIGURE 2.51: An example query result from VisualSEEk's WebSEEk interface. [258]

into 18 hues, 3 saturations and 3 intensities with an additional 4 grey values through the centre to form 166 discrete colour values. Regions are located by back projecting appropriate colours onto the image and considering colour similarity. Each region contains a sufficient proportion of the appropriate colour(s) and is stored in the database. The bits corresponding to these colour(s) and similar colours are set. This allows direct indexing of colour features. Throughout colour similarity is calculated as follows [259] [260]. The scaling value $\sqrt{5}$ ensures the value is in the range $[0, 1]$ since the hue-saturation plane has a diameter of 2.

$$1 - \frac{1}{\sqrt{5}}((i_q - i_t)^2 + (s_q \cos h_q - s_t \cos h_t)^2 + (s_q \sin h_q - s_t \sin h_t)^2)^{\frac{1}{2}}$$

For image queries a number of distance measurements are considered. For colour, a histogram distance based on several Minkowsky metrics and the histogram quadratic distance similar to that used by QBIC are examined [259]. The latter is used with the above colour similarity for colour sets, i.e.

$$d^{set}_{q,t} = (c_q - c_t)^T A (c_q - c_t)$$

For location the Euclidean distance between centroids $d^s_{q,t}$ is the default although bounded query location is considered. For area and spatial extent the absolute difference $d^a_{q,t}$ and Euclidean distance between the points $(height, width)$ of the minimum bounding rectangle $d^m_{q,t}$ are used respectively. Overall the single region query distance is given by the following distance according to user defined weight α [259].

$$d_{q,t} = \alpha_c d_{q,t}^{set} + \alpha_s d_{q,t}^{s} + \alpha_a d_{q,t}^{a} + \alpha_m d_{q,t}^{m}$$

Multiple region queries consist of combining single region queries with additional appropriate spatial relationships. These basically consider regions independently (absolute spatial query) or compare the distances and directions between them (relative spatial query). The relative spatial query is far more sensitive and typically results in no images being found even for two regions.

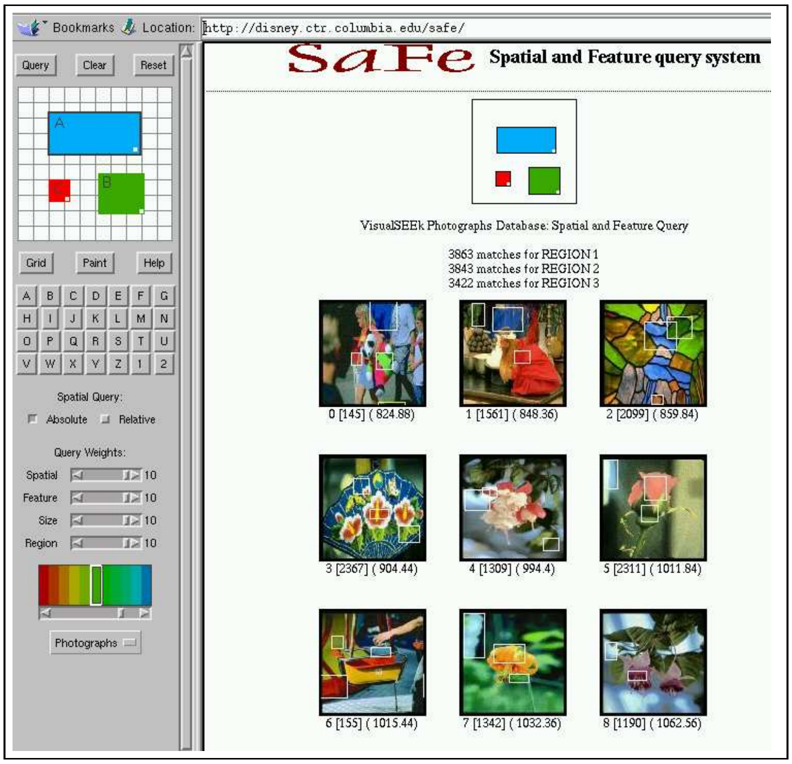


FIGURE 2.52: An example query result from VisualSEEk's SaFe interface. [258]

An example of the online VisualSEEk system is illustrated in Figure 2.52. This query took about 5 seconds (plus additional image downloading time) to perform using a database of several thousand photographs. In general VisualSEEk is still very much a hit and miss system, although it does assist in finding images with known colour relationships.

Further research [232] considers texture and shape. A system is developed which allows searches according texture using Wavelet decomposition, although results are subjective and appear no better than QBIC [258]. While shape is considered and discussed no mechanism for representing shape is adopted but rather it "is worth further study" [232]. Suggestions include simple invariant measures such as perimeter, aspect ratio and area, and Fourier descriptors, chain codes and principal components decomposition.

2.7.4 Berkeley digital library project

This section covers components of research on the *UC Berkeley Digital Library* project. The goal is to "develop technologies for intelligent access to massive, distributed collections comprising multi-terabyte databases of photographs, satellite images, maps, and text documents" [262]. The database which includes *image features* and significant amounts of textual meta-data is expected to exceed 3.4 terabytes of storage when complete. The research focus is to identify objects in pictures such as "animals, trees, flowers, buildings, and other kinds of 'things' that the user might request" [262]. The general approach is to group low level features into progressively higher levels motivated by the observation that objects are comprised of appropriately arranged coherent colour and texture regions. As such the framework is far more sophisticated than systems such as QBIC and VisualSEEk.

The HSI conical space is quantised into 13 perceptual colour channels [262] [263] [264] [265] such as red, orange, green and grey. This quantisation overcomes the problems associated with the conical space outlined in Section 2.6.2. It is suggested that these channels "match human perceptual categories and tend to distinguish objects from their background" [262]. Consequently a scheme is proposed which uses 6 features for each colour. These are the percentage of the image which is that colour, and the number of *coloured dots* of various sizes. These dots are obtained using convolution masks designed to extract these features "with radii of approximately 4, 8, 16, 32 and 64 pixels" [262]. Given the simplicity of the features the results of queries are particularly good. For example manual queries for "Pastoral Scenes" by requesting *green* > 25% AND *light blue* > 25% recalls 93 images, 85 of which are correct, and for "Yellow Cars" by searching for the meta-data text "auto" where there is at least one large or very large yellow dot yields 7 images, 6 of which contain yellow cars. While retrieved image sets are obviously not complete, this technique is very useful for querying very large databases. It is suggested that these features be stored as text to efficiently support SQL queries such as *where dots like '%pink%'*.

Belongie et. al. [266] uses essentially the same colour quantisation,[26] but introduces a simple yet effective texture measure with the intention of grouping features for automated image annotation. The "*windowed image second moment matrix*" is used which measures gradient statistics in a Gaussian window. The eigenvalues λ of the associated covariance matrix provide simple textural information. Constant intensity regions (considered featureless) yield $\lambda_{max}, \lambda_{min} \approx 0$, single orientations, lines of edges cause $\lambda_{max} \gg \lambda_{min}$ and multiple orientations or edges $\lambda_{max} \approx \lambda_{min}$ [266]. Using these, areas are deemed *rough* if λ_{min} is sufficiently high and λ_{max} is below a defined threshold. In all this forms 13 binary images. These are grouped into "blobs" using different strategies which consider 4-connectedness, orientation and diffuse characteristics. The blobs form feature vectors which include area, position, eccentricity, orientation, mean intensity, mean saturation, colour/texture bin and grouping strategy. A decision tree classifier is formed which re-

[26]Only 12 colour bins are used, as outlined in [266] (with *brown* omitted).

sults in the automatic formulation of rules allowing extremely fast classifications. Results for images with one dominant region for 208 images of 12 classes are good although "lawn" and "tree" are misclassified as "vegetation". Similar work and related issues are further discussed in [237].

An extension of the above techniques leads to a more complete system facilitating image classification [263] [264] and user based queries [265]. The HSI space is quantised into 13 colours and a further 6 bins are used for texture features. The texture bins, derived from λ_{max} and λ_{min} for the windowed second moment matrix, correspond to no texture, stochastic texture with no defined orientation, and four bins for oriented texture at 45° rotations. Of these 13 plus 6 disjoint binary image bins, the K *dominant* bins which contain 90% of the area are chosen. These initialise the EM algorithm (see Section 2.1.5) performed in the colour and texture space which results in a number of sufficiently large blobs in a new space referred to as the *blobworld*. Maximum likelihood and a connected components algorithm are used here. In summary this process identifies the large spatially disjoint regions with appropriately the same colour. A full illustration on a scene containing a polar bear and a close up of an oriental building is presented in [264] and [263] respectively.

For classification the *blobworld* is represented as a set of probabilities. The image is divided into 3×3 sections with votes cast into 78 points (13 colour and 6 texture bins) from each blob. A heuristic Bayes classifier is used; "The above scheme represents a naive Bayes classifier" [264] [263]. Results are obtained using images from the Corel professional photo CD library [267], with classes defined according to the name of the CD collection, e.g. *Airshows*, *Tigers*, *Sunsets* and *Fields*. Some images are subjectively omitted due to inappropriateness and others rotated[27]. The reasoning for this rotation is unclear since some image classes are rotated to the same relative size (all images have an aspect ratio of 2×3) causing images to be placed on their sides while others are left as is. No reasoning is given and this rotation which obviously affects classification due to the 3×3 subdivision. A final classification rate of 54.5% is obtained using over 1000 images of 12 classes.

In [265] the same techniques are used without the classification stage. Furthermore shape features based on geometric moments are added. The 3 values which represent "approximate area, anisotropy and orientation" [265] are obtained from the eigenvalues and orientation of the principal eigenvector from the spatial covariance matrix as the determinant $\sqrt{\lambda_{max}\lambda_{min}}$, $1 - \lambda_2/\lambda_1$ and θ. It is noted that these are far from ideal, essentially treating the *blob* as an ellipse (see 2.3). Image retrieval is performed using an example blob (which may be chosen from a previously processed image.) A measure of similarity based on colour, texture, position and shape is calculated from previously stored blob information. Results illustrated in [265] are far superior to typical results from QBIC or VisualSEEk although this is obviously query dependent and subjective.

Overall these techniques establish a framework which attempts to extract features

[27] Acknowledgements go to Serge Belongie from the Department of Electrical Engineering and Computer Sciences, University of California Berkeley, for supplying the list of images used, omitted and rotated.

based on structure rather than simply global distributions. This is demonstrated by the effective classification of a large set of images and the impressive user query results. As outlined in [265] characterising shape "is perhaps most important" along with an improved segmentation algorithm to begin to capture and describe more meaningful regions or objects. Such techniques will ideally lead to an automated procedure to extract higher level image content.

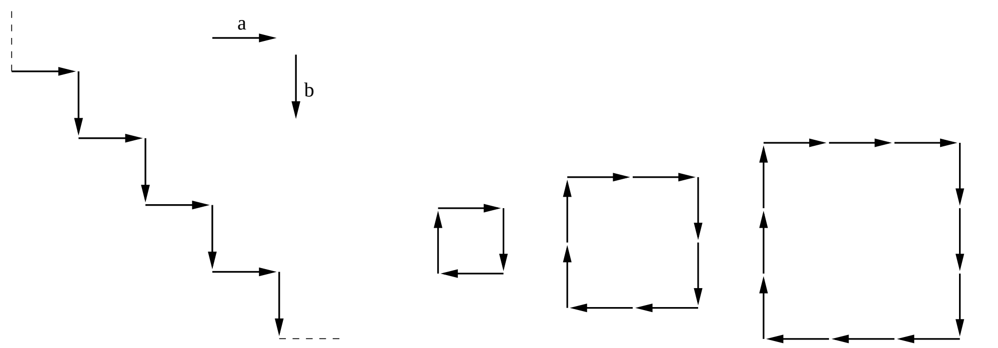
2.8 Syntactic pattern recognition

This section provides an overview of traditional syntactic pattern recognition and outlines novel approaches for automatically extracting structure from natural language text and binary stroke images. The similar frameworks used in these novel examples justify the approach used throughout this thesis as discussed in the next chapter.

2.8.1 Overview

Pattern recognition can be defined as "the categorization of input data into identifiable classes via the extraction of significant features or attributes of the data from a background of irrelevant detail" [268]. Syntactic pattern recognition is typically based on representing patterns using symbolic data structures such as strings, trees, graphs and arrays. These allow the description of relations between elementary pattern components and provide hierarchical models showing how complex patterns are constructed from simpler parts. Recognition is achieved by comparison with predefined models which determine whether a pattern agrees with the rules of a grammar describing all members of a pattern class. [269].

King Sun Fu [164], a pioneer of syntactic pattern recognition and the methodology described above, suggests representing a pattern discretely using a set of simple and predefined "pattern primitives" [270]. For example, the staircase in Figure 2.53(a) is represented as a series of 'a' and 'b' primitives. In [268] it is noted that such as representation is efficient in terms of data requirements by taking advantage of the inherent geometric



(a) staircase (...abababab...) (b) squares (abcd, aabbccdd and aaabbbcccddd)

FIGURE 2.53: The representation of patterns using primitives.

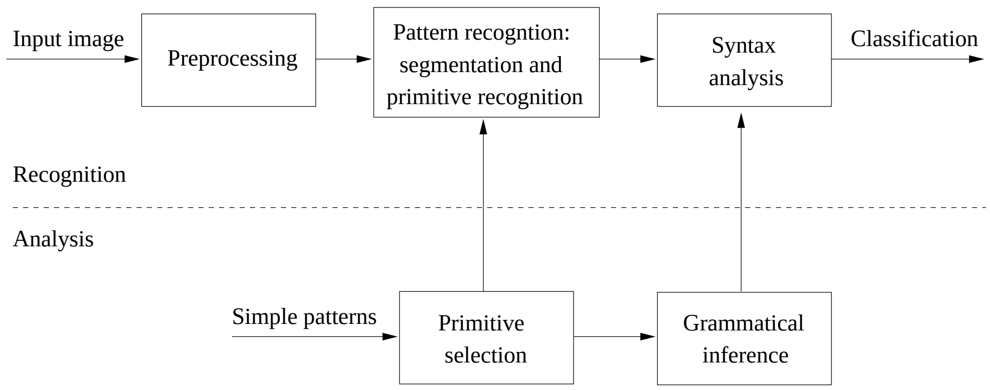


FIGURE 2.54: Overview of a simple traditional syntactic pattern recognition system, from [271, Figure 20.1].

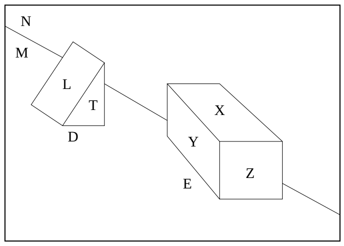
arrangement. Figure 2.53(b) illustrates the primitives required to form a square, which can be generalised by the rule, or grammar, $a^N b^N c^N d^N$.

The structure found in these descriptions and the terminology adopted by Fu bears a close resemblance to that used by linguists and in formal language theory. Consequently his overall approach is based on the mathematical models of grammars by Noam Chomsky in the 1950s.

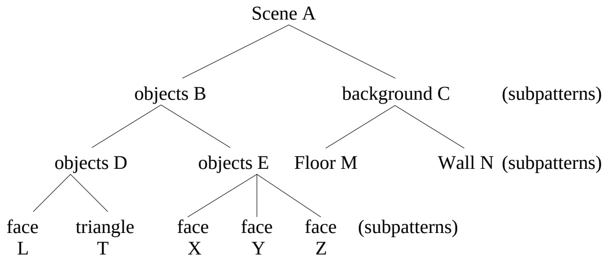
Given a number of sample patterns, bottom-up analysis forms primitives which can be described by a *grammar* of a *pattern description language*. Hence, in the ideal case, a grammar can be automatically inferred for each pattern class by a process known as grammatical inference. Grammatical inference "provides automatic methods for learning the model of the pattern class analogous to applying formulae giving the estimation of the parameters of a Gaussian distribution in $\mathbb{R}^N$" [269, p239]. Unfortunately the procedure is not that simple. In the recognition phase, input patterns are decomposed into primitives and classified according to these pre-trained grammars. An overview is shown in Figure 2.54.

With further analogies to linguistic structure, Fu suggests that images are decomposable into components over several levels, with certain geometric arrangements of entities at one level providing legitimate entities at the next. "Patterns are specified as being built up out of subpatterns in various ways of composition, just as phrases and sentences are built up by concatenating words, and words are built up by concatenating characters" [164]. Hence, the scene of Figure 2.55 is considered as a hierarchy of objects most easily represented by a tree.

Grammars associated with trees as well as traditional grammars and those associated with more complex web and graph representations are thoroughly discussed in [268], as are procedures for standardising, comparing and performing grammatical inference on limited classes of grammars. Methods for converting push down automata into finite automata and other members of the Chomsky hierarchy are in [272]. These issues are outside the scope



(a) cube scene



(b) tree

FIGURE 2.55: The hierarchical decomposition of a simple scene as presented by Fu [164].

of this thesis. However, it should be noted that there exists a one to one correspondence between grammars as language generators and automata as language recognisers [269].

For practical recognition systems it is suggested that decision-theoretic techniques be used at the lowest level to identify primitives hence accounting for noise or distortions. For example, "template matching or multiple thresholding of various feature values can be applied to primitive recognition" [164]. Furthermore, prior knowledge of the frequency of primitives allows for a Bayesian type formulation, thus producing a stochastic grammar potentially modelled by a Markov process [268]. For such a model inferring a grammar involves determining transition probabilities with recognition based on the most probable classification.

In [273] texture analysis is examined using a stochastic tree grammar with neighbours of binary pixels as primitives. The analysis is heavily restricted to well behaved homogeneous textures, using an ideal artificial texture for modelling. The basic patterns of the texture are obtained by taking windows from the model image at different offsets to form trees using "black as primitive 1, and white as primitive 0." Illustrated on two well defined textures, reconstructions from the stochastic models bear similarity to the originals although are binary (hence blocks), and lack desirable periodicity. In conclusion Fu asserts that maximum likelihood or Bayes decision rule could be used for texture classification and outlines the need for a general grammatical inference scheme to model the images realistically.

In general applications of syntactic pattern recognition are discussed qualitatively, theoretically, or applied to specific areas with well defined characteristics [269, p369–532]. One of the more promising applications is online Chinese character recognition in which stroke primitives are more easily identified and importantly ordered. Classification of around 95% is achieved using template matching with these ordered primitives [274].

This traditional approach to syntactic pattern recognition is useful, although retarded by the requirement to specify low level primitives and predefined spatial relationships or primitive ordering. For images, the methods lack rotational invariance and the representations are not robust in terms of noise, occlusion and overlapping or connected objects. For example, the decomposition of the digit 9 into small stroke primitives contains one sub-pattern corresponding to the loop and another to the remaining stokes [69, Fig 4.4]. However the primitive for a hand-drawn digit could quite easily form a sub-pattern for the right hand stroke, thus dramatically altering the string representation and making it difficult for grammatical inference. Much the same can be said for a house sketch consisting of a triangle over a square [164, Fig 3.13].

2.8.2 Grammatical inference of natural language

This section continues from Section 1.1.1 which introduced and overviewed the basic concepts behind using a local predictor to form a stochastic language model. These techniques use local statistical information automatically obtained from training data to syntactically extract structure. Processing, which is performed hierarchically from the bottom-up, makes few prior assumptions to infer a grammar. To some extent this approach does fit the traditional syntactic recognition system depicted by Figure 2.54 since once a language model has been formed, the probability of a string belonging to the language can be assigned.

At the lowest level an entropic *chunking* procedure is used to segment strings into chunks possibly corresponding to words [277]. This uses an n-th order Markov model, which when given a sequence of symbols is able to predict the next according to a probability distribution over all possible symbols. The information I of a symbol ω in context s and entropy H of the statistical language model M are defined according to information theory [4] [3].

$$
\begin{aligned}
I(\omega|M,s) &= -\log_2 P(\omega|M,s) \\
H(M,s) &= \sum P(\omega|M,s)I(\omega|M,s)
\end{aligned}
$$

Hutchens [277] used 3.5 megabytes of Sherlock Holmes data to train a second order Markov model. After training this model provides a probability distribution based on the previous two symbols which are used to measure instantaneous entropy as seen in Figure 2.56. The chunking procedure thresholds this entropy to successfully identify 3350 frequent chunks most of which correspond to English words. The most frequent are *the*, *a*, *of*, *to*, *and* and *that* (6000–22000 occurrences) and the non-English words consist of two or more words, e.g. *in the*, *it was* and *do you think that* [277]. Experiments using all uppercase and removing non-alphabetic characters, such as white space, are also remarkably effective.

In terms of hierarchical decomposition, once words are identified it is desirable to group these into higher level entities such as verbs or nouns. Hutchens [278] suggest that contexts may be considered similar and hence grouped if “the language model makes similar predictions from two different contents.” Consequently groups are determined by clustering prediction vectors from the language model. Table 2.8 shows the most frequent

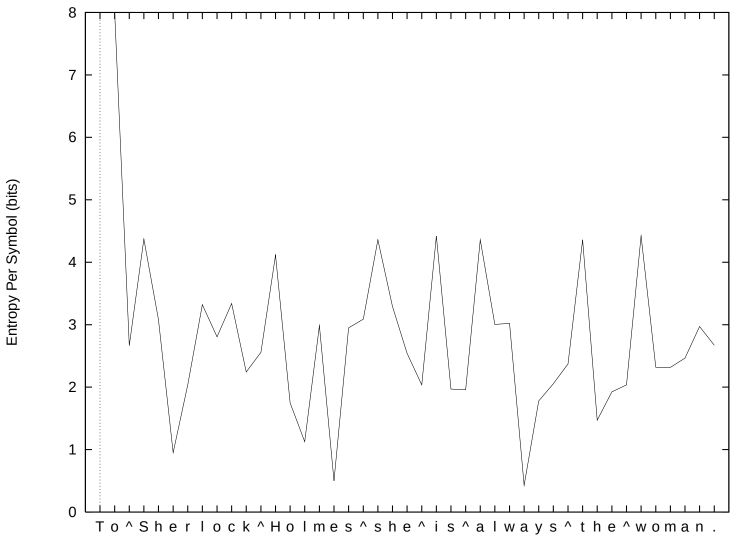


FIGURE 2.56: Entropy across a text segment, from [277].

symbol from these classes from clustering[28] using 200 kilobytes of Sherlock Holmes stories. Only a fraction of text is used due to the computational complexity of the clustering algorithm. Results are very impressive, corresponding to what seems intelligent higher level information even though no prior assumptions or inbuilt knowledge is used.

!	.	?	and
,	-	:	;
is	was	were	are
find	have	get	take
beautiful	attractive	gorgeous	startling
carried	recounted	aided	fabricated
America	Chicago	England	
Douglas	Hudson	Larbey	
detective	inspector		
pistols	six-shooters		

TABLE 2.8: Class symbols found in natural language text. Taken from [278, Table 2].

Using these same predictive models at the word level provides analysis of sentence structure as well as a mechanism for producing sentences based on the model. A second order Markov model is sufficient to produce output resembling natural language [1]. Such modelling is used by an online program called MegaHAL to simulate human conversation [279]. The program uses a forward and backward model to construct sentences based on keywords obtained from user input. Keywords are selected based on information (e.g. *learned* is chosen over *how* since it is less frequently used and hence contributes more information) with responses generated by expanding the sentence until valid start and end

[28] Although not outlined, a simple cluster tree algorithm (see page 33) is used.

words are reached. Several hundred responses are generated each second with the response chosen to maximise information in an attempt to provide the most unique answer. The program begins with no knowledge or language model, and iteratively *learns* from human input. Considering the simplicity of the modelling responses seem phenomenally intelligent at times, although its context-free nature often makes for disjointed conversations.

The techniques outlined here provide methods for automatically and hierarchically extracting structure from natural language with virtually no assumptions. It is important to note the reuse of the same techniques for modelling at the character and word level. Other than areas of computer intelligence, the main applications lie in compression where this predictive modelling outperforms *gzip* on text files and is comparable with binary files [278] although significantly slower. The ability to achieve such compression indicates the power of the model as a predictor. See [14, p383–428] for further discussion.

2.8.3 Stroke detection in binary images

This problem is discussed in Section 1.1.2.

While picture recognition and scene analysis are considered by Fu [164], these proposed methods require the selection of a suitable set of graphical primitives. These are typically manually chosen and designated discrete symbols. The requirement to initially identify primitives suffers from problems where similar looking sets of pixels are assigned to different primitives. In general, such premature quantisation prejudices subsequent classification which cannot easily be accounted for. Hence, these systems lack robustness and are impractical. Furthermore, even given correct primitive assignment, simple images give rise to grammars which are not even context-free let alone finite state, making grammatical inference extremely difficult [280].

An alternative which avoids premature quantisation is to use a continuous or *topological grammar* [281]. In this class of grammars elements at any abstract level can deform continuously rather than being initially quantised to a symbol or predefined primitive. For example, "positive semi-definite, symmetric, quadratic forms" are used as primitives in [282]. These correspond to the covariance matrix formed by taking second order central geometric moments. Using these primitives, i.e. taking local geometric moments, provides a level of robustness and insensitivity to noise. Primitive extraction or recognition only requires a choice of scale and an assignment of object colour, i.e. black pixels on a white background or vice versa. Algorithms, methods and a general framework are now discussed.

An algorithm for extracting primitive for performing stroke detection is used by McLaughlin [7] [283] [284] [285] [286]. It locates black line segments in binary images by these steps.

- Choose a black pixel or point in $\mathbb{R}^2$. In implementation this is achieved by scanning the image sequentially.

- Define a small circular local neighbourhood with a *resolution radius* r. Assuming the neighbourhood contains sufficient points (i.e. is not just noise) tag these points to form a subset of the image.

- Calculate the mean $[\mu_x \mu_y]^T$ and covariance matrix $\mathbf{C}$ of this subset. The matrix $\mathbf{C}$ is a positive semi-definite quadratic form[29] and the set E

$$E = \left\{ \begin{bmatrix} x \\ y \end{bmatrix} : \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix}^T \mathbf{C}^{-1} \begin{bmatrix} x - \mu_x \\ y - \mu_y \end{bmatrix} = 1 \right\}$$

  forms an ellipse [283], which models the distribution of the neighbourhood. The most appropriate representation of this neighbourhood is unclear. Intuitively it may be modelled by the number of data points, mean and eigenvalues and principal eigenvector orientation of $\mathbf{C}$. However, it is desirable to form a point in Euclidean space, in this case $\mathbb{R}^6$. Possibilities are to represent $\mathbf{C}$ by its unique variance and covariance elements or by the unique elements of $\mathbf{B}$, where $\mathbf{B}^T\mathbf{B} = \mathbf{C}$, since these are linear.

- Remove points from the image that are within the modelled neighbourhood [286] [285]. An alternative is to remove only those that agree with this local model, for example [283] points within the ellipse E or a standard sized ellipse.

- Repeat this process over the entire image. An example is shown in Figure 2.57.

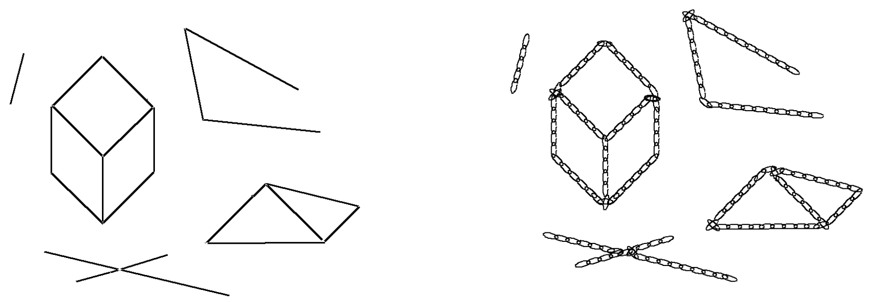


FIGURE 2.57: The Spot Algorithm in action. [283]

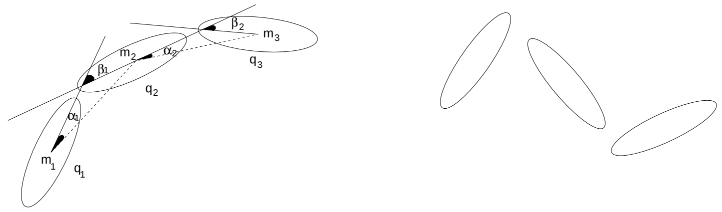
The above algorithm, named *The Spot Algorithm* [33], transforms data points in $\mathbb{R}^2$ (which correspond to pixels) into points in $\mathbb{R}^6$ corresponding to local line segments or solid objects. Given at least one adjacent line segment primitive it is possible to predict the location of potential neighbouring primitives analogous to predicting the next character in natural language. There are several approaches for amalgamating line segments into strokes, a process referred to as *chunking*.

[29] In general the eigenspace of the matrix gives rise to a basis in which the standard inner product is positive definite [44]. However, if the matrix is degenerate, such as that formed by a straight line with absolutely no orthogonal variation, one of the eigenvalues is reduced to zero making it only semi-definite.

In [284] each quadratic form is initially labelled *unused*. Starting with the next unlabelled local model, *adjacent* quadratic forms are assigned the same label if they have "similar eigenvectors (to within some tolerance)". This process is repeated until all quadratic forms are labelled. A variation to consider gradual changes present in curved strokes line circles and ellipses is also presented.

Lim [287] formulates an entropy measurement for combining local models using a p.d.f. based on shape, displacement and angular changes assuming independence of these. This provides a measure of predictability between primitives similar to that used by Hutchens for natural language. The entropy value is calculated by modelling entire images and thresholded for recognition. For 128 images of 3 classes (curtains, heads and trees with progressively higher entropy) a recognition rate of 84% is achieved. This is remarkably good considering only a simple global measurement is used.

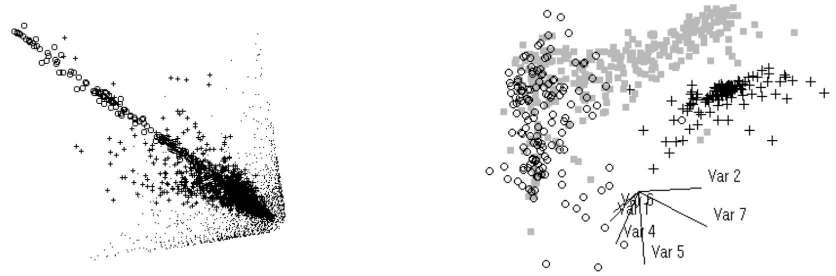
Perhaps the most general amalgamating scheme introduces the concept of the *smooth triple* [288] [289] [287], as depicted in Figure 2.58. Three neighbouring quadratic forms are examined to determine the parameters α_1, α_2, β_1 and β_2. Rather than defining a function or tolerance parameter, supervised training is performed by manually identifying smooth and irregular triples. As is intuitively expected these form a tight cluster around the origin (corresponding to a straight line). Amalgamation is performed by determining if the next triple is smooth or the end of the sequence.



(a) Smooth triple (b) Irregular triple

FIGURE 2.58: The concept of a smooth triple used to determine whether line segments belong to a stroke. From [287, Figures 7.4 and 7.3(b)].

Once a stroke has been detected it must be represented for classification or higher level processing. One approach is to backtrack, modelling all pixels which make up the stroke as a single entity. In [284] second and third order geometric moments are used to form a point in $\mathbb{R}^7$ exhibiting translation invariance. As seen in Figure 2.59(a) for strokes corresponding to lines, circles and ellipses, this space degenerates to a singularity as the scale reduces to zero (bottom right). Furthermore it is apparent that circles are a special case of an ellipse and thin ellipses look similar to lines. The obvious disadvantage of discarding the original primitives is the lack of distinction between lines and ellipses. It is also noted that the N degrees of freedom of an object class gives rise to N dimensional manifolds in the new space. To reduce this effect Zernike moments are suggested [8] [33]



(a) geometric moments (order 2 and 3) (b) Zernike moments (order 2 to 4)

FIGURE 2.59: Projections of the feature space obtained by modelling ideal and hand-drawn strokes corresponding to lines, circles and ellipses, from [284, Figure 11] and [33, Figure 5]. In (a) .:line; o:circle; +:ellipse. In (b) +:line; o:circle; square:ellipse.

to provide scale invariance and a naturally separable rotation component. As discussed in [33] the set of all strictly defined lines corresponds to a single point and ellipses to an embedded one dimensional structure according to the ratio of major axes. This is apparent in Figure 2.59(b) although effected by irregular hand-drawn objects.

An alternative and conceptually more attractive approach is to extract information from a set of primitives as points in $\mathbb{R}^6$ in the same way as information is extracted from pixels in $\mathbb{R}^2$. Taking up to second order moments in $\mathbb{R}^6$ leads to a single zeroth order moment (the number of points), 6 first order moments (the means) and a 6 by 6 covariance matrix with 21 unique elements, thus a point in $\mathbb{R}^{28}$. This approach is used in [290] for describing an entire aircraft silhouette as a single point in $\mathbb{R}^{28}$ and similarly for distinguishing between left and right hands by inferring structure [291]. It is also successful for distinguishing between cartoon characters [292] and performing real-time head recognition [287].

These techniques provide a framework for mapping a set of objects to a single higher level entity which is referred to as the *UpWrite*. In general the UpWrite is an "operation which takes a set of points in $\mathbb{R}^N$ and produces a smaller number of points in a generally higher dimensional space $\mathbb{R}^M$" [292]. For stroke detection a set of pixels in $\mathbb{R}^2$ from a set of line segments in $\mathbb{R}^6$, which are in turn used to represent an object in $\mathbb{R}^{28}$. In this case it is practical for a first UpWrite to use the spot algorithm or similar to identify strokes and remodel each stroke as a single point in $\mathbb{R}^6$. A second UpWrite would combine these points appropriately to model several strokes or possibly an entire image. Sufficiently close points are recognised as the same type of object by clustering methods or the systems uses classified examples for training.

The UpWrite is a hierarchical process with multiple levels which provides a framework for automatic unsupervised syntactic pattern recognition. In principle it is general assuming only the level of localisation, although this can be determined by correlating structure at multiple resolutions. In practice appropriate representations and methods for determining things like *smooth triples* are required. Even so the methods are easily transferable from one application to another within the same image class.

In terms of practical applications these principles have been successfully applied to autonomous robot navigation [13], recognition of simple shapes [283] [7] inference of hand structure [293] [291], recognition of aircraft [290] [294], recognition of whales [11], distinction between cartoon characters [287] [292], and real-time head detection [295] [287] [296] [288] [297] [298]. The approach is fast, making use of local information and hence is less dependent on global image complexity. For example the location of an ellipse from a simple image takes about 1.2 seconds compared to about 100 seconds for the Hough transform. Furthermore it is scalable, taking under 2 seconds for 5 ellipses with an accuracy of 80% compared to 1800 seconds and 20% for the Hough transform [284]. The head detection system takes 270 milliseconds incurring an overhead of 125 milliseconds from the frame grabber [287]. Extensions and applications to generalised grey scale and colour images with region based characteristics are the topic of this thesis. Previously these have been preprocessed to one dimensional stroke problems.