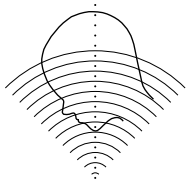
Nonparametric Markov Random Field Models for Natural Texture Images

Rupert D. Paget

BEng(Honours), Electrical and Communications Engineering University of Tasmania, Australia

Cooperative Research Centre for
Sensor Signal and Information Processing,
Department of Computer Science & Electrical Engineering,
The University of Queensland,
St Lucia, Queensland 4072, Australia.



Submitted as a requirement for the degree of
Doctor of Philosophy (Ph.D.),
The University of Queensland,
February 1999.

Abstract

The underlying aim of this research is to investigate the mathematical descriptions of homogeneous textures in digital images for the purpose of segmentation and recognition. The research covers the problem of testing these mathematical descriptions by using them to generate synthetic realisations of the homogeneous texture for subjective and analytical comparisons with the source texture from which they were derived. The application of this research is in analysing satellite or airborne images of the Earth's surface. In particular, Synthetic Aperture Radar (SAR) images often exhibit regions of homogeneous texture, which if segmented, could facilitate terrain classification.

In this thesis we present noncausal, nonparametric, multiscale, Markov random field (MRF) models for recognising and synthesising texture. The models have the ability to capture the characteristics of, and to synthesise, a wide variety of textures, varying from the highly structured to the stochastic. For texture synthesis, we introduce our own novel multiscale approach incorporating a new concept of local annealing. This allows us to use large neighbourhood systems to model complex natural textures with high order statistical characteristics. The new multiscale texture synthesis algorithm also produces synthetic textures with few, if any, phase discontinuities. The power of our modelling technique is evident in that only a small source image is required to synthesise representative examples of the source texture, even when the texture contains long-range characteristics. We also show how the high-dimensional model of the texture may be modelled with lower dimensional statistics without compromising the integrity of the representation. We then show how these models – which are able to capture most of the unique characteristics of a texture – can be for the "open-ended" problem of recognising textures embedded in a scene containing previously unseen textures. Whilst this technique was developed for the practical application of recognising different terrain types from Synthetic Aperture Radar (SAR) images, it has applications in other image processing tasks requiring texture recognition.

Key words

- Markov Random Fields;
- Gibbs Distributions;
- Parametric Estimation;
- Nonparametric Estimation;
- Parzen Window Density Estimator;
- ANOVA Model;
- Texture Modelling;
- Texture Synthesis;
- Open Ended Texture Classification;
- Multi-resolution;
- High-order Statistics;
- Stochastic Relaxation;
- Deterministic Relaxation;
- Multiscale Relaxation;
- Simulated Annealing;
- Parallel Processing;
- Discriminant Analysis;
- Wilcoxon Test;
- Kruskal-Wallis Statistic;
- Terrain Mapping;
- Synthetic Aperture Radar.

Contents

List of Figures

List of Tables

Abbreviations and acronyms

ANOVA	Analysis-of-Variance
AR	Autoregressive
ARMA	Autoregressive Moving Average
CRC	Cooperative Research Centre
CSSIP	CRC for Sensor Signal and Information Processing
DSTO	Defence Science and Technology Organisation
EROS	Earth Resources Observation Systems
FFT	Fast Fourier Transform
GLCM	Grey Level Co-occurrence Matrix
GMRF	Gaussian Markov Random Field
ICM	Iterative Conditional Modes
IFFT	Inverse Fast Fourier Transform
JPL	Jet Propulsion Laboratory
LANDSAT	Land Satellite
LCPDF	Local Conditional Probability Density Function
MA	Moving Average
MAP	Maximum *A Posterior*
MCMLE	Monte Carlo Maximum Likelihood Estimator
MLE	Maximum Likelihood Estimator
MPLE	Maximum Pseudo-Likelihood Estimator
MPM	Maximum Posterior Marginal
MR	Multiscale Relaxation
MRF	Markov Random Field
NASA	National Aeronautics and Space Administration
PDF	Probability Density Function
RADAR	Radio Detection and Ranging
SAR	Synthetic Aperture Radar
SAR	Simultaneous Autoregressive
SARMA	Simultaneous Autoregressive Moving Average
SPOT	Système Probatoire d'Observation de la Terre
SR	Stochastic Relaxation

Symbols

$\Re$	Set of Reals.
S	Lattice.
s	A site on the Lattice S.
$\mathcal{N}$	Set of Neighbourhoods.
$\mathcal{N}_s$	Neighbourhood for site s.
$\mathcal{C}$	Set of Cliques for $\mathcal{N}$.
$\mathcal{C}_s$	Local Set of Cliques for $\mathcal{N}$ which all contain the site s.
$\mathbf{A}$	Set of sites on S.
$\mathbf{B}$	Set of sites on S.
$\mathbf{X}$	Set of Variables on S.
$\mathbf{x}$	Set of Values of $\mathbf{X}$.
$\mathbf{Y}$	Set of Variables on S.
$\mathbf{y}$	Set of Values of $\mathbf{Y}$.
Λ	State Space.
Ω	Configuration Space.
$\mathcal{H}$	Hamiltonian Energy.
Π	Joint Distribution.
Π_s	Local Conditional Probability Density Function (LCPDF).

Publications Arising

These are the publications and reports which have been produced by, or in conjunction with, the author, during his Ph.D. candidacy which have arisen from this research:

[1] R. Paget and D. Longstaff, "Texture synthesis and unsupervised recognition with a nonparametric multiscale Markov random field model," in *Proceedings of 14th International Conference on Pattern Recognition*, (Brisbane, Australia), IAPR, vol. 2, pp. 1068–1070, August 1998.

[2] R. Paget and D. Longstaff, "Texture synthesis via a noncausal nonparametric multiscale Markov random field," *IEEE Transactions on Image Processing*, vol. 7, pp. 925–931, June 1998.

[3] R. Paget and D. Longstaff, "Texture synthesis and classification with nonparametric multiscale Markov random field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1997. submitted for publication.

[4] J. Homer, J. Meagher, R. Paget, and I. D. Longstaff, "Terrain classification via texture modelling of SAR and SAR coherency images," in *Proceedings of 1997 IEEE International International Geoscience and Remote Sensing Symposium (IGARSS'97)*, (Singapore), pp. 2063–2065, August 1997.

[5] R. Paget, D. Longstaff, and B. Lovell, "Texture classification using nonparametric Markov random fields," in *Proceedings of the 1997 13th International Conference on Digital Signal Processing*, vol. 1, (Hellas, Santorini, Greece), pp. 67–70, IEEE Signal Processing Society, July 1997. Invited Paper.

[6] R. Paget and D. Longstaff, "Extracting the cliques from a neighbourhood system," *IEE Proceedings Vision, Image and Signal Processing*, vol. 144, pp. 168–170, June 1997.

[7] R. Paget and D. Longstaff, "A nonparametric multiscale Markov random field model for synthesising natural textures," in *Fourth International Symposium on Signal Processing and its Applications*, vol. 2, (Gold Coast, Australia), pp. 744–747, ISSPA 96, August 1996.

[8] R. Paget and D. Longstaff, "Non–parametric Markov random field texture synthesis." CSSIP Tekfest – Centre for Sensor Signal and Information Processing, February 1996.

[9] R. Paget and D. Longstaff, "Texture synthesis via a nonparametric Markov random field," in *Proceedings of DICTA–95, Digital Image Computing: Techniques and Applications* (A. Maeder and B. Lovell, eds.), vol. 1, (Brisbane, Australia), pp. 547–552, Australian Pattern Recognition Society, December 1995.

[10] R. Paget, D. Longstaff, and G. Newsam, "A practical approach to using Markov random fields in texture analysis." CSSIP Tekfest – Centre for Sensor Signal and Information Processing, February 1995.

[11] R. Paget, D. Longstaff, and B. Lovell, "Multiscale texture segmentation of synthetic aperture radar images." IEEE ICASSP–94, International Conference on Acoustics, Speech and Signal Processing – student presentation, April 1994.

[12] R. Paget, D. Longstaff, and B. Lovell, "Multiprocessor implementation of a texture segmentation scheme for satellite radar images." CSSIP Tekfest – Centre for Sensor Signal and Information Processing, February 1994.

[13] R. Paget, D. Longstaff, and B. Lovell, "Multiprocessor implementation of a texture segmentation scheme for satellite radar images," in *DICTA–93, Digital Image Computing: Techniques and Applications* (K. K. Fung and A. Ginige, eds.), vol. 1, (Sydney), pp. 203–211, Australian Pattern Recognition Society, December 1993.

[14] R. Paget, D. Longstaff, and B. Lovell, "Texture recognition in SAR images." CSSIP Tekfest – Centre for Sensor Signal and Information Processing, November 1992.

[15] R. Paget, D. Longstaff, and B. Lovell, "Texture recognition in SAR images," in *Workshop in Pattern Analysis/Recognition*, vol. 1, (Townsville), pp. 1–3, Australian Pattern Recognition Society, October 1992.

Declaration

The work contained in this thesis has not been previously submitted for a degree or diploma at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

Signed: ..... Date:.....

Acknowledgements

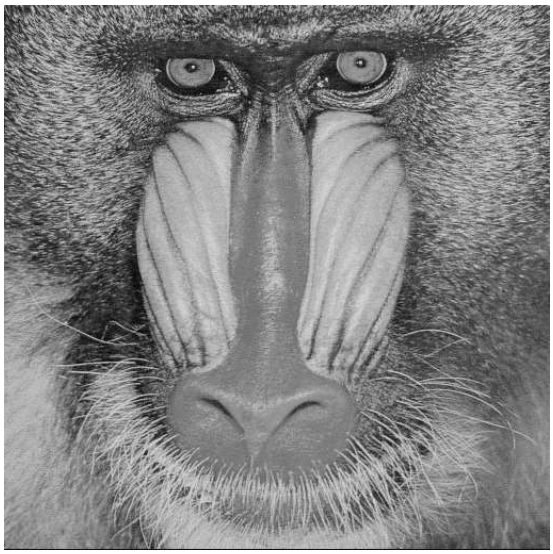
The author would like to thank the University of Melbourne for the extensive use of the MasPar. Also Project Ingara, Microwave Radar Division, Electronics and Surveillance Research Laboratory at DSTO for providing the Airborne SAR image of Cultana and Adelaide, Australia. In acknowledgement of the valuable contributions made towards the research of this thesis, the author would also like to thank the following people: Christine Graffigne, from the Université de Paris-Sud, for advise on Multiscale analysis; Garry Newsam and David Howard, from the Centre for Sensor Signal and Information Processing (CSSIP), for advice on Markov random fields; Andrew Bradley and Andrew Mehnert, from CSSIP, for advise on nonparametric statistical classification; and Dennis Longstaff, also from CSSIP, for bearing and guidance.

Contribution

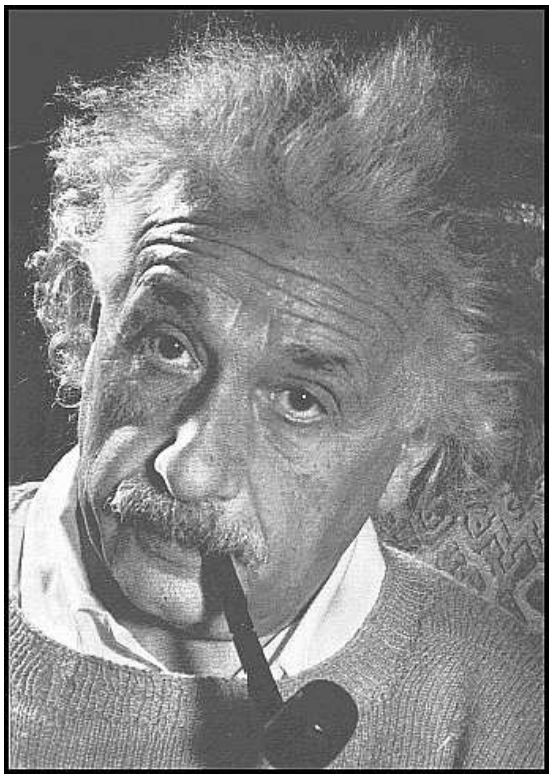
The original work in this thesis is the construction of what appears to be a universal texture model, in that it is capable of capturing the characteristics of almost any given texture. This is demonstrated by our ability to use the model to regenerate any given texture with a high degree of similarity to, and over a larger field than, the training texture. The synthesis algorithm itself also incorporates original work, in particular a novel pixel temperature function. There is also the approach taken by us for the extraction of texture information from our model for the purpose of open ended texture classification. The classification scheme is proposed by us for the purpose of recognising specific texture types within an image when no prior knowledge exists of the background texture types. In this thesis we also show, for the first time, how the Moussouris's [148] Strong Markov random field model is mathematically equivalent to the Analysis-of-variance model.

Chapter 1

Introduction



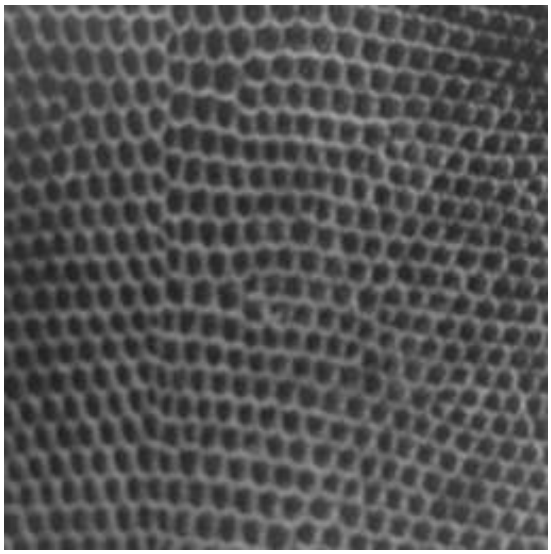
(a) Baboon



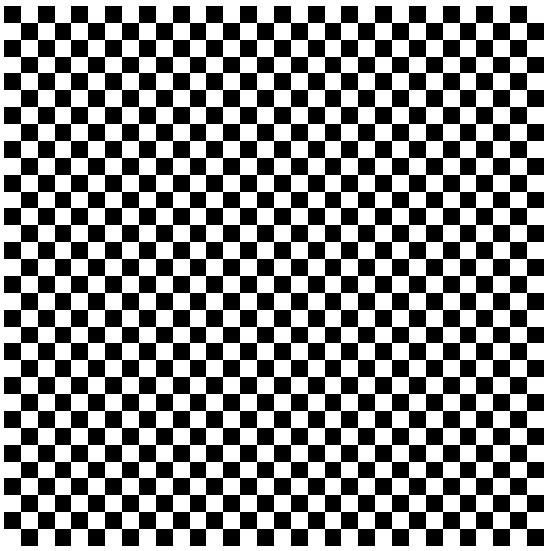
(b) Einstein

Figure 1.1: Texture in images can represent different types of hair, skin, or jumper someone is wearing.

In an image, texture is one of the visual characteristics that identifies a segment as belonging to a certain class, as illustrated in Fig. 1.1. We recognise many parts of the image by texture rather than by shape *e.g.,* fur, hair, knitted. If the texture belongs to a class that has a particular physical interpretation such as grass, hair, water or sand, then it may be regarded as a "natural" texture. Well known photographic examples of some natural textures are shown in the Brodatz album [28], an example of which appears in Fig. 1.2(a). On the other hand, a texture may belong to a class identified by artificial visual characteristics that have a concise mathematical



(a) Brodatz Texture: Reptile skin



(b) Checker Board

Figure 1.2: Two different types of textures. a) Is a stochastic texture. b) Is a deterministic texture.

interpretation. One example would be the checkerboard pattern, which is illustrated in Fig. 1.2(b).

A complete definition of texture has been elusive as there does not exist an all encompassing mathematical model of texture. However from a human perspective we may conjecture that texture is a quality that distinguishes regularity in the visual appearance of local detail. [50, 78, 196].

1.1.1 Human perception of texture

Julesz's [110] classic approach for determining if two textures were alike was to embed one texture in the other. If the embedded patch of texture visually stood out from the surrounding texture then the two textures were deemed to be dissimilar. The comparisons relied solely on pre-attentive human visual perception, where the subjects were only given a very brief time to view the texture [14]. Julesz found that texture with similar first order statistics, but different second-order statistics, were easily discriminated. However Julesz could not find any textures with the same first and second-order statistics, but different third-order statistics, that could be discriminated. This led to the Julesz conjecture that "iso-second-order textures are indistinguishable." This was further substantiated with work on the visual

discrimination of stochastic texture fields [166].

However, later Caelli, Julesz, and Gilbert [31] did produce iso-second-order textures that could be discriminated with pre-attentive human visual perception. Further work by Julesz [111, 112] revealed that his original conjecture was wrong. Instead, he found that the human visual perception mechanism did not necessarily use third-order statistics for the discrimination of these iso-second-order textures, but rather used the second order statistics of features he called *textons*. These textons he describes as being the fundamentals of texture. Julesz [111, 112] found three classes of textons: *colour*, *elongated blobs*, and the *terminators* (end-points) of these elongated blobs. Julesz revised his original conjecture to state that, "The human pre-attentive human visual system cannot compute statistical parameters higher than second order." He further conjectured that the human pre-attentive human visual system actually uses only the first order statistics of these textons.

Since these pre-attentive studies into the human visual perception, psychophysical research has focused on developing physiologically plausible models of texture discrimination. These models involved determining which measurements of textural variations humans are most sensitive to. Textons were not found to be the plausible textural discriminating measurements as envisaged by Julesz [13, 150, 204]. On the other hand, psychophysical research has given evidence that the human brain does a spatial frequency analysis of the image [32, 56, 83]. Therefore a number of models are now based on the responses of orientated filter banks [13, 15, 44, 139]. Tamura *et al.* [190] and Laws [130] identified the following properties as playing an important role in describing texture: uniformity, density, coarseness, roughness, regularity, linearity, directionality, direction, frequency, and phase. However these perceived qualities are by no means independent.

A unified model with discriminating powers equal to that of human visual perception, that has attained universal acceptance, has not been reported. For this reason, a concise definition of texture does not exist in the literature, although some authors claim to give one. Haindl [95] states that: "Texture is generally a visual property of a surface, representing the spatial information contained in object surfaces." Bennis and Gagalowicz [12] suggest texture may represent information that permits the human eye to differentiate between image regions. Another definition by Francos and Meiri [71] states: "Texture is a structure which is made of a large ensemble of elements that resemble each other very much, with some kind of an order in their locations, so that there is no single element which attracts the viewer's

eye in any special way. The human viewer gets an impression of uniformity when he looks at a texture." Coggins [46] has compiled a catalogue of texture definitions in the computer vision literature [98, 102, 172, 184, 191, 212]. The abundance and range of the definitions of texture demonstrates that there is no one particular all encompassing definition but many, where texture is defined with respect to the chosen application. Some are perceptually motivated, and others are driven completely by the application in which the definition will be used.

1.1.2 Computer analysis of texture

The vague definition of texture leads to a variety of different ways to analyse texture [98, 201]. The literature suggests three approaches for analysing textures [3, 43, 59, 98, 196, 205].

Statistical texture measures A set of features is used to represent the characteristics of a textured image. These features measure such properties as: contrast, correlation, and entropy. They are usually derived from the "grey value run length," "grey value difference," or Haralick's "co-occurrence matrix" [100]. Features are selected heuristically and an image similar to the original cannot be re-created from the measured feature set. A survey of statistical approaches for texture is given by Haralick [98].

Structural texture modelling Some textures can be viewed as two dimensional patterns consisting of a set of primitives or subpatterns (*i.e.,* textons [111]) which are arranged according to certain placement rules. Examples of such textures are brick wall, mosaic floor, or wire braid. The primitives used are areas of constant grey level, edges, lines, curves, and polygons. Correct identification of these primitives is difficult. However if the primitives completely captivate the texture, then it is possible to re-create the texture from the primitives. Fig. 1.2(b) illustrates one such texture for which primitives can completely capture the texture. A survey of structural approaches for texture is given by Haralick [98]. Haindl [95] also covers some models used for structural texture analysis.

Stochastic texture modelling A texture is assumed to be the realisation of a stochastic process which is governed by some parameters. Analysis is performed by defining a model and estimating the parameters so that the stochas-

tic process can be reproduced from the model and associated parameters. The estimated parameters can serve as a feature for texture classification and segmentation problems. This approach offers the best possibility for re-creating realistic examples of natural textures from a model. However, until our recent success with MRF models [156], there had been little progress in this regard. In this thesis, we demonstrate that a nonparametric MRF can represent a wide range – if not all – spatially uniform textures. An overview of some of the models used in this type of texture analysis is given by Haindl [95].

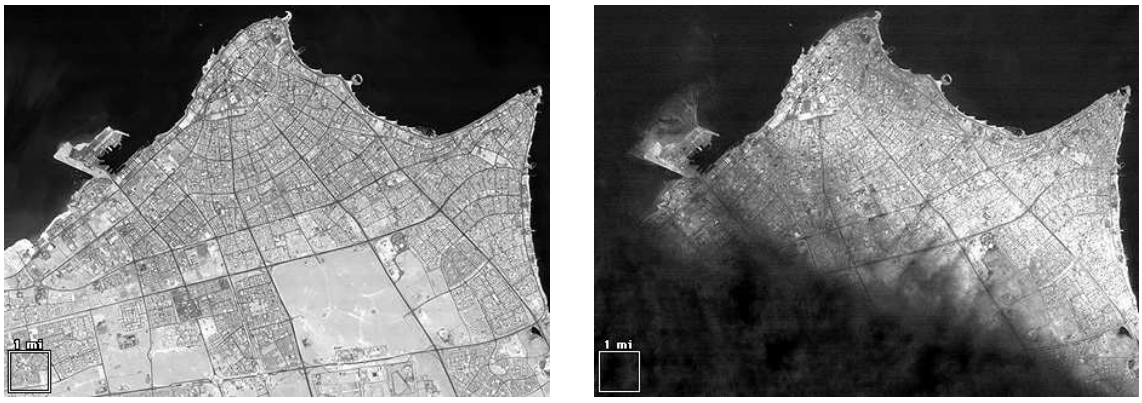
1.2 Application

1.2.1 Satellite images

Satellite images have many applications. Environmental monitoring, development planning, disaster analysis, exploration, and military intelligence are just some of the areas now using satellite data [62, 63]. Remote sensing imaging systems offer an effective approach for generating large scale images of the Earth's surface.

LANDSAT and SPOT satellite images in particular are widely used to extract terrain information from the images. Terrain classification maps have numerous applications ranging from hydrological, geological, geophysical and urban management studies to the detection of deforestation or desertification and the monitoring of vegetation on a global scale [93]. With the increasing need to assess the environmental impact of humans on the global climate and natural ecosystems [62], large scale terrain monitoring of the Earth's surface is essential.

Conventional imaging systems, such as those used by the SPOT and LANDSAT satellites, are equipped with passive imaging systems that produce high resolution photographic images. These systems take pictures at various bands in the visible and infrared spectrum [132, 171]. Identification of the terrain type is accomplished through the analysis of the various spectral responses for a particular pixel location in the image using the natural illumination provided by the sun. From the spectral response it is possible to identify the quantity and type of vegetation that is present [171]. A detailed map of the terrain (with an approximate resolution of 15-20 metres) may be obtained without physically visiting the image site. This is, of course, a huge advantage when large and inaccessible areas need to be surveyed. Two LANDSAT images are shown in Fig. 1.3.



(a) August 1990 (b) February 1991

Figure 1.3: Two LANDSAT images of Kuwait city which show the adverse effects of atmospheric conditions on passive remote sensing. a) Kuwait city before the Gulf War of January-February 1991. b) Kuwait city after the Gulf War, where the image is partially obscured by thick black smoke. The images appear courtesy of Earth Resources Observation Systems [61].

The problem with passive remote sensing, as indicated in Fig. 1.3, is that since the sensors use natural illumination of the sun, in the visible and infrared bands, to illuminate the scene, then cloud (or smoke) can often obscure the scene making analysis impossible.

1.2.2 Synthetic aperture radar (SAR) images

Alternatively, airborne or spaceborne SAR systems are particularly attractive because they are not affected by atmospheric conditions or the degree of light present. The main difference between the SAR imaging system and the other conventional satellite imaging systems of LANDSAT or SPOT is that SAR is an active system, *i.e.,* it produces its own illumination. The conventional imaging systems are passive and require an external illumination source, *i.e.,* the sun [198]. The SAR system transmits radar waves to illuminate the scene. These radar waves can penetrate cloud cover and are generally unaffected by atmospheric conditions. The images produced by the SAR system are consistent irrespective of the weather or the amount of sunlight present. Comparative analysis may then be performed between two images of the same site over any time frame. With their ability to 'cut' through cloud, SAR systems are particularly suitable for monitoring rainforest and glacial regions – both of particular significance to the global environment [62].

However, radar waves have a much longer wavelength than natural light and therefore radar imaging systems require a larger aperture than conventional imaging systems to obtain the same image resolution. Unfortunately the size of the antenna required to obtain this type of resolution is physically infeasible. The SAR system overcomes this problem by digitally creating the response for a large pseudo antenna from the multiple responses of a small moving antenna [29, 69]. Theoretically, the response from almost any antenna size can be artificially created. This means in turn almost any resolution (down to half a wavelength) is possible, but in practice the resolution is limited by hardware constraints.

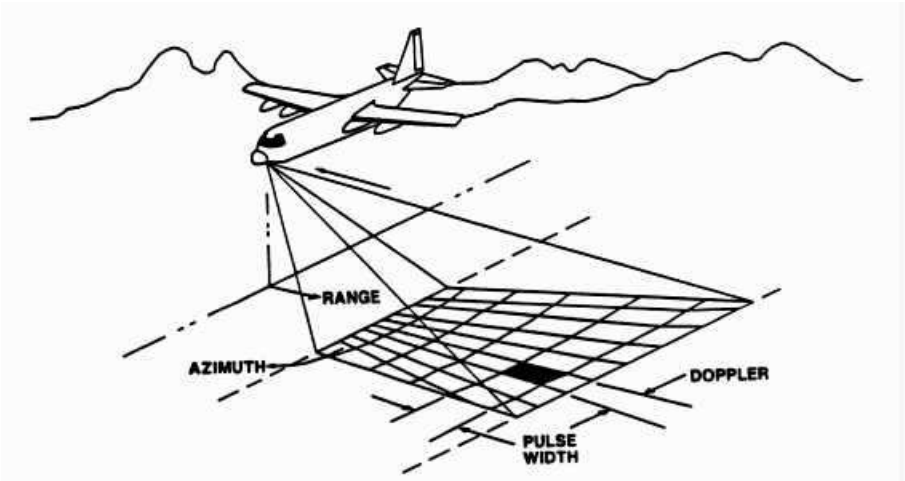


Figure 1.4: Airborne SAR imaging. The Doppler shift in the radar response disseminates azimuth. The delay time disseminates range. This diagram appears courtesy of the Sandia National Laboratories [126].

In order for the SAR system to disseminate range and azimuth information from the returned radar signal, the system needs to transmit a coherent radar signal. This allows range to be determined by the delay of the returned signal. Azimuth information is obtained from the variation in Doppler shift of the returned signal. An image is then formed by recording the intensity of the returned signal for each range and azimuth position as illustrated in Fig. 1.4. Unfortunately, as a consequence of using a coherent signal, the intensity of the signal is corrupted by noise called "speckle." This noise is caused by the coherent signal adding constructively and destructively from many random phases. This randomness is independent of range or azimuth, and is present on all types of terrain including otherwise uniform terrain. The characteristics of the speckle are well documented, Goodman [87].

1.2.3 Terrain mapping

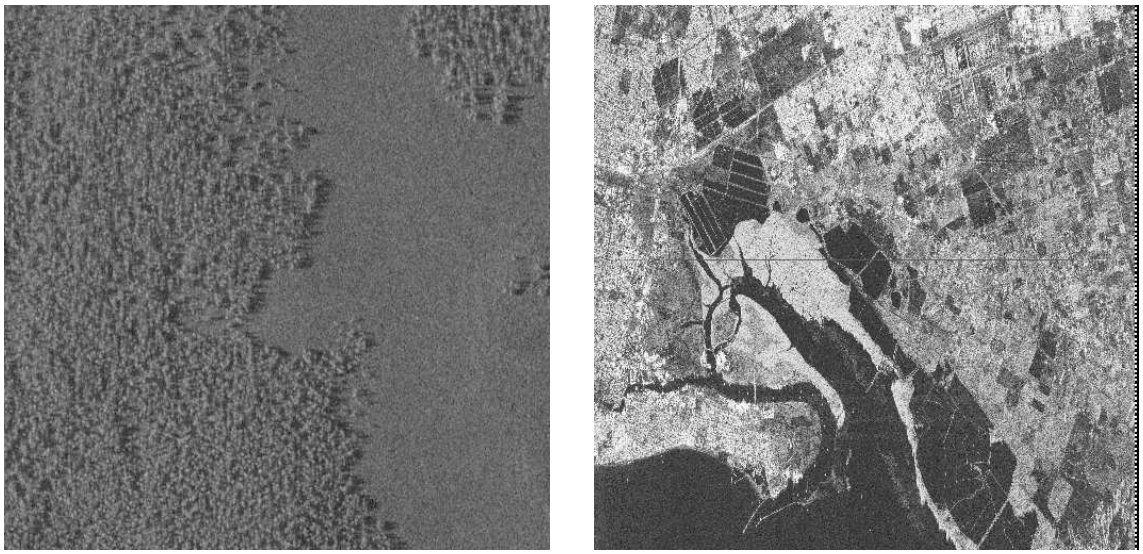
The terrain identification process, as used on the LANDSAT and SPOT images, is not as easily implemented on the SAR images. SAR imaging systems usually only have the capacity to retrieve images at a limited number of frequency bands, but the number of independent terrain discriminating parameters can be increased at least three-fold with polarisation of the pulses (HH, VV, HV, or VH) [73, 173]. Because of the lack spectral information in SAR images, texture offers the best possibility for terrain classification.

A major consideration when using SAR imagery for terrain classification is the presence of speckle noise, which creates a wide variety of pixel intensities from within any single homogeneous terrain region [198]. This results in a greater potential for misclassification of terrain type based on the spectral-polarmetric response for a single pixel [185]. Speckle reduction methods have been used to reduce the problem, but results reported suggest limited success [185].

Speckle corruption is not the hindrance one would first presume; on the contrary, speckle itself has been used to segment a SAR image into a few different terrain types [10, 58, 199]. This requires collecting the statistics of the spectral-polarmetric responses for multiple pixels. Garside and Oliver [77] went further and suggested that speckle is not noise and that its higher order spatial statistics are linked to the local terrain scattering mechanisms. For a homogeneous terrain region, such statistics may be captured from the corresponding spatial statistics of the pixels. All this suggests that classification by texture recognition should be more robust to speckle than the standard spectral-polarmetric pattern recognition approach for a single pixel.

1.2.4 Research purpose

It is evident from inspecting different SAR images, as in Fig. 1.5, that certain types of terrain have unique textural patterns associated with them. It is hoped that texture analysis of these images will reveal that these terrain types lend themselves to texture modelling, where each individual terrain type may be associated with a unique model. In this case the terrain types may be automatically segmented and classified from the SAR image, thereby producing a terrain map.



(a) Cultana, Australia (b) Adelaide, Australia

Figure 1.5: Airborne SAR images show the possibility of terrain identification through texture analysis. These images appear courtesy of Project Ingara, Microwave Radar Division, Electronics and Surveillance Research Laboratory at DSTO, Australia.

1.3 Modelling texture

In the image processing literature, texture is usually defined in terms of the spatial interactions between pixel values. The aim of texture analysis is to capture the visual characteristics of texture in an analytical form by mathematically modelling these spatial interactions [95]. If these spatial characteristics are uniquely modelled, then different examples of textures from one source (population) can be associated analytically, and discriminated from other textures. This allows segmentation of an image into its various textural components, with each component being classified according to how well it fits the mathematical model of a particular texture. This approach requires the number and type of textures to be known in advance. That is, a set of training textures are used to formalise the criteria by which the texture models become unique from each other, but not necessarily unique from any other textures not included in the training set [27, 40, 38, 106]. These conventional models need only capture enough textural characteristics to classify the textures in the training set, via discriminant analysis [60]. This approach is adequate if the image undergoing texture segmentation and classification is known to contain only those textures from the training set.

Irrespective of the approach used, the problem with using a model to define a texture is in determining when the model has captured all the significant visual characteristics of that texture. The conventional method is to use the models to actually classify a number of textures, the idea being to heuristically increase (or decrease) the model complexity until the textures in the training set can be successfully classified. As an example, Haralick *et al.* [100] defined 14 second order texture features extracted from a grey level co-occurrence matrix (GLCM). Heuristic modelling would consist of selecting the most discriminatory of these features via a feature selection process, such as linear discriminant analysis.

A hindrance to universal adoption of such methods is the fact that not only do they require a predefined set of training textures, but the model construction itself is dependent on the training set. The models are heuristically constructed so as to distinguish each texture in the training set from any other. The limitation of this approach is that given a new image, the segmentation process based on these models may not guarantee the performance desired, especially if an unmodelled texture is present (*i.e.,* a texture that is not included in the training set). There is then the likely possibility that the features previously selected for modelling the textures may *not* distinguish the new texture from all of the textures in the training set. In which case, the feature selection process has to be repeated for all textures in the training set plus the new texture (*i.e.,* all models have to be reconstructed). It therefore becomes very difficult to associate any degree of confidence with the performance of a segmentation process based on such models. One of the better approaches to this type of segmentation problem is the one presented by Geman and Geman *et al.* [80], which applied constrained boundary detection to unsupervised segmentation, which aided the discrimination of separate textures in the image.

1.3.1 Ideal texture model

If a texture is to be recognised in a scene containing previously unseen textures, then a new approach is required. The texture models need to capture more than just the characteristics required to distinguish one texture from other known textures – they need to capture all the unique characteristics of that texture. Then, when segmenting and classifying an image, the texture models would be used to determine the probability of a segmented area as having the same unique characteristics as a modelled texture. This being done without having to measure the probability against other possible textures. This would solve the problem of previously unseen

textures being present in the image. Images susceptible to this type of texture classification problem are those obtained via Synthetic Aperture Radar (SAR) from satellites, or airborne SAR, where the images are of the Earth's terrain. These types of images are unlikely to contain only known textures but are more likely to contain a myriad of possible textures. It would be unreasonable to expect a conventional texture classification scheme to have previously identified and modelled all the different types of textures possibly present in the images of the Earth's surface.

An ideal texture model is one that can fully characterises a particular texture, hence it should be possible to reproduce the texture from such a model. If this could be done, we would have evidence that the model has indeed captured the full underlying mathematical description of the texture. The texture would then be uniquely characterised by the structure of the model and the set of parameters used to describe the texture.

1.3.2 Testing the ideal texture model

To date ideal models that describe the full characteristics of a texture have only been found for the very basic of textures (like a checkerboard) Fig. 1.2(b). Several texture models have been investigated [95], but these models are based on rather strict assumptions, and if these assumptions do not hold for a particular texture, then it can not be "ideally" modelled. The problem then becomes: how do we know if a model is ideal for a particular texture; or how do we know whether certain assumptions can be made for a particular texture? What is required is a way of testing the model to determine whether or not it is ideal. Unfortunately, because there is no concise mathematical way of describing texture in general, the only means for testing these models are:

1. Build a totally general model, one that does not require any assumptions, and then see if the ideal model with its specific assumptions can be derived from this general model by fitting the general model to the texture of interest.

2. Synthesise textures from the model and subjectively comparing the similarity of the synthesised textures to the training texture.

The second test is the only practical option. Even if the first test were used, the general model itself would need to be verified via the second test. The second test

requires a subjective analysis, therefore accuracy of the test can not be guaranteed. To completely verify the accuracy though analytical analysis would again require the use of a general model. However reasonable verification is possible through a combination of subjective and analytical analysis. Because we can only obtain reasonable verification through the synthesis of textures from a model, this leads to another problem: how do we verify that the particular texture we are interested in is the only texture class the model encompasses and how big is this class? An ideal model will encompass only the one texture class.

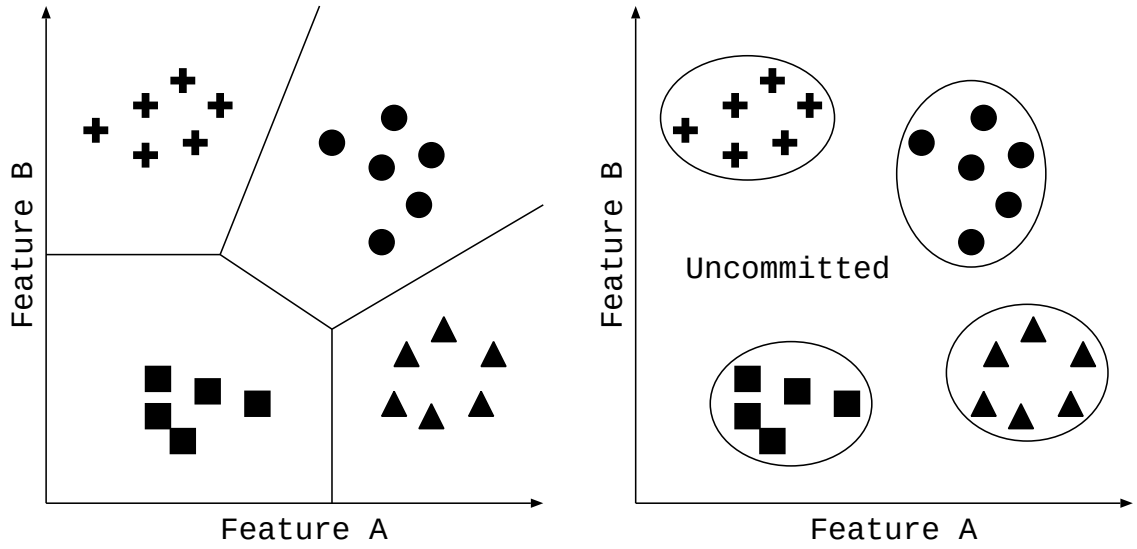
1.3.3 Open-ended classifier

We introduce *open-ended texture classification*, a process whereby an image is segmented into regions of homogeneous texture, but where the region is only classified if it is similar to a predefined texture type. This is unlike supervised classification, where every region has to be classified, even when the region is not even remotely similar to any of the textures with in the training set. Open-ended texture classification is also unlike unsupervised texture segmentation/classification [117, 157, 168], which segments an image into regions of homogeneous texture, but then just classifies these regions with arbitrary labels.

We envisage an open-ended texture classifier to be used in image analysis applications where little prior knowledge exists of the background texture, but where specific types of textures are important. The classifier would be used to train on the specific types of textures and segment these textures from the unknown background. For such a classifier, an ideal model would be required.

The conventional N class classifier, as used for supervised classification [60] and illustrated in Fig. 1.6(a), is based on segmenting the feature space into N distinct domains or classes. These domains are determined by the inter-relationships between classes. Typically, class boundaries are established through Bayes decision theory [60]. This results in every point in the feature space being designated a particular class. The inclusion of more training classes would undermine these designations, requiring the classification boundaries to be completely reconfigured. Therefore the conventional N class classifier is a closed classifier because all information as to the number and type of textures classes must be prior known so as to correctly determine the classification boundaries.

On the other hand, the open-ended classifier, as illustrated in Fig. 1.6(b), does



(a) Conventional N class classifier (b) open-ended classifier

Figure 1.6: Opened and closed classifiers: (a) The conventional N class classifier is a closed classifier since further input of training data would require a complete reconfiguration of the designated classes. (b) open-ended classifier is open-ended because further input of training data would not destroy the overall configuration of the designated classes, but would instead enhance individual classes.

not require all the texture classes to be prior known. This classifier is based on the assumption that the feature space is complete, in that any new class would occupy its own distinct region in this feature space. The classifier is also based on the assumption that each class can be represented by an ideal model, so that not only does each model fully characterise the class, but the model may be used to give a precise likelihood that any new data is of the same class. In this way, inter-relationships between classes do not need to be defined, and therefore classification boundaries do not need to be constructed. Instead an ideal model is used to define the intra-relationships of the training data within each class. When new data is presented to the classifier, one of the ideal models may determine there is a high likelihood that that data is from the same class. On the other hand, if all the ideal models determine that there is a low likelihood that the data is from their class, then the data is defined as being from an unknown class. While the closed classifier can not correctly classify data from an unknown class, the open-ended classifier can. That is the open-ended classifier is "open" to unknown classes.

open-ended texture classification is achieved through the direct use of an ideal texture model. Not only that, but the feature space on which the classifier is based

needs to be complete. Both criteria can be tested via synthesis. If the most likely texture synthesised by the model is one subjectively similar to the training texture, then not only is the model ideal, but the feature space on which it resides must also be complete.

The goal of this research is to produce a method by which a human operator may be able to take a radar satellite image, segment a small portion from the image where the terrain is known, and use this as the training texture to an ideal texture model. Then, with respect to the texture model, find other similar terrain types within the image, independent of the background textures. Such a method of open-ended texture classification would be ideal for terrain mapping of Synthetic Aperture Radar (SAR) images [104], as it would not require a complete library of textures as needed for conventional models using supervised classification. With such a method, any operator would be able choose the type of terrain they wished to segment and classify from any arbitrary SAR image, without the need to supervise the classification by modelling all texture types within the image. Open-ended texture classification therefore lends itself towards the automation of the process, which would be specially usefully if a large database of images needed to be analysed.

1.4 Building a texture model

The previous sections outline various problems associated with modelling texture. The problems derive from the inconcise nature of texture, resulting in texture analysis techniques requiring additional underlying assumptions or being *ad-hoc*. The one underlying factor in these texture analysis techniques is the requirement that the texture models need to be able to distinguish between different textures. Conventional techniques have relied on the number and type of textures being a prior known. We would like to move beyond this restriction without jeopardising the integrity of the model. Therefore we need to build a model that can take one texture and capture its full underlying characteristics.

1.4.1 Determining structure of the model

Unfortunately, with the present knowledge of texture, obtaining a model that captures *all* the unique characteristics specific to a particular texture is an open problem [78]. Texture is not fully understood, and therefore, what constitutes the unique

characteristics has not been defined. However, a reasonable way to test whether a model has captured all the unique characteristics is to use the same model to synthesise the texture and subjectively judge the similarity of the synthetic texture to the original.

Conventional texture models, like the auto-models [17], autoregressive (AR) models [37], moving average (MA) models [95], or combination of both (ARMA) models [115], have not been found to provide a basis for realistically synthesising natural textures [95]. Although Bader, JáJá and Chellappa [6] did achieve modest results with a GMRF model. However recent advances in texture synthesis have produced models that are capable of synthesising natural textures [28], textures that contain both structural and statistical elements. These models are based on the stochastic modelling of various multi-resolution filter responses [23, 103, 149, 211], but they do not use third or higher order statistics, and it is undetermined whether the chosen filters are globally optimal for all textures. Julesz [110] suggested there was textural information in the higher order statistics, and Gagalowicz *et al.* [75] used third order statistics to generate some natural textures. Popat and Picard [164] successfully used a high-order, causal, nonparametric, multiscale MRF model to synthesise some structured natural textures.

Although the synthesis test may indicate if a model has captured the specific characteristics of a texture, it does not determine whether the model is suitable for segmentation and classification. Based on Zhu, Wu and Mumford's philosophy [211], a texture model should maximise its entropy while retaining the unique characteristics of the texture. The principle behind this philosophy is that a texture model should only model known characteristics of a texture and no more. The model should remain completely noncommittal towards any characteristics that are not part of the observed texture. Zhu, Wu, and Mumford [211] used this philosophy to build their *minimax* model, which was designed to obtain low entropy for characteristics seen in the texture while maintaining high entropy for the rest, thereby sustaining a model that infers little information about unseen characteristics. This minimax entropy philosophy is equivalent to reducing the statistical order of a model while retaining the integrity of the respective synthesised textures.

1.4.2 Fitting the model to the texture

Depending on the statistical order of the model, its parameterisation requires a certain sized area of homogeneous texture. Although it would be informative to fit models of increasing statistical order, there is a limit to the statistical order which may be successfully modelled. The statistical order defines a multi-dimensional space over which sample data resides. The density estimate that forms from the placement of the sample data in this multi-dimensional space describes characteristics of the texture. The model captures these characteristics by conforming the density function defined by the model to the density estimate defined by the sample data. Increasing the statistical order of the model increases the number of dimensions over which this density is defined. The modelling of high statistical order models is limited by the *curse of dimensionality*, which occurs when the respective high-dimensional space becomes infeasible large for modelling [11]. Silverman [183] showed that to maintain reasonable accuracy in fitting the model, the amount of sample data needs to grow almost exponentially with the dimensionality of the space. As we are dealing with a limited amount of sample data, approximately equal to the size of the homogeneous textured area, the accuracy of the model will rapidly decrease as the dimensionality of the space increases.

An estimate of the size of the data set required to build a model of a particular statistical order is shown in Table 1.1. The table gives the number of sample data required to fit a density function with respect to the dimensionality of the space in which the density resides. It is assumed that the underlying density function is multivariate normal, and is of interest to estimate the mean of the multivariate normal with a relative mean square error of less than 0.1.

These results are likely to be optimistic for estimating a density function derived from a texture. Not only did Silverman predefine the density function, but only the mean of a multivariate normal was fitted. To highlight why the sample size increases so rapidly; given a ten-dimensional normal distribution, 99% of the mass of the distribution is at points whose distance from the mean is greater than 1.6 standard deviations. In the one-dimensional case, nearly 90% of the distribution lies between ± 1.6 standard deviations. Thus it is likely to be difficult to fit a high-dimensional density function without an enormous number of data samples. Even if it was possible to fit a density function to the sample data, it still may give a superficially false impression of the likely behaviour of sample data. Unfortunately if the tail of the density is eliminated altogether by considering a uniform distribution

Table 1.1: Sample size required to estimate a standard multivariate normal density at the mean for an error of less than 0.1, from Silverman [183]

Dimensionality	*Required sample size*
1	4
2	19
3	67
4	223
5	768
6	2 790
7	10 700
8	43 700
9	187 000
10	842 000

over a box, similar behaviour is still observed [180].

1.5 Our texture model

In this thesis we focus on the Markov Random Field (MRF) as a texture model. The MRF model was chosen because it requires few underlying assumptions to be made about a texture, and it also allows choice over the degree of statistical order applied in the model. A first order statistical model is one that just models the histogram of the texture, *i.e.,* the intensity distribution of single pixels. Second order statistics describes the intensity distribution of pairs of pixels. The Grey Level Co-occurrence Matrix (GLCM) of Haralick *et al.* [100] is a typical example of second order statistics. The parametric versions of the MRF model mainly use just second order statistics and are referred to as auto-models by Besag [17]. One of the most commonly used auto-models is the Gaussian MRF (GMRF) model [39]. This is because of its generality and its ability to require only an FFT to synthesise texture instead of a long iteration process [59]. However, the GMRF has been unable to realistically model natural textures from the Brodatz album [6, 95]. It has been hypothesised that to model such structured textures third, or higher, order statistics are required [75, 111]. We show a model that is capable of modelling these high order statistical properties is the nonparametric MRF model.

1.5.1 Synthesising realistic examples of texture

In this thesis, we present a noncausal, nonparametric, multiscale, Markov Random Field (MRF) model which can be trained from a small portion of a natural texture. We have also developed a technique for synthesising a texture image from the model. The synthesis process is assisted with the incorporation of our novel pixel temperature function into a multiscale synthesis algorithm [86]. The pixel temperature function acts to perform *local annealing*.

By subjectively comparing the synthesised texture with the training texture, we are able to establish the adequacy of the model. The power of our modelling technique is evident in that only a small image of either stochastic or structured texture is required to produce a model capable of synthesising representative examples of the training texture, even when the training texture contains long range characteristics. This unique model may then be used to identify other examples of the training texture from any other texture, and not just those textures contained in the training set.

We find that the complexity of the model can be adjusted to find the minimum order necessary for the synthesised texture to look representative of the training texture. If the complexity is reduced below this order, the synthesised texture becomes dissimilar to the training texture. If the complexity is increased above this order, the synthesised texture remains fully representative, but the higher order complicates the segmentation and classification process for which we have designed the model.

1.5.2 Open-ended classification of texture

In this thesis, we also present a method for reducing the statistical order of the nonparametric MRF model to a set of lower order properties based on the *cliques* of the MRF. We show that this reduced model still contains the specific characteristics required for synthesising representative texture, but due to the compact statistics, is able to perform better at segmentation and classification. By judicious reduction of the statistical order, the model may be optimised to capture the most unique characteristics while retaining the integrity of the synthesised textures, thereby producing a model suitable for open-ended classification of texture in an image. This second model is based on Moussouris' strong MRF model [148]. In this thesis we show that Moussouris' strong MRF model is equivalent to the Analysis-of-variance

(ANOVA) model [67], which allows us to use the theorems of the ANOVA model for estimating the strong MRF model.

For the open-ended classification of texture, we use our strong MRF model to find the lowest order statistics that may be used to uniquely represent the texture. The model is then used to collect a sample of statistics from an image segment we wish to classify. This sample of statistics is compared to a sample of statistics obtainable from the training texture. The classification is made on the basis of how confident we are that the two sets of statistics are from the same distribution. Our open-ended texture classification results are presented in the form of a probability map showing the associated goodness-of-fit for each pixel and its surrounding area being classed as being from the same model that describes the training texture.

1.6 Outline of thesis

Chapter 2 gives a broad overview of texture models and how they have been used in the field of digital image processing. In this chapter we outline some of the reasons why, out of all these models, we have chosen to direct our attention towards the nonparametric MRF model. In Chapter 3 we look more closely at the MRF model and give some background theory on the model. In particular we reiterate some interesting results that demonstrate the equivalence between the MRF model and the Gibbs distribution. In subsequent chapters we employ these results to the development of the strong nonparametric MRF model.

Chapter 4 reviews the development of the parametric MRF model. It includes various standard models and a section on how the parameters for these models are estimated. It also looks at how these models are applied in synthesising, segmenting and classifying textures.

Chapter 5 gives the construction details of our nonparametric MRF model. In this chapter we also look at alternative approaches to building a nonparametric MRF model. In Chapter 6 we incorporate the theory from the MRF-Gibbs distribution equivalence into our own theory that elegantly demonstrates the equivalence between the strong MRF model and the ANOVA log-linear construction. From this proof we are able to derive the general ANOVA log-linear construction formula. To our knowledge, this relationship has not been demonstrated before.

Chapter 7 details our own multiscale texture synthesis algorithm incorporating

our novel pixel temperature function. Other variations of this synthesis algorithm are also outlined. From this chapter we have published four papers showing the merits of the nonparametric and strong nonparametric MRF models [152, 155, 153, 156]. Of particular interest is the texture synthesis by the strong nonparametric MRF model, which identified textures that could be completely represented by just third order statistics [153].

Chapter 8 outlines the common method for supervised classification. Because of the problems we see for supervised classification in the area of terrain recognition for SAR images, we introduce our own method for texture classification that does not use supervised classification. From the results of this chapter we have published another three papers demonstrating the use of our open-ended texture classification algorithm, and in particular the advantage of the added functionality of the strong nonparametric MRF model [104, 155, 158].

Finally Chapter 9 gives a short summary and conclusion. Possible future research to cover any short comings in the application of our algorithms are also discussed. There are also three appendices. The first one Appendix A, is taken from our published paper "Extracting the cliques from a neighbourhood system" [154]. The other two appendices show more results.

Chapter 2

Background

Research into texture models seeks to find a compact, and if possible a complete, representation of the textures commonly seen in images. The objective is to use these models for such tasks as texture classification, segmenting the parts of an image with different textures, or detecting flaws or anomalies in textures.

Such research tends to focus on the key issues of feature selection, model selection, parameter estimation, image sampling and goodness-of-fit. One recognised problem is that many of the existing models only conform to specific types of natural texture. A simple model, with the power to represent all textures has not been reported. In this thesis we develop a model, based on the MRF, which can indeed capture a very wide range of textures, perhaps all textures. The only constraint is that the textures should be statistically stationary over the spatial extent of the image. We demonstrate this model is able to represent all the texture it was tested on by using the model to generate new texture images. These had the same visual characteristics as the one from which the model was developed.

2.1 Texture models

In order to define a texture model, one needs to assume something about the process that created the texture. The models grouped as stochastic texture models are designed on the assumption that the texture is a realisation of a stochastic process on a random field. These models attempt to find the joint distribution over this random field, with samples from the joint distribution being representative of the texture of concern. Dubes and Jain [59] provided a taxonomy of the various different

models used in describing a stochastic texture (see Fig. 2.1). Each model has its own unique definition of how the local sites in the random field are related. These local interactions are modelled to form a joint distribution.

2.1.1 Taxonomy

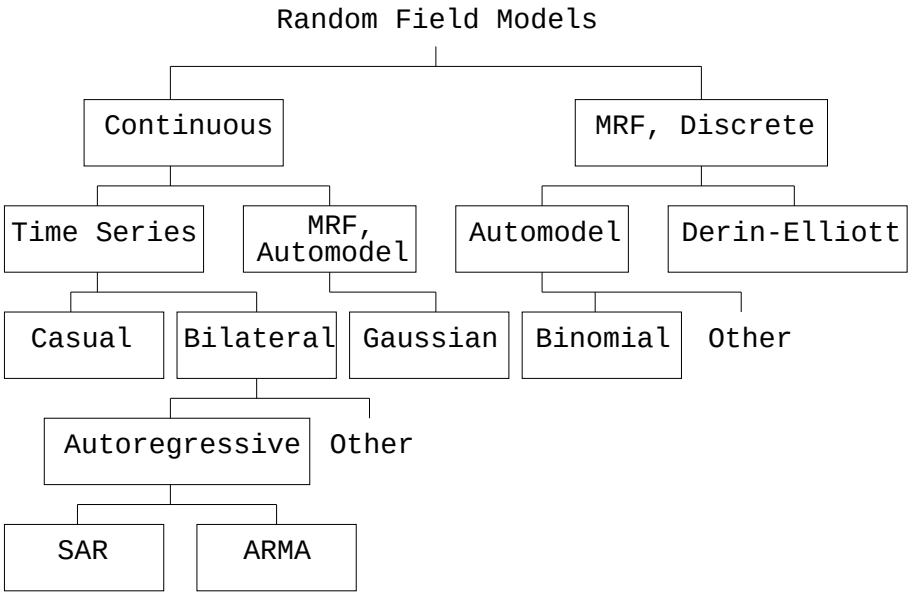


Figure 2.1: Taxonomy of image models courtesy of Dubes and Jain [59].

Gaussian models

Digital images, as used in image processing, are defined as lattices of discrete intensity, therefore to model such images as a continuous process is an artificial abstraction, nevertheless it does allow for some convenient processing to occur. A texture realisation from a Gaussian model can be obtained reasonably quickly via the FFT (Fast Fourier Transform) [59]. However, the Gaussian model imposes restrictions on the texture it models. The Gaussian model requires all interactions between pixels to be Gaussian distributed, it also only allows multiple pairwise interactions to occur [39, 210].

Autoregressive models

Autoregressive (AR) models, or simultaneous autoregressive (SAR) models, come from the family of simultaneous models. These are 2D models derived from 1D time series models [113, 114, 115]. While 1D time series have a past and present, an equivalent ordering does not exist on the 2D discrete lattice, instead some sort of ordering has to be imposed on the lattice. Other such models are simultaneous moving average (SMA) and simultaneous autoregressive and moving average (SARMA) models [95]. For every SAR model there exists a unique conditional MRF model, but the converse is not true, except for the Gaussian case [17]. The advantage of the SAR model is that it is generally characterised by fewer parameters than its equivalent MRF model. As a note, SMA and SARMA models do not have equivalent MRF models. However that does not mean that an MRF model can not model a texture generated by SMA or SARMA. Chellappa and Kashyap [37] demonstrated the use of noncausal autoregressive models for the generation of textures.

Markov random field models

Markov random fields (MRFs) are popular for modelling images. They are able to capture the local spatial textural information in an image. These models assume that the intensity at each pixel in the image depends on the intensities of only the neighbouring pixels. Theoretically they should be able to model any homogeneous texture, unfortunately there are many practical barriers that make correct estimation of the MRF very difficult. To make the estimation process a little easier it is common to constrain an MRF model as an auto-model, which only contains pairwise interactions. There are still many variations of the auto-model [17], *e.g.,* Derin-Elliot model [54], auto-normal model [40, 48], and the auto-binomial model [17, 50]. These models were able to capture micro-textures well, but failed with regular and inhomogeneous textures. MRF models have been applied to various image processing applications such as texture synthesis [50], texture classification [40, 120], image segmentation [48, 194], image restoration [82], and image compression.

Spatial domain filtering

Spatial domain filters are the most direct way to capture image texture properties. Early attempts concentrated on measuring the density of edges in an image. Fine

textures tended towards a higher density than those of the coarser textures. The Robert's operator and the Laplacian operator [85, 130] are classical edge detectors. Malik and Perona [136] used differences of offset Gaussian (DOOG) functions with a nonlinearity function to model the pre-attentive texture perception of the human visual system. The nonlinearity was introduced to discriminate between textures with the same mean and second order statistics. Their method works well for both synthetic textures and natural textures. Similar work was done by Unser and Eden [200] who also used spatial filters in conjunction with a nonlinear operator. A review of a number of spatial domain and spatial frequency techniques is presented by Reed and Wechsler [170].

Alternatively, spatial moments [130] may be used as spatial filters. From these spatial filters a set of filtered images may be obtained, which may then be used as texture features. Tuceryan [195] did this to successfully use moment-based features for texture segmentation.

Fourier domain filtering

From psychological research it was determined that the human visual system uses orientated frequency components to decompose an image for textural analysis [32]. Frequency analysis of a textured image can be achieved by applying an FFT to the image and analysing the resulting image of the respective Fourier domain. Band pass filtering leads to multi-resolution analysis of the image. Coggins and Jain [47] used a set of frequency and orientation selective filters in a multi-band filtering approach. They were able to successfully segment and classify a variety of natural and synthetic textures. Smith [186] uses a set of band pass filters followed by zero crossing detection to successfully generate a tree classifier of textures.

Gabor and wavelet models

The Fourier transform of a whole image may not be appropriate for some applications, in which case spatially localised analysis of the frequency content may be more appropriate. To accomplish this, a windowing function is introduced into the frequency analysis. This is applied by the window Fourier transform, defined for a one-dimensional signal $f(x)$ as:

$$F_w(\mu, \xi) = \int_{-\infty}^{\infty} f(x) w(x - \xi) e^{-j2\pi\mu x} dx \tag{2.1}$$

When the window function $w(x)$ is Gaussian, the transform becomes a Gabor transform.

A wavelet transform is also a window Fourier transform, but it uses a set of windowing functions that increase with frequency. This is done to increase the resolution in the time (or space) domain as the central frequency of the band pass filter is increased. The wavelet transform can be written as:

$$W_{f,a}(\mu,\xi) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} f(x) h\left(\frac{x-\xi}{a}\right) e^{-j2\pi\mu x} dx \tag{2.2}$$

The use of 2D Gabor filters for modelling texture was proposed by Turner [197] and Clark *et al.* [45] after Daugman [52] had proposed the use of Gabor filters for the modelling of simple cells in the visual cortex of some animals. Later Gabor filters were successfully used in segmentation and classification of textured images by Farrokhnia and Jain [65, 109].

Fractal models

Many natural surfaces have a statistical quality of roughness and self-similarity at different scales. Fractals are very useful in modelling these image properties. Mandelbrot [137] proposed fractal geometry and was the first to notice its existence in nature. Fractals are further explained by Barnsley [7] and Feder [66], and a short introduction to fractal models is given by Haindl [95]. These models are independent of scale, display self-similarity and are able to model some natural textures like clouds, leaves and rivers [119, 118]. Fractal models are also used in segmentation [76, 159, 160] by fitting a fractal function to local regions within an image and segmenting the image according to the parameters associated with each local fractal function. Some fractal texture segmentation techniques are given by Chaudhuri *et al.* [35], Mosquera *et al.* [147]. Fractal models have also been used for discriminating textures [76].

2.1.2 Applications for texture models

There are many applications for texture analysis. They range from automated inspection [57, 107, 182], medical image processing [41, 127, 135], document processing [108, 109, 192], and of particular interest to us: remote sensing.

Texture analysis has been extensively used for land classification of remotely sensed images. Here the different terrain types can be identified by modelling their respective texture. Haralick *et al.* [100] used grey level co-occurrence features to identify differences in the texture of various terrain types. For a seven-class classification problem, they obtained approximately 80% classification accuracy. Rignot and Kwok [174] also used texture features from the co-occurrence matrix, but they applied them to SAR images pre-processed with speckle filters. Schistad and Jain [178] compared various texture features for analysing SAR images. These included fractal dimension, autoregressive Markov random field model, and grey level co-occurrence texture features. Classification errors ranged from 25% for the fractal based models to 6% for the MRF features. On the other hand, Du [58] and Lee and Philpot [131] used local spectral texture features to segment SAR images, in particular Du [58] used Gabor filters to successfully segment SAR images into water covered regions, new forming ice, older ice and multi-year ice.

2.1.3 Performance of texture models

Ohanian and Dubes [151] studied the performance of various texture features under the guidelines of detecting which features were optimal for classification. Out of four fractal features, 16 co-occurrence features, four auto-binomial Markov random field features, and 16 Gabor features, they found the co-occurrence features gave the best classification rate at 88%, followed by fractal features at 84%. The MRF and Gabor features achieved only 65% classification rate. However when they combined the fractal features with the co-occurrence features they obtained a classification rate of 91%.

This does not suggest that co-occurrence features are the single most important features. For one, it is not possible to reconstruct texture from the features, therefore it can not be demonstrated that the features capture all the characteristics of texture. However, what is more at odds with Ohanian and Dubes [151] findings, are the results obtained by Smith [187] when he repeated the experiment. Smith [187] carried out the experiment over the same data set and obtained the results shown in Table 2.1. These results differ significantly from those of Ohanian and Dubes, with the classification rates being generally higher than Ohanian and Dubes best results. Smith [187] notes that the models were not implemented exactly as by Ohanian and Dubes, but with functionality common within the literature [40, 70].

Table 2.1: MeasTex winner-take-all (percentage correct) scores

Test Suite	*Gabor*	*GLCM*	*Fractal*	*GMRF*
all16	0.9360	0.8115	0.7256	0.9688
fractal	0.9414	0.5840	0.7188	1.0000
leather	0.8379	0.7363	0.4844	0.8848
mrf	0.9980	0.9961	0.9551	0.9980
paint	0.9668	0.9355	0.8477	0.9941
SUMMARY	0.9360	0.8127	0.7463	0.9691

These results indicate the care that must be taken when accepting comparison results. The same texture images and the same paradigms (albeit different implementations) were used, but significantly different results were obtained. In fact, if these models were to be ranked based on Smith's [187] results we would have (best to worse) Gaussian MRF (GMRF), Gabor Energy, GLCM, and Fractal Dimension. This is in stark contrast to rankings based on Ohanian and Dubes [151] findings of GLCM, Fractal, GMRF and Gabor Energy.

Table 2.2 gives the MeasTex [187] test suite scores for the implementations of GLCM, Gabor Energy and GMRF methods as described above. Two patterns emerge from this table. On microtextures, the algorithms are ranked thus: GMRF, Gabor Energy and GLCM. On macrotextures, the algorithms are ranked thus: Gabor Energy, GMRF and GLCM. Natural textures (Brodatz, grass, material, and VisTex [203]), although containing a mix of macro- and microtextures they are predominantly microtextures.

DeBonet and Viola[25] used a nonparametric multiscale statistical model to achieve 100% correct classification on the MeasTex Brodatz test suite [187]. They were also able to obtain 98% correct classification on the MeasTex Lattice test suite [187]. This is a substantial improvement on the results shown in Table 2.2.

2.1.4 Accuracy of texture models

Our model based texture analysis methods are constructed on the premise that not only can they be used to describe texture, but they can be used to synthesise texture as well. As outlined in Section 2.1.1, there are many different models on which to base a synthesis algorithm, and many attempts to demonstrate

Table 2.2: MeasTex test suite summary scores

Test Suite	*GLCM*	*Gabor*	*GMRF*
bomb	0.8406	0.8460	0.9446
bombRot	0.6829	0.9248	0.9603
Brodatz	0.9239	0.9451	0.9713
grass	0.9162	0.8900	0.9483
material	0.9650	0.9678	0.9797
VisTex	0.8523	0.9066	0.9355
lattice	0.6953	0.8919	0.7396
latticeRot	0.6643	1.0000	0.9647
mortar	0.7150	0.8758	0.7551
mortarRot	0.6055	0.9921	0.9626
mortarRotS	0.6248	1.0000	0.9763

adequate performance. These models have ranged from the fractal [159], autoregressive (AR) [36, 37, 53], moving average (MA) [95], autoregressive moving average (ARMA) [115], Markov [101], auto-binomial MRF [1, 50], auto-normal MRF [40, 48], Derin-Elliott [54], Ising [121, 161, 189], and log-SAR model [72] which was used to synthesise synthetic aperture radar images. A summary of these texture synthesis algorithms is provided by Haindl [95], Haralick [99], and Tuceryan and Jain [196]. These synthesis algorithms have been successful in modelling microtextures, but they have been less than successful in modelling macrotextures. As natural textures [28] consist of a mix of macro– and microtextures, we can surmise that these models are less than optimal for modelling such textures. This also implies that these models have not been able to capture all the characteristics of natural textures, and therefore can not be regarded as "ideal" texture models for the purpose of open-ended texture classification.

Recent advances in texture synthesis have produced models that are capable of synthesising natural textures, textures that contain both structural and statistical elements (*i.e.,* a mix of macro– and microtextures). These models are based on the stochastic modelling of various multi-resolution filter responses [23, 103, 149, 211], but they do not use third or higher order statistics, and it is undetermined whether the chosen filters are globally optimal for all textures. Julesz [110] suggested there was textural information in the higher order statistics, and Gagalowicz *et al.* [75] used third order statistics to generate some natural textures. Popat and Picard [164] successfully used a high-order, causal, nonparametric, multiscale MRF model to

synthesise some highly structured natural textures.

2.2 Hypothesis: high order statistics are required to model texture

Texture models contain the high dimensional statistical information of an image. The dimensionality of the statistically information refers to the number of pixels that are inclusively statistically dependent. A one dimensional texture model describes the statistical information for a single pixel. This information is readily captured by the histogram of the image. Two dimensional statistical information describes the intensity distributions of two pixels spatially separated from each other. The GLCM textural features designed by Haralick *et al.* [100] is an example of features that use a set of two dimensional statistical information.

Texture models, such as the auto-models, AR, MA, and ARMA, capture the high dimensional statistical information as a multi-dimensional probability density function (PDF) defined over neighbourhood of spatially separate pixels. Each dimension uniquely represents a particular pixel in this neighbourhood. However, parametric texture models typically compact the multi-dimensional PDF into a set sub-functions. The sub-functions are each defined over a subset of the total number of dimensions, whereby the multi-dimensional PDF is evaluated as a function of the sub-functions.

In this thesis, statistical order is defined with respect to the sub-functions. Given that a model has been broken down into is sub-functions, the statistical order of the texture model is defined as the maximum number of dimensions encompassed by any one of the sub-functions. The texture models previously listed all use sub-functions that are defined over no more than two dimensions (or pixels in the neighbourhood), and are therefore referred to as second order models or auto-models by Besag [17].

Markov random field texture models are defined as a multi-dimensional PDF over a neighbourhood. Parametric Markov random field models evaluate this multi-dimensional PDF as a function of sub-functions called potential functions. Each potential function is defined over a subset of pixels in the neighbourhood. These subsets are known as *cliques* [78]. A second order MRF model is therefore one that contains only single and pairwise cliques, *i.e.,* cliques that contain no more than two pixels.

One of the most commonly used second order models is the GMRF model [39], popular because of its generality and because it requires only three Fast Fourier Transforms (FFTs) [59] instead of a long iteration process to synthesise texture. However the GMRF is unable to realistically model complex natural textures, for instance as seen in the Brodatz album [6, 95]. We hypothesise that to model such natural textures, third or higher order models are required [111].

We have chosen the MRF as the basis of our texture model because of its flexibility in defining the statistical dimensionality and order. However parametric versions of the MRF model mainly use just second order statistics (*i.e.,* auto-models [17]). This is due to parameter estimation problems for MRF models when the statistical order is greater than two [181]. However we show the nonparametric version of the MRF model is capable of modelling these higher order statistical properties.

2.3 Markov random field (MRF) model

The first step of texture analysis is to select a model. We have chosen "Markov Random Field" (MRF) as our model for the reason that it requires only one assumption to be made of the texture. That assumption is that the value of a single pixel is only conditionally dependent on its *neighbouring* pixels. This is how an MRF is defined [82]. In a limiting case, a pixel could be dependent on all other pixels of the same texture, *i.e.,* has a complex spatial structure. The other limiting case being that a pixel is not dependent on any other pixel (in which case the texture can be described by a simple histogram distribution), *i.e.,* has no spatial structure at all. Either way the assumption does not hinder the generality of the model. Theoretically every homogeneous texture can be modelled as an MRF.

For an MRF model to be valid, it must have a valid joint distribution for a random field. The joint distribution defines the probability for a particular realisation of that field. This probability has to be defined such that for any two realisations of the same texture, the probabilities are similar [17].

The joint distribution of an MRF model is calculated from the Local Conditional Probability Density Function (LCPDF) [17], which defines the probability distribution of a pixel being a particular value given the neighbouring pixel values. This is fortuitous, as in general we do not have an ensemble of textured images from which to directly construct the joint distribution. Instead we usually only have the one

image of any one texture. Therefore we use the LCPDF to capture the statistics of the texture and assume they are the general statistics needed to describe all images of the texture.

To build the LCPDF one needs to sample the texture at every pixel position to estimate the LCPDF from the samples [19, 20, 81, 181]. The accuracy of the LCPDF estimate is dependent on the relative sample space size compared with the number of samples [169, 175]. Now the number of samples is approximately equal to the number of pixels in the image. However the sample space, (*i.e.,* the domain over which the LCPDF is to be estimated) is equal to the number of grey levels raised to the power of the number of pixels in the neighbourhood plus one. The smaller the sample space the more reliable the estimate is for a given training texture [175, 183]. If the sample space is too big then the LCPDF estimate will not capture the general statistics of the texture and the joint distribution will not represent the texture [59].

For the LCPDF estimate to capture the general statistics of the texture, the number of neighbouring pixels used in the estimate has to be limited. However this is dependent on the type of function artificially imposed on the texture and used as the LCPDF. If the function is a compact parametric function like the Gaussian function then the number of neighbouring pixels that can be used in the estimate is slightly higher than if the function was completely nonparametric [183], provided the texture fitted the GMRF model. Every constraint imposed on the function to be used to represent the LCPDF [1, 89, 140, 175], increases the number of neighbouring pixels that can be used in the estimate. Of course one seldom has prior knowledge about the texture that allows the LCPDF to be properly constrained. However there are also some unobvious compulsory restrictions on the LCPDF that are needed to make the joint distribution of the MRF legitimate (which will be discussed in Chapter 3). Failer to observe these restrictions will result in problems when trying to use the LCPDF to classify or generate realisations of the MRF [59].

Two new problems now assert themselves:

1. How to find which neighbouring pixels interact with their local pixel and therefore need to be included in the neighbourhood of the LCPDF estimate.

2. How to model the interactions between the neighbouring pixels and the reference pixel.

To date there has not been a conclusive answer to the first problem [78], although

there have been some attempts at defining a cost function to neighbourhood selection [181, 188] based on the stochastic complexity of Rissanen [176]. The problem is that under normal circumstances two or more variables are said to interact if their values are correlated. However the presence of long range correlations in an image is not necessarily evidence of long range pixel interaction, some long range correlations can be fully accounted for by (strong) short range interactions between pixels [121, 161]. In this thesis we introduce a new approach to identifying the lower bound of a neighbourhood.

The second question has been answered with some degree of success [17]. Unfortunately, for the interactions to be adequately modelled, a specific model has to be imposed onto the LCPDF. The most common models used are the auto-models which only allow pairwise interactions to occur. Commonly used auto-models are the GMRF model [39] and the binomial auto-model of Cross and Jain [50].

Previous experiments have not been able to correctly model natural textures with auto-models when structure and a reasonable range of grey levels have been present [1, 6, 59, 95, 105]. In particular Chen and Dubes [42] used a goodness-of-fit test to demonstrate that second-order models are not appropriate for binarised versions of Brodatz (*i.e.,* natural) textures. In this thesis we show that it is possible to model and synthesise these natural textures if more complex interactions between pixels are incorporated into the texture model. These interactions between pixels involve groups of pixels known as *cliques*, where the number of pixels in a clique defines the statistical order of the interaction, and the maximum order defines the statistical order of the MRF model. Like the neighbourhood problem stated in question one, there is again no concise way for determining if an interaction is present between pixels, whether it be third-order, pairwise, or singular (no interaction) [181].

Parametric versions of the MRF model are prohibitive for modelling statistical interactions greater than second-order. This is due to the relatively large number of parameters in the LCPDF that need to be estimated with respect to the low density of sample data in the sample space. The size of this sample space is equal to the number of grey levels raised to the power of the number of dimensions, where the number of dimensions is equal to the number of pixels in the neighbourhood. With high-dimensional LCPDFs, especially when there are many grey levels, the feature space becomes very barren of sample data [11]. This leads to problems when estimating the many parametric sub-functions designed to characterise the interactions in the cliques, although Modestino and Zhang [144] did try to formalise

the sub-functions design. The sparse nature of the sample space is less of a problem for the nonparametric MRF model [164].

2.3.1 Problems in using MRF models

If the texture is a GMRF then there are less problems in modelling it. The neighbourhood size can be adequately selected [116], the model parameters can be efficiently estimated [59, 95], and if the lattice is toroidal a rapid FFT based synthesis algorithm can be used [95]. The GMRF model is the exception of all the MRF models, for the rest, there is little formal methodology for neighbourhood selection [181], model selection is almost arbitrary [105, 181], and parameter estimation can involve a long arduous iterative process, especially if the Markov Chain Monte Carlo method is used [105, 181].

With texture synthesis there is a problem of phase discontinuity when either the Metropolis algorithm [142] or the Gibbs sampler [82] is used to synthesise an image via an iterative process [1]. Phase discontinuity occurs when the parameters of an MRF model have not been properly constrained [181]. Dubes and Jain [59] demonstrated that under such conditions when parameters are specified so that local correlations develop into long range correlations [121, 162], the iterative synthesis algorithm does not converge to the desired texture. Instead one particular grey level can start to dominate the iterative process. To solve this problem Cross and Jain [50] used an exchange algorithm which kept the histogram of the image constant through the synthesis process. However this fix does not properly embrace the sentiment that any realisation should be possible from an MRF.

The iterative process of texture synthesis is not nearly as arduous as that required for texture segmentation. For texture synthesis all that is required is to produce an image with a highly likely representation with respect to the joint probability [78]. On the other hand, texture segmentation is an optimisation problem requiring a search for the maximum likely representation with respect to the joint probability[38]. This is generally achieved through an optimisation algorithm such as simulated annealing [78, 82, 79].

The difficulty with MRF texture classification is that not only does it require prior knowledge of the types of textures present, but the normalising constant of the MRF model may need to be evaluated [59]. This is tractable for GMRF models [34, 40], but difficult for other MRF models [59]. Alternatively, a more practical approach

is the one used by Chen [43] whereby the parameters themselves, estimated from the samples of each pattern class, are used as the feature vectors to a standard maximum likelihood algorithm.

Besides these difficulties, MRF models have demonstrated their ability to model texture [50, 54, 101], for such applications as; image restoration [82], texture segmentation [54, 106], and classification [34, 40]. However there are still issues that need to be further researched, such as the specification of MRF models, modelling noise processes, performance evaluation, parameter estimation, the phase transition phenomenon, and the comparative analysis of alternative procedures [59]. Also there has been little work in the nonparametric estimation of the MRF model [78].

2.4 Nonparametric Markov random field

The nonparametric MRF model is based on estimating the LCPDF by building a multi-dimensional histogram of the homogeneous textured image. The number of dimensions used in the histogram is equal to the number of neighbours in the neighbourhood plus one, which is equal to the statistical order of the model. Although it would be nice to test larger and larger statistical orders of the texture, there is a limit to the order which may be successfully modelled. This is due to the *curse of dimensionality* [11], which occurs when modelling with a limited amount of sample data in a high dimensional space. Silverman [183] showed that to maintain reasonable accuracy in the model, the amount of sample data needs to grow almost exponentially with the dimensionality of the histogram. As we are dealing with a limited amount of sample data from our training image, the accuracy of the model will rapidly decrease as the dimensionality of the histogram increases. In such cases where the sample data is sparsely dispersed over the histogram space, nonparametric estimates of the model will tend to be more reliable than their parametric counterparts. This is because nonparametric estimates only model those areas of the histogram that contain data rather than trying to fit the model to the whole of the histogram space as with parametric estimates.

One nonparametric method of modelling the multi-dimensional histogram is to cluster the histogram data and model each cluster as a standard multi-dimensional Gaussian density. Popat and Picard [164] used precisely this method with great success to produce a nonparametric causal model for texture synthesis. With their method, they were able to model up to 14-dimensional histograms. However, the

method did not extrapolate well to higher dimensional histograms. In this thesis we model the histogram in a similar way, except that we model each point in the histogram with a standard multi-dimensional Gaussian density. This is the Parzen density estimation technique [60].

Texture synthesis is performed via our own multiscale synthesis algorithm incorporating a novel pixel temperature function. As part of the synthesis process we will show that the pixel temperature function initially reduces the dimensionality of the multi-dimensional histogram, thereby alleviating the problem associated with correctly estimating the model for a high-dimensional space. This means we are able to use large multi-dimensional histograms to represent the texture. In fact the model presented in this thesis has been used to synthesise textures using 81-dimensional histograms with gratifying results. For synthesis purposes, we see no theoretical limit to the size of the multi-dimensional histogram that may be used to represent the texture, provided that it can be accommodated for by the training image.

Whilst the nonparametric MRF model just described is shown to allow texture synthesis, it is not suited as it stands to texture segmentation and classification. This is because we need to find the lowest order model which is capable of representing the texture. If the model has an unnecessarily high order, it will only classify textures if unnecessary detail is matched [211]. A balance must be attained so that the statistics contain the unique characteristics of the texture. For this we introduce the *strong* MRF model. With it we can adjust the order of the statistics used to model the texture. This second model is based on Moussouris' strong MRF model [148]. In this thesis we show that Moussouris' strong MRF model is equivalent to the Analysis-of-variance (ANOVA) model [67], which allows us to use the theorems of the ANOVA model [22] for the strong MRF model.

For the open-ended classification of texture, we use our strong MRF model to find the lowest order statistics that may be used to uniquely represent the texture. The model is then used to collect a sample of statistics from an image segment we wish to classify. This sample of statistics is compared to a sample of statistics obtainable from the training texture. The classification is made on the basis of how confident we are that the two sets of statistics are from the same distribution. Our open-ended texture classification results are presented in the form of a probability map showing the associated goodness-of-fit for each pixel and its surrounding area being classed as being from the same model that describes the training texture.

2.4.1 Advantages in using nonparametric MRF models

We present a nonparametric MRF model that can capture the high order statistical characteristics of a texture, demonstrated by the synthesis of some quite complex textures from the Brodatz album [28]. To show that our nonparametric MRF model is correctly synthesising these natural textures, we take a small section of texture and synthesise the same texture over a larger area. The subjective similarity between the original and synthesised textures over the larger area demonstrates that the texture model has indeed captured all of the essential characteristics of the texture. We found that the synthesis process generated textures with few phase discontinuities.

We show how we can adjust the statistical order of our nonparametric MRF model while retaining the original dimensionality. This was accomplished through some original work showing the mathematical equivalence of Moussouris' strong MRF model [148] and the analysis-of-variance (ANOVA) model [67]. With this new strong nonparametric MRF model we present new results for open-ended classification of texture.

The model presented in this thesis would be unsuitable for image compression or fast texture synthesis, since for such purposes a parametric version of the model would be required. However, as the model is capable of accurately synthesising a wide variety of textures, it does represent the basis of a complete texture model. With such a model it would be possible to start contemplating the existence of an ideal model, one that could be used for open-ended texture classification. When we used this model for such an application we obtained a classification accuracy of 87% on a set of a hundred VisTex texture mosaics [49]. The advantage of using a nonparametric model for such an application is that it does not involve a long training process to obtain some thing close to an ideal model. That is a model that is capable of synthesising visually similar representations of a training texture. Therefore new textures can be presented to the model to perform open-ended texture classification at will, and the overall computational time would be substantially more favourable than for a parametric MRF model.

Chapter 3

MRF Model

This thesis is primarily focused towards MRF models. Consequently, we present here a review of some of the fundamental theorems and proofs on MRFs. However in order to be consistent, the terminology has been standardised with the remainder of this thesis. These theorems and proofs also provide a foundation for our own work on the strong nonparametric MRF in Chapter 6.

3.1 Random field preliminaries

Geman [78] uses the following notation for random fields:– Denote the *sites* of a lattice by $S = \{s_1, s_2, \ldots, s_N\}$, with a variable X_s at each site $s \in S$. The complete set of variables for the whole lattice is denoted by $\mathbf{X} = \{X_s, s \in S\}$. Each variable X_s is assigned a value x_s from its *state space* Λ_s, such that $X_s = x_s,\ x_s \in \Lambda_s \subset \Re\, \forall s \in S$. A particular configuration for the lattice is given as $\{X_1 = x_1, X_2 = x_2, \ldots, X_N = x_N\}$ which is abbreviated to $\{\mathbf{X} = \mathbf{x}\}$ where $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ or $\mathbf{x} = \{x_s\}$ for convenience. The *configuration space* for the variable $\mathbf{x}$ is denoted by Ω, whereby,

$$\Omega = \prod_{s \in S} \Lambda_s, \qquad \Lambda_s \subset \Re \tag{3.1}$$

For simplicity we may assume a common state space, $\Lambda \doteq \{0, 1, 2, \ldots, L-1\}$, where L is the number of *grey levels*. An image is modelled by defining each pixel in the image as a random variable $X_s, s \in S$, and the grey level associated with the pixel equal to the value x_s. Given that x_s comes from a common state space Λ, all possible images $\mathbf{x} = \{x_s\}$ are then contained within the configuration space $\Omega = \Lambda^N$.

Let Π be the (joint) probability measure on Ω with $\Pi(\mathbf{X} = \mathbf{x}) > 0 \ \forall \ \mathbf{x} \in \Omega$. Besag [17] proved that the joint distribution $\Pi(\mathbf{x})$ is uniquely determined by its Local Conditional Probability Density Function (LCPDF) $\Pi(X_s = x_s | X_r = x_r, r \neq s)$, which we will rewrite as:

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = \Pi(X_s = x_s|X_r = x_r, r \neq s), \qquad s \in S, \mathbf{x} \in \Omega \tag{3.2}$$

where $\mathbf{x}_{(s)} = \{x_r, r \neq s\}$.

We state this as,

Proposition 3.1 *The joint distribution of* $\mathbf{X} = \{X_s\}$ *is uniquely determined by its LCPDFs.*

Proof: This proof is from Besag [17]. We will verify that for any $\mathbf{x}, \mathbf{y} \in \Omega$:

$$\frac{P(\mathbf{x})}{P(\mathbf{y})} = \prod_{i=1}^{N} \frac{P(x_i|x_1, \ldots, x_{i-1}, y_{i+1}, \ldots, y_N)}{P(y_i|x_1, \ldots, x_{i-1}, y_{i+1}, \ldots, y_N)} \tag{3.3}$$

Proof of Eq. (3.3) follows. Clearly, we may write

$$P(\mathbf{x}) = P(x_N|x_1, \ldots, x_{N-1})P(x_1, \ldots, x_{N-1}). \tag{3.4}$$

However, $P(x_1, \ldots, x_{N-1})$ cannot be easily factorised in a useful way since, for example, $P(x_{N-1}|x_1, \ldots, x_{N-2})$ is not easily obtained from the given conditional distributions. Nevertheless, we can introduce y_N, write

$$P(\mathbf{x}) = \frac{P(x_N|x_1, \ldots, x_{N-1})}{P(y_N|x_1, \ldots, x_{N-1})} P(x_1, \ldots, x_{N-1}, y_N) \tag{3.5}$$

and now operate on x_{N-1} in $P(x_1, \ldots, x_{N-1}, y_N)$. This yields

$$P(x_1, \ldots, x_{N-1}, y_N) = \frac{P(x_{N-1}|x_1, \ldots, x_{N-2}, y_N)}{P(y_{N-1}|x_1, \ldots, x_{N-2}, y_N)} P(x_1, \ldots, x_{N-2}, y_{N-1}, y_N), \tag{3.6}$$

after similar introduction of y_{N-1}. Continuing the reduction process, we eventually arrive at Eq. (3.3). Assuming that two probability measures Π, μ on Ω have the same LCPDFs, we have $\frac{\Pi(\mathbf{x})}{\Pi(\mathbf{y})} = \frac{\mu(\mathbf{x})}{\mu(\mathbf{y})}$ which implies $\Pi = \mu$ [78].

□

3.2 General Markov random field model

The property of a Markov Random Field (MRF) is that: a variable X_s at a site s on a lattice S can be equal to any value $x_s \in \Lambda$, but the probability of $X_s = x_s$ is, and only is, conditional upon those values x_r at the *neighbouring sites* of s. Given that the set of neighbouring sites of s is denoted by the *neighbourhood* $\mathcal{N}_s \subset S$, then an MRF is a process $\mathbf{X} = \{X_s\}$ such that:

$$\Pi(\mathbf{x}) > 0 \qquad \forall \mathbf{x} \in \Omega \tag{3.7}$$

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = P(x_s|x_r, r \in \mathcal{N}_s) \qquad \forall \mathbf{x} \in \Omega, s \in S \tag{3.8}$$

For each site s there is associated a neighbourhood $\mathcal{N}_s$, and the set of neighbourhoods is the *neighbourhood system* denoted as $\mathcal{N} = \{\mathcal{N}_s \subset S, s \in S\}$.

The *Hammersley-Clifford theorem*, which is also referred to as the *MRF-Gibbs equivalence theorem*, and proved in [17, 92, 148, 78], gives form to the LCPDF so as to define a valid joint distribution $\Pi(\mathbf{x})$. The theorem implicitly requires the neighbourhood system to adhere to the following criteria.

1. $s \notin \mathcal{N}_s$

2. $s \in \mathcal{N}_t \Leftrightarrow t \in \mathcal{N}_s$

This implies that the neighbourhoods must be symmetrical and self similar for homogeneous MRFs.

The symmetrical neighbourhood systems employed in this thesis are the same as used in [78, 82] for which the neighbourhood system $\mathcal{N}^o = \{\mathcal{N}_s^o, s \in S\}$ is defined as

$$\mathcal{N}_s^o = \left\{r \in S : 0 < |s - r|^2 \leq o\right\}, \tag{3.9}$$

where $|s - r|$ is the Euclidean distance between two points $s, r \in S$. The neighbourhood $\mathcal{N}_s^o$ is defined by the neighbourhood order o (but this does not refer to the statistical order of the neighbourhood). A first order neighbourhood system $o = 1$ is shown in Fig. 3.1(a), which is also called the *nearest neighbour* neighbourhood system consisting of the four nearest adjacent pixels. The second and eighth order neighbourhood systems for $o = 2$ and $o = 8$ are shown in Figs. 3.1(b), and (c) respectively.

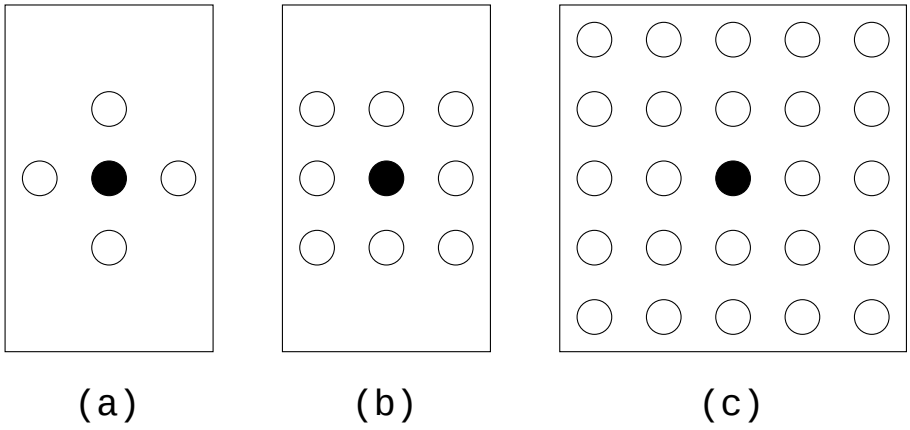


Figure 3.1: Neighbourhoods: (a) The first order neighbourhood $o = 1$ or *nearest-neighbour* neighbourhood for the site $s =$ '$\bullet$' and $r =$ '$\circ$' $\in \mathcal{N}_s$; (b) second order neighbourhood $o = 2$; (c) eighth order neighbourhood $o = 8$.

3.3 Gibbs distribution

The *Gibbs distribution* is a representation for a positive measure Π on Ω. The Gibbs distribution is defined with respect to a set of *potential functions* $V = \{V_A : A \subset S\}, V_A : \Omega \to \Re$. The essential ingredients of the potential functions V are $V_\emptyset(\mathbf{x}) = 0$ and $V_A(\mathbf{x}) = V_A(\mathbf{x}')$ if $x_s = x'_s,\ \forall\ s \in A$. V is *normalised* if $V_A(\mathbf{x}) = 0$ whenever $x_s = 0$ for some $s \in A$, where we assume $0 \in \Lambda$, although any other consistent value would do. Normalised potentials ensure unique representation but really have no other practical importance.

The *energy* associated with a particular realisation $\{\mathbf{X} = \mathbf{x}\}$ is defined as,

$$U(\mathbf{x}) = -\sum_{A \subset S} V_A(\mathbf{x}) \tag{3.10}$$

and the joint probability is given as,

$$\Pi(\mathbf{x}) = \frac{1}{Z} \exp\{-U(\mathbf{x})\} \tag{3.11}$$

where Z is the *normalising constant* or *partition function*,

$$Z = \sum_{\mathbf{x} \in \Omega} \exp\{-U(\mathbf{x})\}. \tag{3.12}$$

Generally Z is intractable both analytically and numerically. It is however tractable

for the GMRF model [34, 40]

3.3.1 Cliques

Given a neighbourhood system $\mathcal{N}$, a *clique* is a set $C \subseteq S$ if every pair of distinct sites in C are neighbours. That is, given $s, r \in C, s \neq r$ implies $s \in \mathcal{N}_r$. The single site subset is also a clique. Let $\mathcal{C}$ denote the set of cliques defined on S with respect to $\mathcal{N}$, and let $\mathcal{C}_s$ denote the *local clique set* for a neighbourhood $\mathcal{N}_s$ such that $\mathcal{C}_s = \{C \in \mathcal{C}, s \in C\}$. Cliques are important when considering the equivalence between MRFs and the Gibbs distribution.

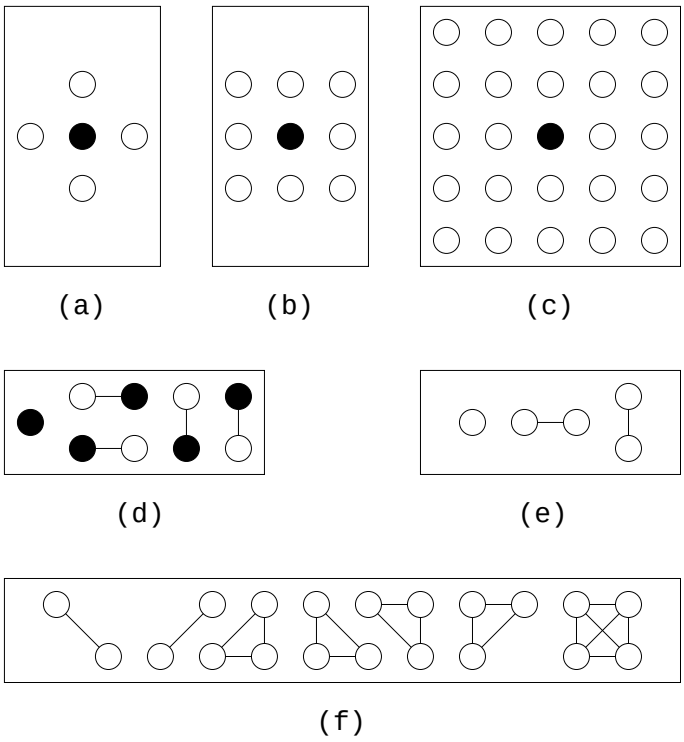


Figure 3.2: Neighbourhoods and cliques: (a) The first order neighbourhood $o = 1$ or *nearest-neighbour* neighbourhood for the site $s =$ '$\bullet$' and $r =$ '$\circ$' $\in \mathcal{N}_s$; (b) second order neighbourhood $o = 2$; (c) eighth order neighbourhood $o = 8$. (d) local clique set for nearest-neighbour neighbourhood; (e) clique types for nearest-neighbour neighbourhood; (f) additional clique types for second-order neighbourhood.

Figures 3.2 (a), (b), and (c) show the neighbourhood configurations for $o = 1, 2$ and 8 respectively. If we represent the lattice S on a rectangular grid $Z_m = \{(i, j) : 1 \leq i, j \leq m\}$ where $S = Z_m, N = m^2$, then the first-order or nearest-neighbour system $\mathcal{N}^1_{i,j} = \{(i, j-1), (i, j+1), (i-1, j), (i+1, j)\}$. The cliques

associated with this neighbourhood system are then those subsets of S whereby $\{(i,j)\}, \{(i,j),(i,j+1)\}$ and $\{(i,j),(i+1,j)\} \subset Z_m$, as shown in Fig. 3.2 (e). The cliques contained in the local clique set $\mathcal{C}_s$ of $\mathcal{N}^1_{i,j}$ are then those cliques $\{(i,j)\}$, $\{(i,j),(i,j+1)\}$, $\{(i,j),(i+1,j)\}$, $\{(i,j),(i,j-1)\}$ and $\{(i,j),(i-1,j)\}$, as shown in Fig. 3.2 (d). For the second-order neighbourhood $\mathcal{N}^2_s$, the set of cliques $\mathcal{C}$ are those of type shown in Figs. 3.2 (e) and (f).

The number of clique types grows almost exponentially with increasing order o. In Appendix A we illustrate a general method for extracting the local clique set from any neighbourhood system. We have published this method in "IEE Proceedings Vision, Image and Signal Processing" [154].

Geman and Geman [82] mainly experimented with small clique sizes. They found that they could model their $\mathbf{X}$ quite well with single site and pairwise cliques. However they realised that more complex images might have to be modelled with more complex cliques, *i.e.,* cliques of three or more sites. They also suggested that this extra complexity could be accommodated while maintaining modest neighbourhood sizes by developing a hierarchy of MRFs, *i.e.,* model $\mathbf{X}$ with various little neighbourhoods that contain simple clique structures, instead of one large neighbourhood.

3.3.2 The $\mathcal{N}$-Potential V

An $\mathcal{N}$-Potential V not only fulfils all previous criteria for a potential function but, given a neighbourhood system $\mathcal{N}$ and its corresponding set of cliques $\mathcal{C}$, a $\mathcal{N}$-Potential V is defined such that,

$$V_C(\mathbf{x}) = 0 \qquad \text{if} \qquad C \notin \mathcal{C}. \tag{3.13}$$

The Gibbs distribution Π is then defined as,

$$\Pi(\mathbf{x}) = \frac{1}{Z} \exp\left\{ \sum_{C \in \mathcal{C}} V_C(\mathbf{x}) \right\}, \tag{3.14}$$

where the normalising constant or partition function is,

$$Z = \sum_{\mathbf{x} \in \Omega} \exp\left\{ \sum_{C \in \mathcal{C}} V_C(\mathbf{x}) \right\}. \tag{3.15}$$

A representation for the $\mathcal{N}$-Potential V is given by Grimmett [92] and stated below in Proposition 3.2. However we will first introduce the Möbius inversion theorem [177] which is used in Grimmett's representation.

Theorem 3.1 (Möbius inversion theorem) *For arbitrary real functions F and G defined on the subsets A, B and C of some finite set.*

$$F(A) = \sum_{B \subseteq A} G(B) \qquad \textit{iff} \qquad G(B) = \sum_{C \subseteq B} (-1)^{|B|-|C|} F(C) \tag{3.16}$$

or, equivalently,

$$F(A) = \sum_{B \subseteq A} \sum_{C \subseteq B} (-1)^{|B|-|C|} F(C) \tag{3.17}$$

where $|A|$ = number of sites in set A.

Proof of Theorem 3.1: This elegant proof is by Moussouris [148]. The identity Eq. (3.17) holds because F(A) occurs only once in the sum, whereas any C with m fewer elements than A can be extended $\binom{m}{j}$ ways to Bs containing j of the missing elements; since $|B| - |C| = j$, the total coefficient of $F(C)$ is then $\sum_{j=0}^{m} (-1)^j \binom{m}{j} = (1-1)^m = 0$.

□

Before we continue, we need to introduce more notation. For $\mathbf{x} \in \Omega, A \subset S$, denote

$$\mathbf{x}^A = \{x_s^A, s \in S\}, \qquad x_s^A = \begin{cases} x_s, & s \in A \\ 0, & s \notin A. \end{cases} \tag{3.18}$$

For the rest of the thesis the notation C and C' will be specifically reserved for representing cliques for which $C, C' \in \mathcal{C}$.

Proposition 3.2 *Any $\Pi > 0$ is a Gibbs distribution with respect to $\mathcal{N}$-potentials,*

$$V_C(\mathbf{x}) = \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log \Pi(\mathbf{x}^{C'}), \qquad \forall \mathbf{x} \in \Omega, s \in S. \tag{3.19}$$

Moreover, for any element $s \in C$,

$$V_C(\mathbf{x}) = \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log \Pi_s(x_s^{C'} | \mathbf{x}_{(s)}^{C'}), \qquad \forall \mathbf{x} \in \Omega, s \in S, s \in C \tag{3.20}$$

where $C, C' \in \mathcal{C}$.

Proof of Proposition 3.2: is given by Grimmett [92] and Moussouris [148], but a thorough proof is given by Geman [78], which is the one given here.

1. Π *is Gibbs with respect to the potential:*

$$V'_C(\mathbf{x}) = \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log[\Pi(\mathbf{x}^{C'})/\Pi(\mathbf{0})] \tag{3.21}$$

where,

$$\begin{cases} F(C') &= \log[\Pi(\mathbf{x}^{C'})/\Pi(\mathbf{0})] \\ G(C) &= V_C(\mathbf{x}) \end{cases} \tag{3.22}$$

where $\mathbf{x}$ is fixed and $\mathbf{0} = \{0, 0, \ldots, 0\}$. An important difference between the potential V_C Eq. (3.19) and the above potential V'_C Eq. (3.21) is

$$V_\emptyset = \log \Pi(\mathbf{0}) \tag{3.23}$$

where as

$$V'_\emptyset = \log[\Pi(\mathbf{0})/\Pi(\mathbf{0})] = 0 \tag{3.24}$$

which is how a Gibbs potential is defined. This is the only difference, which is inconsequential as indicated by the *note* below. Assuming Eq. (3.21) and using the Möbius inversion formula,

$$\log \frac{\Pi(\mathbf{x})}{\Pi(\mathbf{0})} = \log \frac{\Pi(\mathbf{x}^S)}{\Pi(\mathbf{0})} = F(S) = \sum_{C \subseteq S} G(C) = \sum_{C \subseteq S} V'_C(\mathbf{x}) \tag{3.25}$$

Thus, $\Pi(\mathbf{x}) = \Pi(\mathbf{0}) e^{-U(\mathbf{x})}$, where $U(\mathbf{x}) = -\sum_{C \subseteq S} V'_C(\mathbf{x})$, and $Z = (\Pi(\mathbf{0}))^{-1}$, *i.e.,* Π is a Gibbs distribution. Note:

$$\begin{aligned} \Pi(\mathbf{x}) &= \Pi(\mathbf{0}) \exp\left\{ \sum_{C \subseteq S} V'_C(\mathbf{x}) \right\} \\ &= \Pi(\mathbf{0}) \exp\left\{ \sum_{C \subseteq S} \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log[\Pi(\mathbf{x}^{C'})/\Pi(\mathbf{0})] \right\} \\ &= \Pi(\mathbf{0}) \exp\left\{ \left[\sum_{C \subseteq S} \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log \Pi(\mathbf{x}^{C'}) \right] - \right. \end{aligned} \tag{3.26}$$

$$\left[\sum_{C\subseteq S}\sum_{C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi(\mathbf{0})\right]\Bigg\}$$

$$\begin{aligned}
&= \Pi(\mathbf{0})\exp\left\{\sum_{C\subseteq S}\sum_{C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi(\mathbf{x}^{C'})\right\}\exp\{-\log\Pi(\mathbf{0})\}\\
&= \frac{\Pi(\mathbf{0})}{\Pi(\mathbf{0})}\exp\left\{\sum_{C\subseteq S}\sum_{C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi(\mathbf{x}^{C'})\right\}\\
&= \exp\left\{\sum_{C\subseteq S}V_C(\mathbf{x})\right\} \qquad (3.27)
\end{aligned}$$

From the Möbius inversion of Eq. (3.26) we obtain Eq. (3.21) but from the Möbius inversion of Eq. (3.27) we obtain Eq. (3.19), therefore V_C and V'_C are interchangeable provided the correct corresponding Eq. (3.26) or Eq. (3.27) is used.

2. *V is normalised:* For any $s \in C$,

$$\begin{aligned}
V_C(\mathbf{x}) &= \sum_{s\notin C'\subseteq C}(-1)^{|C-C'|}\log\Pi(\mathbf{x}^{C'}) + \sum_{s\in C'\subseteq C}(-1)^{|C-C'|}\log\Pi(\mathbf{x}^{C'})\\
&= \sum_{C'\subseteq C-s}(-1)^{|C-C'|}\left[\log\Pi(\mathbf{x}^{C'}) - \log\Pi(\mathbf{x}^{C'+s})\right] \qquad (3.28)
\end{aligned}$$

If $x_s = 0$, then $\mathbf{x}^{C'} = \mathbf{x}^{C'+s} \Rightarrow V_C(\mathbf{x}) = 0$.

3. *Eq. (3.19) $\Leftrightarrow$ Eq. (3.20).* This follows from Eq. (3.28) by applying the identity

$$\frac{\Pi(\mathbf{x}^{C'})}{\Pi(\mathbf{x}^{C'+s})} = \frac{\Pi_s(x_s^{C'}|\mathbf{x}_{(s)}^{C'})}{\Pi_s(x_s^{C'+s}|\mathbf{x}_{(s)}^{C'+s})}, \qquad s \notin C' \qquad (3.29)$$

4. $V_A = 0\ \forall A \notin \mathcal{C}$. This will be proved via the equivalence theorem in Section 3.4, where it will be shown that $V_A = 0\ \forall A \notin \mathcal{C}$ if and only if Π is a MRF.

□

3.4 MRF — Gibbs distribution equivalence

The MRF–Gibbs distribution equivalence was first established by Hammersley and Clifford [96] but never published as they preferred the proof given by Besag [17]. An alternative proof based on the Möbius inversion theorem [177] was given by Grimmett [92] and Moussouris [148] which was rewritten by Geman [78].

The MRF–Gibbs distribution equivalence theorem gives form to the LCPDF of an MRF by expressing it in terms of $\mathcal{N}$-potentials $V_C(\mathbf{x})$ [82, 78],

$$P(x_s|x_r, r \in \mathcal{N}_s) = \frac{1}{Z_s} \exp\left\{ \sum_{C \in \mathcal{C}_s} V_C(\mathbf{x}) \right\}, \tag{3.30}$$

where Z_s is the *local normalising constant* $Z_s = \sum_{\lambda_s \in \Lambda} P(\lambda_s|x_r, r \in \mathcal{N}_s)$ and the summation is over the *local clique set* denoted by $\mathcal{C}_s = \{C \in \mathcal{C}, s \in C\}$.

An important point made by Dubes and Jain [59] is that, although the MRF–Gibbs Equivalence states that there exists an MRF model for every Gibbs distribution and vice-versa, this is only true when the Gibbs distribution is defined with respect to $\mathcal{N}$-Potentials and the MRF may be expressed as Eq. (3.30). It is possible to define an MRF that does not obey Eq. (3.30) but this could lead to a model whose joint distribution does not exist, and therefore produce inconsistent results. Alternatively it is also possible to define a Gibbs distribution that does not use $\mathcal{N}$-Potentials, but then this could lead to a model whose local distribution does not exist. For such a model it would be difficult to estimate the parameters and to sample the distribution to produce realisations. Therefore it is advisable to limit oneself to MRF and Gibbs models that are defined with respect to the clique structure.

Theorem 3.2 (MRF–Gibbs Equivalence) *Given a neighbourhood system $\mathcal{N}$, Π is a Gibbs distribution with respect to $\mathcal{N}$ if and only if Π is a MRF with respect to $\mathcal{N}$; in which case $\{V_C\}$ is a $\mathcal{N}$-potential given by*

$$V_C(\mathbf{x}) = \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log \Pi(\mathbf{x}^{C'}), \qquad \forall C, C' \in \mathcal{C}, \mathbf{x} \in \Omega, \tag{3.31}$$

3.4.1 Proof 1: by G. Grimmett

This proof is by Grimmett [92] but was rewritten by Geman [78]. Grimmett [92] based the proof on the Möbius inversion theorem [177], which we gave in Theo-

rem 3.1.

Proof - part 1. Suppose Π is a MRF w.r.t. $\mathcal{N}$ and V is defined as in Eq. (3.31), then V is an $\mathcal{N}$-potential if $V_A(\mathbf{x}) = 0$ if $A \notin \mathcal{C}$.

Choose $A \notin \mathcal{C}$. Then $\exists s, r \in A$ such that $r \notin \mathcal{N}_s + s$. Then:

$$\begin{aligned} V_A(\mathbf{x}) &= \sum_{B\subseteq A} (-1)^{|A|-|B|} \log \Pi_s(x_s^B|\mathbf{x}_{(s)}^B) \\ &= \sum_{B\subseteq A-s-r} (-1)^{|A-B|} \log \Pi_s(x_s^B|\mathbf{x}_{(s)}^B) + \\ &\quad \sum_{B\subseteq A-s-r} (-1)^{|A-(B+s)|} \log \Pi_s(x_s^{B+s}|\mathbf{x}_{(s)}^{B+s}) + \\ &\quad \sum_{B\subseteq A-s-r} (-1)^{|A-(B+r)|} \log \Pi_s(x_s^{B+r}|\mathbf{x}_{(s)}^{B+r}) + \\ &\quad \sum_{B\subseteq A-s-r} (-1)^{|A-(B+s+r)|} \log \Pi_s(x_s^{B+s+r}|\mathbf{x}_{(s)}^{B+s+r}) \\ &= \sum_{B\subseteq A-s-r} (-1)^{|A-B|} \log \left[\frac{\Pi_s(x_s^B|\mathbf{x}_{(s)}^B)\Pi_s(x_s^{B+s+r}|\mathbf{x}_{(s)}^{B+s+r})}{\Pi_s(x_s^{B+s}|\mathbf{x}_{(s)}^{B+s})\Pi_s(x_s^{B+r}|\mathbf{x}_{(s)}^{B+r})} \right] \end{aligned} \tag{3.32}$$

But $r \notin \mathcal{N}_s + s$ implies that $\Pi_s(x_s^B|\mathbf{x}_{(s)}^B) = \Pi_s(x_s^{B+r}|\mathbf{x}_{(s)}^{B+r})$ and that $\Pi_s(x_s^{B+s}|\mathbf{x}_{(s)}^{B+s}) = \Pi_s(x_s^{B+s+r}|\mathbf{x}_{(s)}^{B+s+r})$, and consequently that $V_A(\mathbf{x}) = 0$.

Proof - part 2. Now suppose that Π has a Gibbs representation w.r.t. $\mathcal{N}$ for some potential function V, whereby $V_A(\mathbf{x}) = 0$ for $A \notin \mathcal{C}$ and

$$\Pi(\mathbf{x}) = \frac{1}{Z} \exp\left\{ \sum_{C\in\mathcal{C}} V_C(\mathbf{x}) \right\} \qquad \forall \mathbf{x} \in \Omega. \tag{3.33}$$

Then

$$\begin{aligned} \Pi_s(x_s|\mathbf{x}_{(s)}) &= \frac{\exp\{\sum_{C\in\mathcal{C}} V_C(\mathbf{x})\}}{\sum_{\lambda_s\in\Lambda} \exp\{\sum_{C\in\mathcal{C}} V_C(\lambda_s, \mathbf{x}_{(s)})\}} \\ &= \frac{\exp\{\sum_{s\in C\in\mathcal{C}} V_C(\mathbf{x}) + \sum_{s\notin C\in\mathcal{C}} V_C(\mathbf{x})\}}{\sum_{\lambda_s\in\Lambda} \exp\{\sum_{s\in C\in\mathcal{C}} V_C(\lambda_s, \mathbf{x}_{(s)}) + \sum_{s\notin C\in\mathcal{C}} V_C(\lambda_s, \mathbf{x}_{(s)})\}} \\ &= \frac{\exp\{\sum_{s\in C\in\mathcal{C}} V_C(\mathbf{x})\}}{\sum_{\lambda_s\in\Lambda} \exp\{\sum_{s\in C\in\mathcal{C}} V_C(\lambda_s, \mathbf{x}_{(s)})\}} \\ &= \frac{1}{Z_s} \exp\left\{ \sum_{C\in\mathcal{C}_s} V_C(\mathbf{x}) \right\} \end{aligned} \tag{3.34}$$

since $V_C(\lambda_s, \mathbf{x}_{(s)}) = V_C(\mathbf{x})$ if $s \notin C$. Now $\Pi_s(x_s|\mathbf{x}_{(s)})$ only depends on V_C for $C \in \mathcal{C}_s$, *i.e.,* $C \subset \mathcal{N}_s + s$, and therefore it follows that,

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = P(x_s|x_r, r \in \mathcal{N}_s). \tag{3.35}$$

□

3.4.2 Proof 2: by J. Besag

This proof is by Besag [17]. From the positivity condition Eq. (3.7), $\Pi(\mathbf{x}) > 0\ \forall \mathbf{x} \in \Omega$, we may define

$$Q(\mathbf{x}) \equiv \log\{\Pi(\mathbf{x})/\Pi(\mathbf{0})\}, \qquad \forall \mathbf{x} \in \Omega. \tag{3.36}$$

Given any $\mathbf{x} \in \Omega$, denote

$$\mathbf{x}_i = \{x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_N\} \tag{3.37}$$

The problem to which Hammersley and Clifford [96] addressed themselves may now be stated as follows: *given the neighbours of each site, what is the general form of* $Q(\mathbf{x})$ *which may give a valid probability structure to the system?* Since

$$\begin{aligned} \exp\{Q(\mathbf{x}) - Q(\mathbf{x}_i)\} &= \Pi(\mathbf{x})/\Pi(\mathbf{x}_i) \\ &= \frac{\Pi_i(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)}{\Pi_i(0|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)} \end{aligned} \tag{3.38}$$

Besag's alternative proof to the Hammersley-Clifford theorem rests upon the observation that any probability distribution Π, subject to the above conditions, there exists an expansion of $Q(\mathbf{x})$, unique on Ω and of the form

$$\begin{aligned} Q(\mathbf{x}) &= \sum_{1 \le i \le N} x_i G_i(x_i) + \sum_{1 \le i < j \le N} \sum x_i x_j G_{i,j}(x_i, x_j) + \\ &\quad \sum_{1 \le i < j < k \le N} \sum \sum \sum x_i x_j x_k G_{i,j,k}(x_i, x_j, x_k) + \cdots + \\ &\quad x_1 x_2 \ldots x_N G_{1,2,\ldots,N}(x_1, x_2, \ldots, x_N). \end{aligned} \tag{3.39}$$

Hammersley and Clifford's result may be stated in the following manner:

Proposition 3.3 *For any* $1 \le i < j < \cdots < r \le N$, *the function* $G_{i,j,\ldots,r}$ *in*

Eq. (3.39) may be non-null if and only if the sites $i, j, \ldots, r$ form a clique. Subject to this restriction, the G-functions may be chosen arbitrarily.

Proof. It follows from Eq. (3.38) that, for any $\mathbf{x} \in \Omega, Q(\mathbf{x}) - Q(\mathbf{x}_i)$ can only depend upon x_i itself and the values at sites which are neighbours of site s_i. Without loss of generality, we shall only consider site s_1 in detail. We then have from Eq. (3.39),

$$\begin{aligned} Q(\mathbf{x}) - Q(\mathbf{x}_1) &= x_1 \Bigg\{ G_1(x_1) + \sum_{2 \le j \le N} x_j G_{1,j}(x_1, x_j) + \\ &\qquad \sum_{2 \le j < k \le N} \sum x_j x_k G_{1,j,k}(x_1, x_j, x_k) \cdots + \\ &\qquad x_2 x_3 \ldots x_N G_{1,2,\ldots,N}(x_1, x_2, \ldots, x_N) \Bigg\}. \end{aligned} \tag{3.40}$$

Now suppose site s_l ($l \neq 1$) is *not* a neighbour of site s_1. Then $Q(\mathbf{x}) - Q(\mathbf{x}_1)$ must be independent of x_l for all $\mathbf{x} \in \Omega$. Putting $x_i = 0$ for $i \neq 1$ or l, we immediately see that $G_{1,l}(x_1, x_l) = 0$ on Ω. Similarly, by suitable choices of $\mathbf{x}$, it is easily seen successively that all 3-, 4-, $\ldots, n$-variable G-functions involving both x_1 and x_l must be null. The analogous result holds for any pair of sites which are not neighbours of each other and hence, in general, $G_{i,j,\ldots,r}$ can only be non-null if the sites $i, j, \ldots, r$ form a clique.

On the other hand, any set of G-functions gives rise to a valid probability distribution Π which satisfies the positivity condition. Also since $Q(\mathbf{x}) - Q(\mathbf{x}_i)$ depends only upon x_l if there is a non-null G-function involving both x_i and x_l, it follows that the same is true of $\Pi_i(x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_N)$.

□

3.5 Factorisation of the probability distribution

The following construction is by Moussouris [148]. Given Eq. (3.19),

$$V_C(\mathbf{x}) = \sum_{C' \subseteq C} (-1)^{|C| - |C'|} \log \Pi(\mathbf{x}^{C'}), \qquad \forall \mathbf{x} \in \Omega, s \in S \tag{3.41}$$

and its Möbius inversion Eq. (3.27),

$$\Pi(\mathbf{x}) = \exp\left\{\sum_{C\subseteq S} V_C(\mathbf{x})\right\} \tag{3.42}$$

then

$$\Pi(\mathbf{x}) = \exp\left\{\sum_{C\subseteq S}\sum_{C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi(\mathbf{x}^{C'})\right\}. \tag{3.43}$$

From this equation Moussouris [148] produced the following reconstruction of the clique decomposition formulae,

$$\Pi(\mathbf{x}) = \prod_{C\subseteq S}\Pi(\mathbf{x}^C)^{n_{SC}}, \qquad \text{where} \qquad n_{SC} = (-1)^{|C|}\sum_{C\subseteq C'\subseteq S}(-1)^{|C'|} \tag{3.44}$$

A similar equation can be obtained for the LCPDF from Eq. (3.20) and given Eq. (3.30),

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = \frac{1}{Z_s}\exp\left\{\sum_{C\in\mathcal{C}_s} V_C(\mathbf{x})\right\}. \tag{3.45}$$

The normalising term Z_s can be calculated by equating $x_s = 0$ whereby,

$$\Pi_s(0|\mathbf{x}_{(s)}) = \frac{1}{Z_s} \tag{3.46}$$

Now by substituting Eq. (3.20) into Eq. (3.45) we obtain,

$$\begin{aligned}
\Pi_s(x_s|\mathbf{x}_{(s)}) &= \Pi_s(0|\mathbf{x}_{(s)})\exp\left\{\sum_{C\in\mathcal{C}_s}\sum_{C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi_s(x_s^{C'}|\mathbf{x}_{(s)}^{C'})\right\} \\
\frac{\Pi_s(x_s|\mathbf{x}_{(s)})}{\Pi_s(0|\mathbf{x}_{(s)})} &= \exp\left\{\sum_{C\in\mathcal{C}_s}\left[\sum_{s\in C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi_s(x_s^{C'}|\mathbf{x}_{(s)}^{C'}) + \right.\right. \\
&\qquad \left.\left.\sum_{s\notin C'\subseteq C}(-1)^{|C|-|C'|}\log\Pi_s(x_s^{C'}|\mathbf{x}_{(s)}^{C'})\right]\right\} \\
&= \exp\left\{\sum_{C\in\mathcal{C}_s}\sum_{s\in C'\subseteq C}(-1)^{|C|-|C'|}\left[\log\Pi_s(x_s|\mathbf{x}_{(s)}^{C'}) - \log\Pi_s(0|\mathbf{x}_{(s)}^{C'})\right]\right\} \\
&= \exp\left\{\sum_{C\in\mathcal{C}_s}\sum_{s\in C'\subseteq C}(-1)^{|C|-|C'|}\log\left(\frac{\Pi_s(x_s|\mathbf{x}_{(s)}^{C'})}{\Pi_s(0|\mathbf{x}_{(s)}^{C'})}\right)\right\}
\end{aligned} \tag{3.47}$$

Although the Eq. (3.47) is defined over the sites $\mathcal{N}_s + s$, if we just regard the sites

of the neighbourhood $\mathcal{N}_s$ we find that Eq. (3.47) is consistent with the Möbius decompostion of Eq. (3.17). Therefore we may apply the same reconstruction of Eq. (3.47) as Moussouris' [148] did to Eq. (3.43). This gives an equivalent clique decomposition of Eq. (3.47) as,

$$\frac{\Pi_s(x_s|\mathbf{x}_{(s)})}{\Pi_s(0|\mathbf{x}_{(s)})} = \prod_{C\in\mathcal{C}_s}\left(\frac{\Pi_s(x_s|\mathbf{x}^C_{(s)})}{\Pi_s(0|\mathbf{x}^C_{(s)})}\right)^{n_{\mathcal{C}_s C}} \qquad \text{where} \qquad n_{\mathcal{C}_s C} = (-1)^{|C|}\sum_{C\subseteq C'\in\mathcal{C}_s}(-1)^{|C'|} \tag{3.48}$$

Chapter 4

Parametric MRF Model

This chapter reviews the development of the parametric MRF model. It includes various standard models and a section on how the parameters for these models are estimated. It also looks at how these models are applied to synthesising, segmenting and classifying textures.

4.1 Introduction

To model an image as an MRF we first need to assume that the image can be represented as an MRF. Such an image would consist of a homogeneous texture, like the Reptile Skin from the Brodatz Album [28] as shown in Fig. 4.1.

Four basic issues need to be addressed in order to create an accurate MRF model of a textured image.

1. *Model definition,* the process of establishing the mathematical form of the model and the number of parameters. Some typical parametric MRF models are presented in Section 4.2. As part of defining the model, the following questions need to be considered.
   - What neighbourhood should be used?
   - Which clique functions should be used?
   - What form should these clique functions take?
2. *Parameter estimation,* the process by which the parameters of a given defined model are evaluated in order to fit that model to the textured image. This is

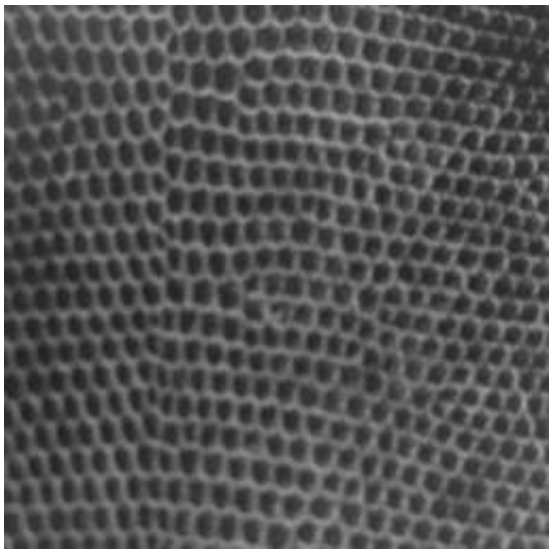


Figure 4.1: An example of texture which is to be modelled by an MRF. This texture is Reptile skin from the Brodatz Album [28]

addressed in Section 4.3.

3. *Sampling (realising, synthesising),* the process for creating a realisation of the model, *i.e.,* a textured image. This is addressed in Section 4.4.

4. *Goodness-of-fit testing.* Given an MRF model with parameters θ and a single texture image $\mathbf{x}$, the goodness-of-fit problem is to test the following null hypothesis.

$$H_0 : \text{The given image } \mathbf{x} \text{ is an observation from the MRF with parameters } \theta. \tag{4.1}$$

This is addressed in Section 4.5.

4.2 Auto-models

Auto-models are the simplest of the MRF models. They are defined as having an energy function that is dependent only upon the cliques containing no more than two sites.

$$U(\mathbf{x}) = -\sum_{C \in \mathcal{C}_s} V_C(\mathbf{x}) = -\sum_{1 \le s \le N} x_s G_s(x_s) - \sum_{1 \le s < r \le N} \sum x_s x_r G_{s,r}(x_s, x_r). \tag{4.2}$$

These types of models are commonly found in technical papers dealing with MRFs [17, 40, 48, 50, 105, 120, 181, 194]. Auto-models are the easiest to construct and to find parameters for [17, 181, 189], however the textures that have been represented by these models have so far been very limited in structure. The textures have tended to look like either random noise, blobs of different colours, vertical horizontal or diagonal lines, or a checker-board [181]. A picture from the Brodatz Album [28] such as the one shown in Fig. 4.1, has so far eluded duplication via an auto-model [95, 105].

As mentioned in Section 3.2 Geman and Geman [82] specified their neighbourhood system in the following form,

$$\mathcal{N}_s^o = \left\{ r \in S : 0 < |s - r|^2 \leq o \right\}, \tag{4.3}$$

For a neighbourhood of order o, Fig. 4.2 (a) shows the neighbourhood of s as all those pixels whose indicated value is less than or equal to o.

5	4	3	4	5
4	2	1	2	4
3	1	s	1	3
4	2	1	2	4
5	4	3	4	5

(a)

β_{-11}	β_{-7}	β_{-6}	β_{8}	β_{12}
β_{-9}	β_{-3}	β_{-2}	β_{4}	β_{10}
β_{-5}	β_{-1}	α	β_{1}	β_{5}
β_{-10}	β_{-4}	β_{2}	β_{3}	β_{9}
β_{-12}	β_{-8}	β_{6}	β_{7}	β_{11}

(b)

Figure 4.2: Neighbourhood order and parameters. a) Hierarchically arranged neighbourhood system. b) The parameter placement for the Auto-model.

Figure 4.2 (b) shows the numbering of the parameters for an auto-model given a neighbourhood order. The pairwise cliques are represented by the parameters β_i, and the single clique is represented by the single parameter α. If β_i is not included in the neighbourhood then $\beta_i = 0$. If the field is stationary, *i.e.,* when $\Pi_s(x_s|\mathbf{x}_{(s)})$ is independent of s, then $\beta_{-i} = \beta_i$.

4.2.1 Ising model

Let $\mathcal{N}$ be the nearest neighbour system $\mathcal{N}^1$. The classical Ising model is defined with respect to the common state space $\Lambda \equiv \{-1, +1\}$, corresponding to *spin up*, *spin down*, and

$$U(\mathbf{x}) = -\frac{\alpha}{T}\sum_{s\in S} x_s - \frac{\beta}{T}\sum_{\langle s,r\rangle\in S} x_s x_r, \tag{4.4}$$

where $\langle s, r\rangle$ denotes a nearest neighbour pair, T stands for *temperature*, α is the external magnetic field strength and β is the coupling strength. The model is "attractive" if $\beta > 0$ and "repulsive" if $\beta < 0$, see Kindermann and Snell [121].

4.2.2 Auto-binary

$$U(\mathbf{x}) = -\sum_{s} \alpha_s x_s - \sum_{s<r}\sum \beta_{sr} x_s x_r \qquad x_s \in \{0, 1\} \tag{4.5}$$

where $\beta_{rs} = \beta_{sr}$. The LCPDF is then defined as,

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = \frac{\exp\left\{x_s(\alpha_s + \sum_{r\in S}\beta_{sr}x_r)\right\}}{1 + \exp\left\{\alpha_s + \sum_{r\in S}\beta_{sr}x_r\right\}} \tag{4.6}$$

4.2.3 Auto-logistic

The *auto-logistic* model is a special case of the auto-binary model with $\alpha_s \equiv \alpha$, $\beta_{sr} \equiv \beta_v$ for vertical bonds, and $\beta_{sr} \equiv \beta_h$ for horizontal bonds. Note that an auto-logistic model is obtained if we assume the auto-binary model to be stationary, *i.e.,* independent of s.

4.2.4 Auto-binomial

Given the common state space $\Lambda \doteq \{0, 1, 2, \ldots, L\}$ then,

$$U(x) = -\sum_{s}\left[\log\left(\frac{L!}{x_s!(L-x_s)!}\right) + \alpha_s x_s\right] - \sum_{s<r}\sum \beta_{sr} x_s x_r \tag{4.7}$$

Therefore the LCPDF is defined as,

$$\begin{aligned}
\Pi_s(x_s|\mathbf{x}_{(s)}) &= \frac{\exp\left\{\log\left(\frac{L!}{x_s!(L-x_s)!}\right)+\alpha_s x_s+\sum_{r\in S}\beta_{sr}x_s x_r\right\}}{\sum_{\lambda_s\in\Lambda}\exp\left\{\log\left(\frac{L!}{\lambda_s!(L-\lambda_s)!}\right)+\alpha_s\lambda_s+\sum_{r\in S}\beta_{sr}\lambda_s x_r\right\}} \\
&= \left(\frac{L!}{x_s!(L-x_s)!}\right)\frac{\exp\left\{x_s(\alpha_s+\sum_{r\in S}\beta_{sr}x_r)\right\}}{\sum_{\lambda_s\in\Lambda}\left(\frac{L!}{\lambda_s!(L-\lambda_s)!}\right)\exp\left\{\lambda_s(\alpha_s+\sum_{r\in S}\beta_{sr}x_r)\right\}} \\
&= \left(\frac{L!}{x_s!(L-x_s)!}\right)\frac{\exp\left\{x_s(\alpha_s+\sum_{r\in S}\beta_{sr}x_r)\right\}}{(1+\exp\left\{\alpha_s+\sum_{r\in S}\beta_{sr}x_r\right\})^L} \\
&= \left(\frac{L!}{x_s!(L-x_s)!}\right)\tau^{x_s}(1-\tau)^{L-x_s} \qquad (4.8)
\end{aligned}$$

where

$$\tau = \frac{\exp\left\{\alpha_s+\sum_{r\in S}\beta_{sr}x_r\right\}}{1+\exp\left\{\alpha_s+\sum_{r\in S}\beta_{sr}x_r\right\}} \qquad (4.9)$$

In other words the LCPDF $\Pi_s(x_s|\mathbf{x}_{(s)})$ is binomial. This model was applied to texture by Cross and Jain [50].

4.2.5 Derin-Elliott

The Derin-Elliott [54] model has the following form,

$$U(\mathbf{x}) = -\sum_s \alpha x_s - \sum_{s<r}\sum \beta_{sr} I(x_s, x_r) \qquad (4.10)$$

where

$$I(x_s, x_r) = \begin{cases} 1 & x_s = x_r \\ -1 & x_s \neq x_r \end{cases} \qquad (4.11)$$

4.2.6 Auto-normal

A Gaussian Markov Random Field (GMRF) is a continuous random field, where the pixel values have jointly Gaussian distributions with means μ, standard deviations σ, and correlations β. The auto-normal model, described by Besag [17], is a pairwise interaction model of the GMRF. Here $\Pi_s(x_s|\mathbf{x}_{(s)})$ is now interpreted as a density function rather than a probability mass function. Define the image $\mathbf{x}$ on a

rectangular lattice $Z_m = \{(i,j) : 1 \leq i,j \leq m\}$ such that $S = Z_m, N = m^2$, then

$$\Pi_s(x_s|\mathbf{x}_{(s)}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}\left[x_s - \mu_s - \sum_{r\in S}\beta_{sr}(x_r - \mu_r)\right]^2\right\} \tag{4.12}$$

which leads to the joint density function

$$\Pi(\mathbf{x}) = \frac{|\mathbf{B}|^{\frac{1}{2}}}{(2\pi\sigma^2)^{\frac{N}{2}}} \exp\left\{-\frac{1}{2\sigma^2}(\mathbf{x}-\mu)^{\mathrm{T}}\mathbf{B}(\mathbf{x}-\mu)\right\} \tag{4.13}$$

where μ is an $N \times 1$ vector of arbitrary finite means, μ_s, and $\mathbf{B}$ is the $N \times N$ matrix whose diagonal elements are unity and whose off-diagonal (s,r) element is $-\beta_{s,r}$. $\mathbf{B}$ is symmetric, *i.e.,* $\beta_{r,s} = \beta_{s,r}$, but it is also required to be positive definite.

The following exposition is from Dubes and Jain [59]. Now $\sigma^2\mathbf{B}^{-1}$ is the $N \times N$ covariance matrix, where $\mathbf{B}^{-1}$ is the correlation matrix and is block circulant. As shown in Fig. 4.2, a stationary second-order GMRF has only four parameters $\{\beta_1, \beta_2, \beta_3, \beta_4\}$ in addition to μ and σ. The inverse of the correlation matrix for a stationary second-order GMRF is given below.

$$\mathbf{B} = \begin{bmatrix} B_{1,1} & B_{1,2} & . & . & B_{1,m} \\ B_{1,m} & B_{1,1} & . & . & B_{1,m-1} \\ . & . & . & . & . \\ . & . & . & . & . \\ B_{1,2} & B_{1,3} & . & . & B_{1,1} \end{bmatrix} \tag{4.14}$$

where each $B_{i,j}$ is a circulant $m \times m$ matrix defined as follows.

$$\begin{aligned} B_{1,1} &= \text{circulant } (1, -\beta_1, 0, 0, \ldots, 0, -\beta_1), \\ B_{1,2} &= \text{circulant } (-\beta_2, -\beta_3, 0, 0, \ldots, 0, -\beta_4), \\ B_{1,m} &= \text{circulant } (-\beta_2, -\beta_4, 0, 0, \ldots, 0, -\beta_3), \\ B_{1,j} &= \mathbf{0} \text{ for } 2 < j < m. \end{aligned} \tag{4.15}$$

The parameters of a GMRF can be estimated in any of the following ways,

- Maximum Likelihood Estimator (MLE) [181].
- Maximum Pseudo-Likelihood Estimator (MPLE), Besag [19, 20],
- Coding Scheme, Besag [17],

- Minimising Sum of Square Errors, Chellappa [39],

These methods will be discussed later in Section 4.3. A major difficulty with using GMRF models is the selection of parameters β for which the correlation matrix $\mathbf{B}^{-1}$ is positive definite.

The following sampling algorithm was proposed by Chellappa [39]. Define the (i, j) entry of matrix $\mathbf{A}$ w.r.t. $\mathbf{B}$ as,

$$A(i, j) = B(1, j + (i - 1)m) \tag{4.16}$$

where $\mathbf{A}$ is an $m \times m$ matrix, *i.e.,* for the above example

$$\mathbf{A} = \begin{bmatrix} 1 & -\beta_1 & 0 & . & . & 0 & -\beta_1 \\ -\beta_2 & -\beta_3 & 0 & . & . & 0 & -\beta_4 \\ 0 & 0 & 0 & . & . & 0 & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & 0 & 0 \\ -\beta_2 & -\beta_4 & 0 & . & . & 0 & -\beta_3 \end{bmatrix} \tag{4.17}$$

Algorithm for Sampling a GMRF

Step 1 Generate an $m \times m$ array η with i.i.d.element from $N(0, \sigma^2)$.

Step 2 Apply a 2-D FFT on η and save the result in η.

Step 3 Apply a 2-D inverse FFT to $\mathbf{A}$ and save the result in $\mathbf{A}$.

Step 4 Fix colour of pixel (i, j) as:

$$x_{i,j} = \frac{\eta_{i,j}}{\sqrt{A(i, j)}}. \tag{4.18}$$

Step 5 Apply a 2-D inverse FFT to $\mathbf{x}$ and save the result in $\mathbf{x}$.

Step 6 $\mathbf{x} + \mu$ is a sample of the GMRF.

4.3 Parameter estimation

We assume that the conditional distributions $\Pi_s(x_s|\mathbf{x}_{(s)})$ are of a given functional form but collectively contain a number of unknown parameters $\theta = (\alpha, \beta^{\mathrm{T}})^{\mathrm{T}}$ whose values are to be estimated on the basis of a single realisation of, $\mathbf{x}$, of the system. To emphasise this we shall write

$$\Pi_s(x_s|\mathbf{x}_{(s)};\theta) = \frac{\exp\left\{\sum_{C\in\mathcal{C}_s} V_C(\mathbf{x};\theta)\right\}}{\sum_{\lambda_s\in\Lambda}\exp\left\{\sum_{C\in\mathcal{C}_s} V_C(\lambda_s,\mathbf{x}_{(s)};\theta)\right\}} \tag{4.19}$$

We require the optimisation of

$$\Pi(\mathbf{x};\theta) = \frac{1}{Z}\exp\left\{\sum_{C\in\mathcal{C}} V_C(\mathbf{x};\theta)\right\} \tag{4.20}$$

$$Z = \sum_{\mathbf{y}\in\Omega}\exp\left\{\sum_{C\in\mathcal{C}} V_C(\mathbf{y};\theta)\right\} \tag{4.21}$$

such that the value of $\Pi(\mathbf{x};\theta)$ is maximised for the single realisation $\mathbf{x}$. This is what Maximum Likelihood Estimation tries to achieve, however it is hindered by the partition function Z which is usually computationally intractable. This also makes it difficult to avoid phase transitions, which occur when there are near optimal realisations $\mathbf{y}$ for which $\Pi(\mathbf{y};\theta)$ is a local maximum.

An alternative parameter estimation technique is Besag's Maximum Pseudo Likelihood Estimation which optimises the conditional distributions $\Pi_s(x_s|\mathbf{x}_{(s)};\theta)$. However, because this estimate only optimises the local properties, it can not guarantee that any long-range dependence present within the image $\mathbf{x}$ will be accounted for.

4.3.1 Maximum likelihood estimator

This explanation of the Maximum Likelihood Estimator (MLE) comes from Seymour [181]. Although this explanation assumes that the energy function has only single and pairwise potentials, *i.e.,* an auto-model, it will be seen that this will also work for higher order models. First the energy function shall be defined as,

$$U(\mathbf{x};\theta) = -\sum_s V(x_s) - \sum_{s<r}\sum V(x_s,x_r)$$

$$= -\sum_{s} \alpha\vartheta_1(x_s) - \sum_{s<r}\sum \beta_{sr}\vartheta_2(x_s, x_r) \tag{4.22}$$

where $\vartheta_1 : \Lambda \to \Re$ is a known function, and $\vartheta_2 : \Lambda \times \Lambda \to \Re$ is some known symmetric function. The unknown parameters are α and β_{sr}. The β_{sr} functions are also symmetric, *i.e.,* $\beta_{rs} = \beta_{sr}$. For a stationary field, β_{sr} functions are independent of s, so we can re-index them as β_{s-r}, therefore

$$\begin{array}{lcll} \beta_j & = & \beta_{-j} & \text{where} \qquad 0 < \|j\| \leq o \\ \beta_j & = & 0 & \text{otherwise} \end{array} \tag{4.23}$$

where o is the order of the neighbourhood $\mathcal{N}_s$. The likelihood function is defined as

$$\mathcal{L}(\mathbf{x};\theta) \doteq \Pi(\mathbf{x};\theta) = \frac{\exp[-U(\mathbf{x};\theta)]}{Z(\theta)}, \tag{4.24}$$

where partition function is again defined as

$$Z(\theta) = \sum_{\mathbf{y}\in\Omega} \exp[-U(\mathbf{y};\theta)]. \tag{4.25}$$

Any value that maximises $\mathcal{L}(\mathbf{x};\cdot)$ is called the maximum likelihood estimate (MLE) of the parameter θ. This value is found by first taking the derivative of $\mathcal{L}(\mathbf{x};\theta)$ w.r.t. θ and then equating the result to zero and solving for θ. The same result can be obtained for the log of the likelihood.

$$l(\mathbf{x};\theta) = \log \mathcal{L}(\mathbf{x};\theta) = \log \Pi(\mathbf{x};\theta) = -U(\mathbf{x};\theta) - \log Z(\theta) \tag{4.26}$$

It is common to use the log-likelihood function, because the derivative of Eq. (4.26) is easier to calculate. The likelihood function in Eq. (4.24) may be written as an exponential family, so that the log-likelihood may be written as

$$l(\mathbf{x};\theta) = \theta^{\mathrm{T}} Y(\mathbf{x}) - b(\theta) \tag{4.27}$$

Recall that θ is a column vector $(\alpha, \beta^{\mathrm{T}})^{\mathrm{T}}$. By equating Eqs. (4.26) and (4.27), we get

$$\theta^{\mathrm{T}} Y(\mathbf{x}) - b(\theta) = -U(\mathbf{x};\theta) - \log Z(\theta) \tag{4.28}$$

Define

$$\theta^{\mathrm{T}} Y(\mathbf{x}) = -U(\mathbf{x};\theta), \tag{4.29}$$

$$b(\theta) = \log Z(\theta) \tag{4.30}$$

First we will need to find the components of $Y(\mathbf{x})$. From Eqs. (4.22) and (4.29) we obtain the following equation.

$$\begin{aligned} \theta^{\mathrm{T}} Y(\mathbf{x}) &= \sum_s \alpha \vartheta_1(x_s) + \sum_{s<r} \sum \beta_{s-r} \vartheta_2(x_s, x_r) \\ &= \alpha \sum_s \vartheta_1(x_s) + \sum_{j>0} \beta_j \sum_s \vartheta_2(x_s, x_{s+j}) \end{aligned} \tag{4.31}$$

where j is an index as shown in Fig. 4.2. Remember, because the field is stationary, $\beta_j = \beta_{-j}$, so by only summing over the neighbours for which $j > 0$, each clique only occurs once in the summation.

The function $b(\theta)$ may now be derived. From Eq. (4.30) we have

$$\begin{aligned} b(\theta) &= \log Z(\theta) \\ &= \log \sum_{\mathbf{y} \in \Omega} \exp[-U(\mathbf{y}; \theta)] \\ &= \log \sum_{\mathbf{y} \in \Omega} \exp[\theta^{\mathrm{T}} Y(\mathbf{y})] \end{aligned} \tag{4.32}$$

In order to calculate the derivative of the log-likelihood $l(\mathbf{x}; \theta)$ we will need to know the gradient of $b(\theta)$.

$$\begin{aligned} \nabla b(\theta) &= \frac{1}{\sum_{\mathbf{z} \in \Omega} \exp[\theta^{\mathrm{T}} Y(\mathbf{z})]} \sum_{\mathbf{y} \in \Omega} Y(\mathbf{y}) \exp[\theta^{\mathrm{T}} Y(\mathbf{y})] \\ &= \sum_{\mathbf{y} \in \Omega} Y(\mathbf{y}) \frac{\exp[\theta^{\mathrm{T}} Y(\mathbf{y})]}{\sum_{\mathbf{z} \in \Omega} \exp[\theta^{\mathrm{T}} Y(\mathbf{z})]} \\ &= E_\theta(Y(\mathbf{y})) \end{aligned} \tag{4.33}$$

where $E_\theta(Y(\mathbf{y}))$ is the expected value of $Y(\mathbf{y})$ given θ. The derivative of the likelihood function $l(\mathbf{x}; \theta)$, as expressed by Eq. (4.27), is

$$\frac{\partial}{\partial \theta} l(\mathbf{x}; \theta) = Y(\mathbf{x}) - E_\theta(Y(\mathbf{y})) \tag{4.34}$$

By equating $\frac{\partial}{\partial \theta} l(\mathbf{x}; \theta)$ to zero, the likelihood equation may be written as

$$Y(\mathbf{x}) = E_\theta(Y(\mathbf{y})) \tag{4.35}$$

which needs to be solved for θ. This can be done using a numerical minimising routine like the Newton-Raphson method.

Seymour [181] showed the existence, uniqueness, and consistency of the MLE, but in spite of this, the MLE is virtually useless for practical purposes because the partition function $Z(\theta)$ is computationally intractable. However it does provide a building block for less accurate, but easier to compute likelihood estimators.

4.3.2 Monte Carlo maximum likelihood estimator

Again this explanation comes from Seymour [181]. It was originally developed by Geyer and Thompson [84].

Let Π_ψ be the Gibbs distribution with *known* parameter ψ. Simulate an ergodic Markov chain $\{\mathbf{X}(0), \mathbf{X}(1), \ldots, \mathbf{X}(n-1)\}$ of random fields on Ω whose equilibrium distribution is Π_ψ, see Section 4.4. Write the likelihood in the form

$$\mathcal{L}(\mathbf{x}; \theta) = \frac{\exp[\theta^{\mathrm{T}} Y(\mathbf{x})]}{c(\theta)} \tag{4.36}$$

where

$$c(\theta) = Z(\theta) = \sum_{\mathbf{y} \in \Omega} \exp[\theta^{\mathrm{T}} Y(\mathbf{y})] \tag{4.37}$$

Manipulate $c(\theta)$ into an expectation:

$$\begin{aligned} c(\theta) &= \sum_{\mathbf{y} \in \Omega} \exp[(\theta - \psi)^{\mathrm{T}} Y(\mathbf{y})] \exp[\psi^{\mathrm{T}} Y(\mathbf{y})] \\ &= c(\psi) \sum_{\mathbf{y} \in \Omega} \exp[(\theta - \psi)^{\mathrm{T}} Y(\mathbf{y})] \frac{\exp[\psi^{\mathrm{T}} Y(\mathbf{y})]}{c(\psi)} \\ &= c(\psi) \sum_{\mathbf{y} \in \Omega} \exp[(\theta - \psi)^{\mathrm{T}} Y(\mathbf{y})] \Pi_\psi(\mathbf{y}) \\ &= c(\psi) E_\psi \left[\exp[(\theta - \psi)^{\mathrm{T}} Y(\mathbf{y})] \right] \end{aligned} \tag{4.38}$$

We may then define the ratio

$$r(\theta) = \frac{c(\theta)}{c(\psi)} = E_\psi \left[\exp[(\theta - \psi)^{\mathrm{T}} Y(\mathbf{y})] \right] \tag{4.39}$$

so that the log-likelihood, to within a multiplicative constant, may now be written

as

$$l(\mathbf{x};\theta) = \log[c(\psi)\mathcal{L}(\mathbf{x};\theta)] = \theta^{\mathrm{T}} Y(\mathbf{x}) - \log r(\theta). \tag{4.40}$$

Using the simulated Markov chain, define $r_n(\theta)$ as

$$r_n(\theta) = \frac{1}{n}\sum_{i=0}^{n-1} \exp[(\theta-\psi)^{\mathrm{T}} Y(\mathbf{X}(i))] \tag{4.41}$$

Due to the ergodicity of the Markov chain, we have $r_n(\theta) \to r(\theta)$ as $n \to \infty$. Thus the Monte Carlo approximation to the log-likelihood Eq. (4.40) is given by

$$l_n(\mathbf{x};\theta) = \theta^{\mathrm{T}} Y(\mathbf{x}) - \log r_n(\theta) \tag{4.42}$$

We call the value which maximises $l_n(\mathbf{x};\cdot)$ a Monte Carlo MLE (MCMLE). There are two important issues to consider when using the MCMLE as an estimate of the true parameter. One is that n, the number of Markov chain samples, should be much larger than N the size of the image, to provide a good estimate of θ. Second, Seymour [181] found that the arbitrary parameter ψ had to be chosen close to the true parameter θ so that the number of Markov chain Monte Carlo samples required was not prohibitive. In fact if ψ differed from θ by more than 10%, the estimation of θ could not improve upon ψ itself. This was for a chain length of 500.

4.3.3 Maximum pseudo-likelihood estimator

An alternative to the MLE was developed by Besag [19, 20] called Maximum Pseudo-Likelihood Estimator (MPLE). Its predecessor was the Coding Scheme, Section 4.3.4, also developed by Besag [17]. This explanation of the MPLE again comes from Seymour [181]. The pseudo-likelihood which Besag proposed is simply the product of the local probabilities of the sites in S.

$$\begin{aligned} \mathcal{PL}(\mathbf{x};\theta) &= \prod_{s\in S} \Pi_s(x_s|\mathbf{x}_{(s)};\theta) \\ &= \prod_{s\in S} \frac{\exp[-U_s(\mathbf{x};\theta)]}{Z_s(\theta)} \end{aligned} \tag{4.43}$$

where

$$Z_s(\theta) = \sum_{\lambda_s\in\Lambda} \exp[-U_s(\lambda_s, \mathbf{x}_{(s)};\theta)] \tag{4.44}$$

To find the maximum pseudo-likelihood estimate, we follow the same steps as for the MLE. Therefore we need to solve for θ for when the derivative of the pseudo-likelihood function is equal to zero. Again it is easier if we use the log-pseudo-likelihood function, which will give the same result.

$$\begin{aligned} l(\mathbf{x};\theta) &= \log \mathcal{PL}(\mathbf{x};\theta) \\ &= -\sum_{s\in S} U_s(\mathbf{x};\theta) - \sum_{s\in S} \log Z_s(\theta) \end{aligned} \tag{4.45}$$

If we assume that $U_s(\mathbf{x};\theta)$ is only second order, as for an Auto-model, then $U_s(\mathbf{x};\theta)$ can be expressed in the following form

$$U_s(\mathbf{x};\theta) = -\alpha\vartheta_1(x_s) - \sum_{r\in\mathcal{N}_s} \beta_{s-r}\vartheta_2(x_s, x_{s+r}), \tag{4.46}$$

where the variables are the same as in Eq. (4.22). We may then write the first part of Eq. (4.45) as

$$\begin{aligned} \theta^{\mathrm{T}} Y(\mathbf{x}) &= -\sum_{s\in S} U_s(\mathbf{x};\theta) \\ &= \sum_{s\in S} \alpha\vartheta_1(x_s) + \sum_{s\in S}\sum_{r\in\mathcal{N}_s} \beta_{s-r}\vartheta_2(x_s, x_{s+r}) \\ &= \alpha \sum_{s\in S} \vartheta_1(x_s) + \sum_{r\in\mathcal{N}_0} \beta_r \sum_{s\in S} \vartheta_2(x_s, x_{s+r}) \end{aligned} \tag{4.47}$$

Therefore the derivative of the first part of Eq. (4.45) is just $Y(\mathbf{x})$. However notice that this $Y(\mathbf{x})$ is different to the $Y(\mathbf{x})$ of Eq. (4.31) calculated for the MLE. We can now express the second part of Eq. (4.45) as,

$$\begin{aligned} g(\theta) &= \sum_{s\in S} \log Z_s(\theta) \\ &= \sum_{s\in S} \log \sum_{\lambda_s\in\Lambda} \exp[-U_s(\lambda_s, \mathbf{x}_{(s)};\theta)] \\ &= \sum_{s\in S} \log \sum_{\lambda_s\in\Lambda} \exp[\theta^{\mathrm{T}} Y_s(\lambda_s, \mathbf{x}_{(s)})] \end{aligned} \tag{4.48}$$

$$\tag{4.49}$$

where Y_s is the known components of the potentials associated with site s. Note that

$$Y(\mathbf{x}) = \sum_{s \in S} Y_s(\mathbf{x}) \tag{4.50}$$

We now need to calculate the derivative of $g(\theta)$.

$$\begin{aligned} \nabla g(\theta) &= \sum_{s \in S} \frac{1}{\sum_{\tau_s \in \Lambda} \exp[\theta^{\mathrm{T}} Y_s(\tau_s, \mathbf{x}_{(s)})]} \sum_{\lambda_s \in \Lambda} Y_s(\lambda_s, \mathbf{x}_{(s)}) \exp[\theta^{\mathrm{T}} Y_s(\lambda_s, \mathbf{x}_{(s)})] \\ &= \sum_{s \in S} \sum_{\lambda_s \in \Lambda} Y_s(\lambda_s, \mathbf{x}_{(s)}) \frac{\exp[\theta^{\mathrm{T}} Y_s(\lambda_s, \mathbf{x}_{(s)})]}{\sum_{\tau_s \in \Lambda} \exp[\theta^{\mathrm{T}} Y_s(\tau_s, \mathbf{x}_{(s)})]} \\ &= \sum_{s \in S} E_\theta(Y_s(\lambda_s, \mathbf{x}_{(s)})) \end{aligned} \tag{4.51}$$

Analogous to the likelihood Eq. (4.35), the corresponding pseudo-likelihood equation is

$$Y(\mathbf{x}) = \sum_{s \in S} E_\theta(Y_s(\lambda_s, \mathbf{x}_{(s)})) \tag{4.52}$$

which needs to be solved for θ, again through some kind of numerical minimising routine like the Newton-Raphson method.

The advantage of the MPLE is that the partition functions for the single sites are easily computed. Geman and Graffigne [81] also established the existence, uniqueness, and consistency of the MPLE under very general conditions. However, because the MPLE only optimises the local properties, it may not deal well with images that have long range dependence. This limitation was demonstrated by experiments conducted by Seymour [181]. However, out of all the estimators, the general consensus seems to point to the MPLE as the estimator most likely to give reasonable and consistent results [181].

4.3.4 Coding scheme and other estimators

The coding scheme of Besag [17], is based on the observation that given two sites s and r that are conditionally independent of each other, then by the Bayes formula and the Markov property,

$$P(x_s, x_r | t \in \mathcal{N}_s, t \in \mathcal{N}_r) = P(x_s | t \in \mathcal{N}_s) P(x_r | t \in \mathcal{N}_r) \tag{4.53}$$

Let $S_j \subseteq S$ such that every site in S_j is conditionally independent of every other site in S_j. Consider a first-order neighbourhood, Fig. 4.3 shows a labelling of sites for S_j, where '$\times$'$\in S_j$ and '.'$\notin S_j$.

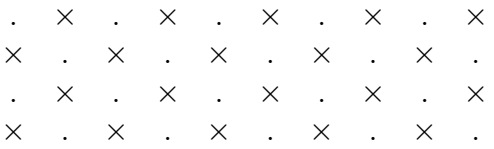


Figure 4.3: Coding Pattern for a first-order neighbourhood

The coding method defines the parameter estimate $\hat{\theta}$ as follows. First find the parameters $\hat{\theta}_j$ which maximise the log-likelihood $L_j(\mathbf{x};\theta)$ for the sites S_j.

$$L_j(\mathbf{x};\theta) = \sum_{s\in S_j} \log[\Pi_s(x_s|\mathbf{x}_{(s)};\theta)] = \sum_{s\in S_j} \log\left[\frac{\exp\left\{\sum_{C\in\mathcal{C}_s} V_C(\mathbf{x};\theta)\right\}}{\sum_{\lambda_s\in\Lambda}\exp\left\{\sum_{C\in\mathcal{C}_s} V_C(\lambda_s,\mathbf{x}_{(s)};\theta)\right\}}\right] \tag{4.54}$$

Again Eq. (4.52) can be applied and the log-likelihood $L_j(\mathbf{x};\theta)$ can be optimised via the Newton-Raphson method. An estimated parameter $\hat{\theta}_j$ is obtained for each separate set $S_j \subseteq S$, whereby the overall parameter estimate $\hat{\theta}$ is given by the average of the estimates $\hat{\theta}_j$.

Cross and Jain [50] used this method for binary images and found that it produced consistent results. However Derin and Elliott [54] used the same method on grey level images and found it produced unreliable results. Besag [18] considered the coding method to be inefficient and therefore proposed the pseudo-likelihood estimator.

Other attempts at finding better estimators have been made. Derin and Elliott [54] proposed the *Least-Squares Error Estimator*, but their method was dependent on subjective criteria, and it did not extend well for images with many grey levels. Chen [43] tried to upgrade the Least-squares Error Estimator by proposing the *Logit Model Fit Estimator*, but this also suffered from inefficiencies when modelling images with many grey levels. However Chen and Dubes [42] found that the best estimator for binary images was the *Minimum Logit* χ^2 *Estimator* [16].

A specific estimator for the GMRF model was proposed by Chellappa [39] called *Sum of Square Errors Minimisation*. However it was later shown by Bader, JáJá and Chellappa [6] that the maximum likelihood estimator performed better than sum

of square errors minimisation. Another specific estimator is *Method-of-Moments* by Possolo [165], which is valid for stationary fields where dependence weakens rapidly with increasing distance.

4.4 Sampling

A texture is synthesised from an MRF model by sampling $\mathbf{x} \in \Omega$ with respect to the joint distribution Π. If the model has captured the complete characteristics of the training texture then most of the "mass" associated with the joint distribution will centre around those images $\mathbf{x} \in \Omega$ which display the same texture as the one modelled. If this is the case, then a sampling of the joint distribution Π will most likely produce a texture similar to the modelled texture. However, the joint distribution Π has only been created from one sample ($\mathbf{Y} = \mathbf{y}$). The amount by which the "mass" of the joint distribution Π clusters around similar textures $\mathbf{x} \in \Omega$ will depend on how the training texture $\mathbf{y} \in \Omega$ was modelled.

For many reasons the joint distribution Π cannot be directly sampled. One reason is that the sample space Ω is usually prohibitively large to perform a direct sampling. Therefore synthesis algorithms employ the fact that the joint distribution Π is uniquely determined by the LCPDFs [17]. These algorithms operate by generating a Markov chain of images $\{\mathbf{X}(0), \mathbf{X}(1), \ldots, \mathbf{X}(n)\}$ which converge to an image $\mathbf{x} \in \Omega$ such that,

$$\lim_{n\to\infty} P(\mathbf{X}(n) = \mathbf{x} | \mathbf{X}(0) = \mathbf{x}(0)) = \Pi(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega, \tag{4.55}$$

irrespective of the initialised image $\mathbf{X}(0) = \mathbf{x}(0)$ at the beginning of the Markov chain [82]. The initialising image will normally be drawn from a computer random number generator and will have a distribution given by that process. Typically each image $\mathbf{x}(0)$ will be uncorrelated and come from a uniform distribution.

Stochastic relaxation (SR) is one method of generating a Markov chain [50, 80, 82]. Two well known SR algorithms are the *Metropolis* algorithm [43, 142], as given in Fig. 4.4, and the *Gibbs Sampler* [78, 82], as given in Fig. 4.5. These algorithms operate by replicating all but one of the pixel values from an image in the Markov chain to the next. The pixel value that is not replicated is updated with a pixel value defined by the LCPDF. Convergence to $\Pi(\mathbf{x})$ occurs if, as the length of the Markov chain increases to infinity, the number of times that a single pixel is updated

also tends to infinity [80, 82].

Metropolis Algorithm

1. Fix a random site-visitation schedule $\{a_k \in S, k = 1, 2, \ldots\}$, such that for each $s \in S$, $a_k = s$ infinitely often.

2. Randomly choose $\mathbf{x}(0)$.

3. **For** $k = 1$ to ∞ **do**

3.1. Randomly choose $\lambda_{a_k} \in \Lambda$.

3.2. Let

$$p = \min\left\{1, \frac{P(\lambda_{a_k}|x_r(k-1), r \in \mathcal{N}_{a_k})}{P(x_{a_k}(k-1)|x_r(k-1), r \in \mathcal{N}_{a_k})}\right\}$$

3.3. Randomly choose $q \in [0 - 1)$

3.4. The image $\mathbf{X}(k)$ is obtained from the previous image $\mathbf{X}(k-1)$ such that,

$$X_s(k) = \begin{cases} \lambda_{a_k}, & \text{if } s = a_k \text{ and } p > q \\ x_s(k-1), & \text{otherwise} \end{cases}$$

4. **Done**

Figure 4.4: Metropolis algorithm

SR algorithms iteratively generate a succession of images which converge to an image with a probability given by the joint distribution Π of an MRF [82]. However the desired sample image is only theoretically reached in the limit as the number of iterations of the SR algorithm tends to infinity, Eq. (4.55). In practice SR algorithms have to terminate. The number of iterations required to obtain equilibrium largely depends on the LCPDF and therefore the training texture ($\mathbf{Y} = \mathbf{y}$). Relaxation algorithms whose termination point does not have to be pre-specified are the deterministic relaxation (DR) algorithms. A prime example is Besag's [19] *Iterative Conditional Modes* (ICM) algorithm, as given in Fig. 4.6. In this case, the pixel whose value is to be updated is equated to the mode of its LCPDF.

The essence of every SR algorithm is that the Markov chain has the opportunity to progress to lower probability states such that $\Pi(\mathbf{x}(i)) < \Pi(\mathbf{x}(i-1))$, however such a progression is not allowed in DR. Therefore a DR algorithm will tend to find an

Gibbs Sampler

1. Fix a random site-visitation schedule $\{a_k \in S, k = 1, 2, \ldots\}$, such that for each $s \in S$, $a_k = s$ infinitely often.

2. Randomly choose $\mathbf{x}(0)$.

3. **For** $k = 1$ to ∞ **do**

3.1. Sample $\lambda_{a_k} \in \Lambda$ from the distribution

$$P(\lambda_{a_k} | x_r(k-1), r \in \mathcal{N}_{a_k})$$

3.2. The image $\mathbf{X}(k)$ is obtained from the previous image $\mathbf{X}(k-1)$ such that,

$$X_s(k) = \begin{cases} \lambda_{a_k}, & \text{if } s = a_k \\ x_s(k-1), & \text{otherwise} \end{cases}$$

4. **Done**

Figure 4.5: Gibbs Sampler

equilibrium state at a local maximum, generally somewhere close to the initial image $\mathbf{X}(0) = \mathbf{x}(0)$. When the ICM algorithm reaches equilibrium at a local maximum of Π, further iterations via the ICM algorithm will not change the synthesised image. This is because at the local maximum all the pixel values in the image will be equal to the modes of their respective LCPDFs,

$$x_s = \arg\max_{\lambda_s \in \Lambda} P(\lambda_s | x_r, r \in \mathcal{N}_s) \qquad \forall s \in S. \tag{4.56}$$

Therefore the ICM algorithm will have effectively reached its own termination point.

If the texture is sufficiently well defined by the model, then the "mass" of the joint distribution Π should cluster around just those images $\mathbf{x} \in \Omega$ which represent textures similar to the training texture. To test that this is the case, the joint distribution Π should be repeatedly sampled via a synthesis algorithm that seeks a highly likely representation. If the synthesised textures are then generally objectively similar to the training texture, it can then be said that most of the mass of Π clusters around those images $\mathbf{x} \in \Omega$ which represent textures similar to the training texture. The texture is then well modelled by the LCPDF. As the ICM algorithm is prone to

Iterative Conditional Modes

1. Fix a random site-visitation schedule $\{a_k \in S, k = 1, 2, \ldots\}$, such that for each $s \in S$, $a_k = s$ infinitely often.

2. Randomly choose $\mathbf{x}(0)$.

3. **For** $k = 1$ to ∞ **do**

3.1. Let

$$\lambda_{a_k} = \arg\max_{\lambda_{a_k} \in \Lambda} P(\lambda_{a_k} | x_r(k-1), r \in \mathcal{N}_{a_k})$$

3.2. The image $\mathbf{X}(k)$ is obtained from the previous image $\mathbf{X}(k-1)$ such that,

$$X_s(k) = \begin{cases} \lambda_{a_k}, & \text{if } s = a_k \\ x_s(k-1), & \text{otherwise} \end{cases}$$

4. **Done**

Figure 4.6: Iterative Conditional Modes

acquiring a representation at a local maxima of Π, it may not be generally suitable for this type of test. However, later in Chapter 7 when multiscale texture synthesis is discussed, it is determined that the ICM algorithm is acceptable if a multiscale texture synthesis algorithm is used.

4.4.1 Optimisation by simulated annealing

Simulated annealing is not essential for texture synthesis, as an MRF texture can be generated by simply sampling from the distribution Π. However this is a good point at which to introduce the concept of a temperature variable. This concept will be later refined for our own application of local annealing. Most of the following work comes from Geman [78, 82]. Consider an MRF with Gibbs distribution,

$$\Pi(\mathbf{x}) = \frac{1}{Z} e^{-U(\mathbf{x})}, \qquad \mathbf{x} \in \Omega \tag{4.57}$$

we wish to find the $\mathbf{x}$ that maximises $\Pi(\mathbf{x})$, or in other words, those *ground states* $\Omega_{\min}$ whereby,

$$\Omega_{\min} = \{\mathbf{x} \in \Omega : U(\mathbf{x}) = \min_{\mathbf{x}\in\Omega} U(\mathbf{x})\}, \tag{4.58}$$

Kirkpatrick *et al.* [122] and Cerny [33] made the analogy between this system and a physical system in statistical physics. To find the ground states in such a system a process of annealing is used, in which the interacting substance of the system is initially heated and then slowly cooled, allowing equilibrium to be reached at each successively lower temperature. A typical physical application is the working of metal, also certain crystals are grown in this way. Annealing is a process designed to obtain the perfect ground state of a system.

In order for the Gibbs distribution to mimic the annealing process we introduce a temperature variable T and redefine the Gibbs distribution as,

$$\Pi_T(\mathbf{x}) = \frac{1}{Z_T} e^{-U(\mathbf{x})/T}. \tag{4.59}$$

Therefore,

$$\lim_{T\to\infty} \Pi_T(\mathbf{x}) = \frac{1}{\|\Omega\|} \tag{4.60}$$

That is at high temperatures $\Pi_T(\mathbf{x})$ tends to a uniform distribution and all states $\mathbf{x} \in \Omega$ have the same probability of occurring. As the temperature decreases and approaches 1, then $\Pi_T(\mathbf{x}) \to \Pi(\mathbf{x})$ the original distribution. However if the temperature is lowered even further we have,

$$\lim_{T\to 0} \Pi_T(\mathbf{x}) = \Pi_0(\mathbf{x}) \doteq \begin{cases} 0, & \mathbf{x} \notin \Omega_{\min} \\ \frac{1}{\|\Omega_{\min}\|}, & \mathbf{x} \in \Omega_{\min} \end{cases} \tag{4.61}$$

Using the distribution $\Pi_0(\mathbf{x})$ is equivalent to applying Besag's [19] ICM algorithm, as given in Fig. 4.6.

Annealing requires a Monte Carlo simulation at each temperature. The aim is to reach equilibrium, Eq. (4.55), in the Markov chain before descending to the new temperature. If the rate of decent is slow enough, then obtaining $\mathbf{x} \in \Omega_{\min}$ is guaranteed. What is required is a *cooling schedule* to determine how long the Markov chain should be at each temperature, and how much the temperature should be decreased at the end of each Markov chain.

Theorem 4.1 (Cooling Schedule) *Given that* $S = \{s_1, s_2, \ldots, s_N\}$*, assume that*

there exists an integer $\tau \geq N$ *so that* $\{\mathbf{X}((k-1)\tau+1), \mathbf{X}((k-1)\tau+2), \ldots, \mathbf{X}(k\tau)\}$ *is a Markov chain at temperature* T_k*. Then*

$$\lim_{k\to\infty} P(\mathbf{X}(k\tau)) = \mathbf{x}|\mathbf{X}(0) = \mathbf{x}(0)) = \Pi_0(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega, \tag{4.62}$$

if T_k *is any decreasing sequence of temperatures for which*

1. $T_k \to 0$ *as* $k \to \infty$*;*
2. $T_k \geq \frac{N\Delta}{\log k}$ $\forall k \geq 2$

and

$$\Delta = \max_{s\in S;\mathbf{x}\in\Omega} [\max_{\lambda_s\in\Lambda} U_s(\lambda_s, \mathbf{x}_{(s)}) - \min_{\lambda_s\in\Lambda} U_s(\lambda_s, \mathbf{x}_{(s)})] \tag{4.63}$$

Proof : See Geman and Geman [82], Theorem B.

□

In practice $N\Delta$ is too large for implementation. Instead an approximation is used

$$T_k = \frac{C}{\log(1+k)}, \qquad k \geq 1 \tag{4.64}$$

where $C \ll N\Delta$. Geman and Geman [82] used $C = 3.0$ or $C = 4.0$, and $\tau = N$ such that every site $s \in S$ is visited in the Markov chain.

4.5 Goodness-of-fit testing

A goodness-of-fit test is required to test the hypothesis that the observed training texture is an expected realisation of the MRF model. An obvious way of testing this hypothesis that the texture model has successfully captured the characteristics of the texture, is to use the model to synthesise a set of realisations. Success can then be assessed in terms of how subjectively similar the synthetic textures are to the original training texture. However care must be taken when subjective analysis is used, because as mentioned in "Numerical Recipes in C" [167] it is deemed courageous to just rely on the model "looking" like it fits the data (*i.e.,* texture). This is regarded as *chi-by-eye* and can be a big trap to the unwary.

There have been some attempts at forming an objective approach to goodness-of-fit testing for texture. Cross and Jain [50] used a chi-square test to determine

if the observed training texture fitted the data distribution defined by the estimated LCPDF. However this can easily turn into an unrealistic task as at least five observations are usually desired for each LCPDF configuration for chi-square testing [67, 169].

Alternatively Dubes and Jain [59] proposed that texture features could be used to measure statistics from both the synthetic textures and the training texture. Then a ranking test could be used to determine if the measured statistics from the synthetic textures came from the same population as those taken from the training texture. However for this type of goodness-of-fit test to be advantageous, the texture features should encompass the complete characteristics of the texture. The problem is, this is exactly what the model itself is trying to achieve.

The reason for applying a goodness-of-fit test to a proposed model is that it not only allows for an evaluation of how well the model fits the texture, but it also allows for the ability to choose the best model from a range of models. In general we require that the model adequately fits the texture, but is not over parameterised. When a model is over parameterised, it is in effect trying to extract more information from the texture than can be expected to exist [211]. Smith and Miller [188] proposed a model selection criterion for MRFs based on the stochastic complexity of Rissanen [176]. A more popular criterion, which is useful for exponential families, is the information criterion of Akaike [4] called AIC. However, the AIC tends to over-parameterise, and does not give a consistent estimate [208]. Bayesian modification of Akaike's information criterion (BIC) has a larger penalty term that makes it less prone to over-parameterisation [5, 179] and more consistent [208].

Seymour [181] took model selection a step further. She noted the connection between parameter estimation and goodness-of-fit testing. In her thesis she presented an approach for model selection based on the parameter estimation routines of MLE, MCMLE, and MPLE. Unfortunately, although this approach was a marked improvement over the AIC and BIC for model selection, it was still limited by the parametric models ability to correctly model the texture. As Seymour [181] mentions, there is still no efficient systematic way of determining the cliques and respective potential functions of a parametric MRF for modelling a particular texture.

Chapter 5

Nonparametric MRF Model

This chapter gives the construction details of our nonparametric MRF model, plus some alternative approaches.

5.1 Introduction

An MRF is modelled by its joint probability distribution Π. In texture analysis usually only one training image ($\mathbf{Y} = \mathbf{y}$) of a texture is given from which to estimate Π. It is desirable that the majority of the "mass" of Π be distributed among those images, $\mathbf{x} \in \Omega$, which are subjectively similar to the training image $\mathbf{y}$. To accomplish this with only one training image, two properties of an MRF are used. The first property is given by Besag [17] and states that Π can be uniquely expressed in terms of its LCPDFs for which the neighbourhoods consist of the rest of the image. However this still means we have only one single independent and identically distributed (i.i.d.) data sample. The second property of an MRF is that the neighbourhood of an LCPDF may be adequately defined over a smaller subset of sites. For a homogeneous MRF, the LCPDF may now be modelled from a set of i.i.d. sample data obtained from a subset of sites $S' \subset S$ and their neighbourhoods $\{\mathcal{N}_s, s \in S'\}$ for which the LCPDFs $P(\lambda_s | x_r, r \in \mathcal{N}_s)\ \forall s \in S'$ are not conditional on any sites $r \in S'$. Therefore the set of sites is $S' = \{s \in S : r \notin S', \forall r \in \mathcal{N}_s\}$. Given a large enough number of i.i.d. sample data, the LCPDF may be modelled well enough to produce a reasonable estimate of the joint probability distribution Π.

The coding scheme, described in Besag [17] and Section 4.3.4, identifies how a

set of sites S may be reduced to separate subsets $S' \subset S$ over which i.i.d. sample data may be obtained. Besag [17] proposed that via this method, an estimate of the LCPDF could be obtained from each subset $S' \subset S$, with the final estimate of the LCPDF being the average of these. Cross and Jain [50] used this method for binary images and found that it produced consistent results. However Derin and Elliott [54] used the same method on grey level images with unreliable results.

Besag [18] considered the coding method to be inefficient and therefore proposed the pseudo-likelihood estimator, Section 4.3.3. Instead of using separate sets of i.i.d. data, the pseudo-likelihood estimator uses all the data from the complete set S to obtain an estimate of the LCPDF. Even though the combined data is no longer i.i.d., Besag [20] showed that this is an efficient technique for estimating the LCPDF of a GMRF. Geman and Graffigne [81] also proved that the pseudo-likelihood estimate converged to the true LCPDF with probability one as the size of the field S tended to infinity. With this evidence we justify our use of non i.i.d. sample data for our nonparametric estimate of the LCPDF.

To model the joint distribution Π by its LCPDF, it is first necessary to define the neighbourhood system. However because there is no direct way of determining the correct neighbourhood system from the image sample, we initially assume one. If subsequently the goodness-of-fit test shows this is in adequate, then a different neighbourhood system is chosen.

Assuming a neighbourhood system, an estimate of the LCPDF is based on defining a distribution for the multi-dimensional histogram of a particular homogeneous textured image, Section 5.2. Each dimension of the histogram represents a site from the neighbourhood of the LCPDF with one dimension for the site itself. The total number of dimensions is the statistical order of the model and is equal to the neighbourhood size plus one. Although it would be informative to test larger and larger neighbourhoods for modelling texture, there is a limit to the size which may be successfully modelled. This is due to the *curse of dimensionality* [11], which occurs when modelling with a limited amount of sample data in a high dimensional space. Silverman [183] showed that to maintain accuracy in a model, the amount of sample data needs to grow almost exponentially with the dimensionality of the histogram. As we are dealing with a limited amount of sample data – approximately equal to the number of pixels in a training texture image – the accuracy of the model will rapidly decrease as the dimensionality of the histogram increases.

The parametric approach for estimating the LCPDF from the multi-dimensional

histogram is to use the Gibbs distribution to define the LCPDF in terms of potential functions, Eq. (3.30). Section 4.2 lists various parametric models, for which each potential function is defined in terms of a parametric function and a set of parameters. These parameters are optimised so as to best match the LCPDF to the multi-dimensional histogram (Section 4.3). However the fitting of the LCPDF to the multi-dimensional histogram occurs over the whole domain of the histogram [43, 181].

In such cases, when the domain is large and only sparsely populated with sample data, nonparametric estimation of the LCPDF tends to be more reliable than their parametric counterparts if the function representing the form of the underlying true distribution is unknown [183]. This is because nonparametric estimation only tries to model those areas of the multi-dimensional histogram that contain the data rather than the whole domain as with parametric estimation.

If the general shape of the LCPDF is unknown, then instead of trying to model the unknown density function in the form of a parametric function, it may be more prudent to use a nonparametric density estimator. In this way the true shape of the unknown density function is not compromised by trying to fit the shape of an assumed parametric function to the data. The only catch is that in using a nonparametric density estimator, the LCPDF may no longer define a valid joint distribution Π as required by the equivalence theorem [17]. In any case, parametric density estimation becomes less accurate as the sample space becomes more sparsely populated with increased neighbourhood size.

5.2 Multi-dimensional histogram

Given a training image $\mathbf{y}$ of a homogeneous texture and a predefined neighbourhood system $\mathcal{N}$, a nonparametric estimate of the LCPDF may be obtained by building a multi-dimensional histogram. First denote a pixel value L_0, for which $L_0 \in \Lambda$. Given that L_0 represents the pixel value at a site $s \in S$, denote pixel values $L_{n_r} \in \Lambda$ for each site $r \in \mathcal{N}_s$, where the indices n_r are integers $1 \leq n_r \leq |\mathcal{N}_s|$ representing the relative position of r to s. Then the set of pixel values $\{L_0, \ldots, L_{|\mathcal{N}_s|}\}$ represents a realisation of a pixel and its neighbours irrespective of the pixel location $s \in S$.

Denote $F(L_0, \ldots, L_{|\mathcal{N}_s|})$ as the frequency of occurrence of the set of grey levels

$\{L_0, \ldots, L_{|\mathcal{N}_s|}\}$ in the image $\mathbf{y}$. The frequency is calculated from the image $\mathbf{y}$ as,

$$F(L_0, \ldots, L_{|\mathcal{N}_s|}) = \sum_{\substack{s \in S, \\ \mathcal{N}_s \subset S}} \delta(y_s - L_0) \prod_{r \in \mathcal{N}_s} \delta(y_r - L_{n_r}), \tag{5.1}$$

where δ is the Kronecker function. The set of frequencies

$$F(L_0, \ldots, L_{|\mathcal{N}_s|}) \qquad \forall L_0, \ldots, L_{|\mathcal{N}_s|} \in \Lambda \tag{5.2}$$

is the multi-dimensional histogram, where each $L_n, 0 \leq n \leq |\mathcal{N}_s|$ is a dimension (or axis) of the histogram.

The LCPDF is estimated from the multi-dimensional histogram as,

$$\hat{P}(\lambda_s | x_r, r \in \mathcal{N}_s) = \frac{1}{Z_s} F(L_0 = \lambda_s, L_{n_r} = x_r, r \in \mathcal{N}_s), \qquad \forall \lambda \in \Lambda \tag{5.3}$$

where,

$$Z_s = \sum_{\lambda_s \in \Lambda} F(L_0 = \lambda_s, L_{n_r} = x_r, r \in \mathcal{N}_s). \tag{5.4}$$

As an example, lets choose a neighbourhood system $\mathcal{N} = \{\mathcal{N}_s = \{s-1\}\}$ as shown in Fig. 5.1(a). The estimation of the respective LCPDF is formed by creating a 2-dimensional histogram with respect to $\mathcal{N}$ and the image $\mathbf{y}$. To build the histogram, first label the dimensions as (L_0, L_1), where L_0 represents the pixel value of y_s and L_1 represents the relative neighbouring pixel value of y_{s-1}. Initialise $F(L_0, L_1) = 0 \ \forall L_0, L_1 \in \Lambda$. Then by raster scanning the image $\mathbf{y}$ increment the value of $F(L_0 = y_s, L_1 = y_{s-1})$ for each site $s \in S, \mathcal{N}_s \subset S$. A representation of the multi-dimensional histogram is shown in Fig. 5.1(b). The estimate of the LCPDF is then given by,

$$\hat{P}(x_s | x_{s-1}) = \frac{F(L_0 = x_s, L_1 = x_{s-1})}{\sum_{\lambda_s \in \Lambda} F(L_0 = \lambda_s, L_1 = x_{s-1})}. \tag{5.5}$$

In building the histogram no allowances have been made for making the sample data (L_0, L_1) i.i.d. This means the estimate of the LCPDF is biased. However, as discussed in Section 5.1, we justify our use of non i.i.d. sample data from Besag's [20] and Geman and Graffigne's [81] findings that the pseudo-likelihood estimator – which also uses non i.i.d. sample data – is an efficient estimator of a GMRF and tends to the true estimate as the size of field S tends to infinity.

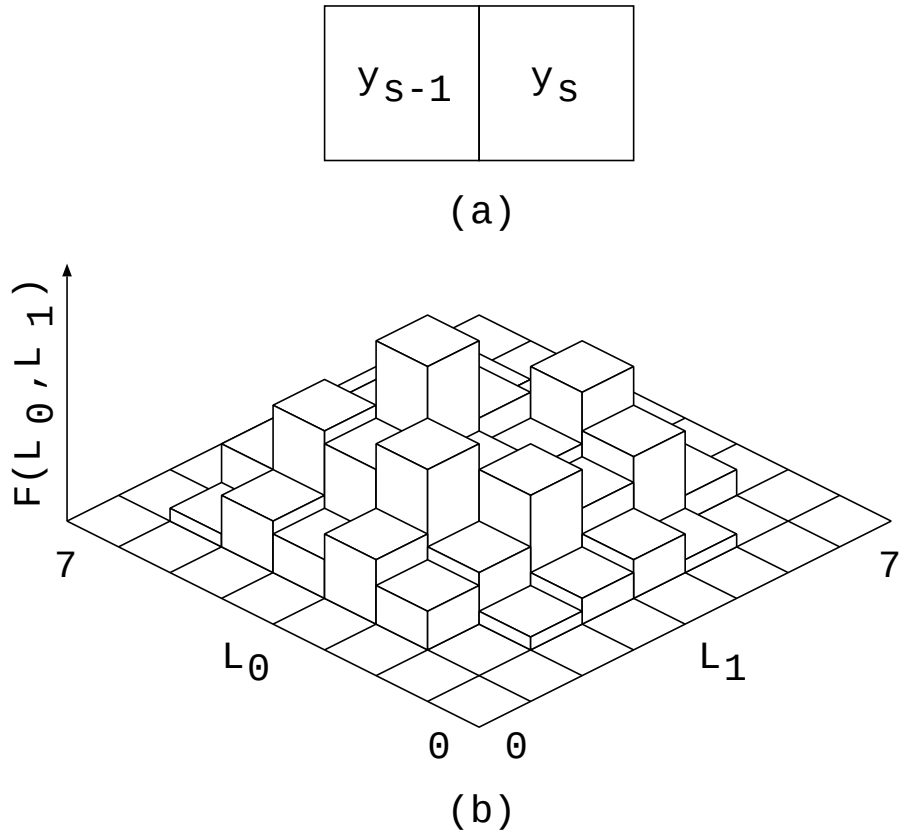


Figure 5.1: A one neighbour neighbourhood and its hypothetical 2-D histogram.

The reliability of the LCPDF estimate via the multi-dimensional histogram is determined by how well the sample data fills the histogram space [183]. Ideally the true probability density function is given by a histogram built from a sample data size approaching infinity, but this is never the case, so a histogram is only an estimate of the probability density function.

Consider the following example for a second order neighbourhood system, order $o = 2$, the one that represents the eight nearest neighbours of a pixel. The histogram required to capture the statistics must be 9-dimensional. Now if the grey levels range from 0–15, then the size of the sample space of the histogram equals $16^9 \approx 6.9 \times 10^{10}$. If we assume that the original image is relatively large, say 1000×1000 pixels, then we would have approximately a million samples to fill this sample space. However the sample data will only fill about 1 in every 70000 histogram bins. Therefore this histogram is a very poor estimate of the probability density function. A random search through this histogram space would reveal little data or structure.

The estimate may be improved by reducing the number of bins (or grey lev-

els), but as the dimensionality of the histogram increases for larger neighbourhood systems the sample space will still tend to be sparsely populated. Decreasing the number of grey levels also erodes the information content of the probability density function. However there is another issue at hand.

For an MRF the joint distribution $\Pi(\mathbf{x})$ must hold the property

$$\Pi(\mathbf{x}) > 0 \qquad \forall \mathbf{x} \in \Omega, \tag{5.6}$$

which implies that for the LCPDF,

$$P(\lambda_s | x_r, r \in \mathcal{N}) > 0 \qquad \forall s \in S, \lambda_s \in \Lambda \tag{5.7}$$

Therefore for a valid LCPDF a positive density needs to be derived from the sparsely populated multi-dimensional histogram. This density should not only conform to Eq. (5.7), but also discern a representative probability density function from the sampled data. In our prior knowledge about this density function we assume that there exists a degree of continuity in the representation, therefore we are expecting a "smooth" distribution. The most common nonparametric density estimator is the Parzen-window density estimator [60] which smoothes the sample data within the multi-dimensional histogram.

5.3 Parzen window density estimator

The Parzen-window density estimator [60] has the effect of smoothing each sample data point in the multi-dimensional histogram over a larger space, possibly in the shape of a multi-dimensional Gaussian surface as indicated in Fig. 5.2.

Given a training image $\mathbf{y} \in \Omega$ of a homogeneous texture and a predefined neighbourhood system $\mathcal{N}$ defined on a lattice S_y, the sample data $\mathbf{Z}_p = \mathrm{Col}[y_p, y_q, q \in \mathcal{N}_p]$ is taken from all sites $p \in S_y$ for which $\mathcal{N}_p \subset S_y$. Denote the variable n as the number of sample data $\mathbf{Z}_p$, *i.e.,* the number of sites $\{p \in S_y, \mathcal{N}_p \subset S_y\}$. Equate $d = |\mathcal{N}_p| + 1$ the number of elements in the vector $\mathbf{Z}_p$, *i.e.,* d equals the dimensionality of the previously defined multi-dimensional histogram. Finally the Parzen-window estimator requires a window parameter h. Thus for a column vector $\mathbf{z} = \mathrm{Col}[L_0, L_{n_r}, r \in \mathcal{N}_s] = \mathrm{Col}[L_0, \ldots, L_{|\mathcal{N}_s|}]$, the Parzen-window density estimated

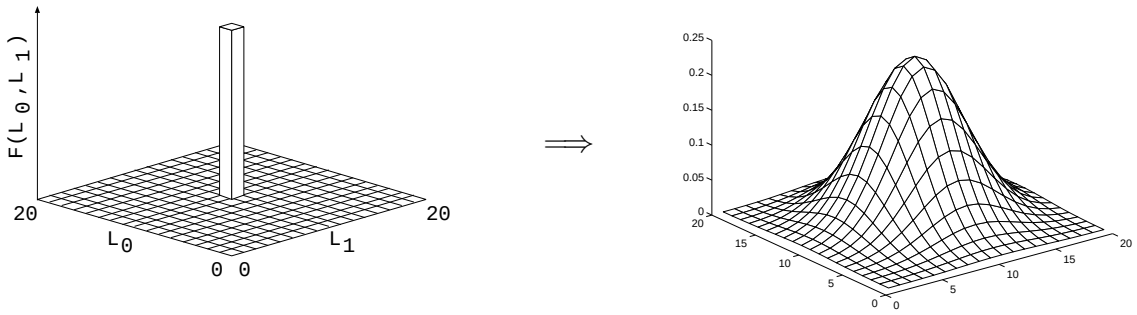


Figure 5.2: The density estimation process involves convolving each histogram data point with multi-dimensional Gaussian.

function $\hat{f}(\mathbf{z})$ of the true density function f is given by [60] as,

$$\hat{f}(\mathbf{z}) = \frac{1}{nh^d} \sum_{\substack{p \in S_y, \\ \mathcal{N}_p \subset S_y}} K\left\{\frac{1}{h}(\mathbf{z} - \mathbf{Z}_p)\right\}, \tag{5.8}$$

The shape of the smoothing is defined by the kernel function K. The kernel function $K(\mathbf{z})$ is defined for d-dimensional $\mathbf{z}$, and must satisfy,

$$\int_{\Re^d} K(\mathbf{z}) d\mathbf{z} = 1. \tag{5.9}$$

Usually K will be a radially symmetric unimodal probability density function (PDF). We chose K as the standard multi-dimensional Gaussian density function,

$$K(\mathbf{z}) = \frac{1}{(2\pi)^{d/2}} \exp(-\frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{z}), \tag{5.10}$$

defined with unit variance.

The size of the kernel function K is defined by the window parameter h. The aim is to correctly choose h so as to obtain the best estimate of the frequency distribution $\hat{f}$ for the LCPDF. If h is too small, random error effects dominate, and a "noisy" estimate of the true density function f results. Furthermore, the LCPDF will not be general enough to represent all subjectively similar textures with the same model. If h is too large, then all the useful information will be lost in a "blur" and detail associated with the texture will be lost. The aim is to correctly choose h so as to "focus" the estimate density function $\hat{f}$ to be as similar as possible to the true density function f.

Given the kernel function defined by Eq. (5.10) and assuming that the unknown density function f has bounded and continuous second derivatives [183], then the optimal window parameter h_{opt} may be determined by minimising the mean integrated square error as shown in [183]. However the equation for this optimal window parameter h_{opt} is defined with respect to the unknown optimal density function f. If it is assumed that this optimal density function f is a standard multi-dimensional Gaussian density, then from [183],

$$h_{opt} = \sigma \left\{ \frac{4}{n(2d+1)} \right\}^{1/(d+4)}, \tag{5.11}$$

where σ^2 is the the average marginal variance. In our case the marginal variance is the same in each dimension of the multi-dimensional histogram, whereby σ^2 equals the variance associated with the one-dimensional histogram of the training image $\mathbf{y}$.

The estimated LCPDF, defined with respect to the Parzen-window density estimated function $\hat{f}$ is then,

$$\begin{aligned} \hat{P}(x_s|x_r, r \in \mathcal{N}_s) &= \frac{\hat{f}(L_0 = x_s, L_{n_r} = x_r, r \in \mathcal{N}_s)}{\sum_{\lambda_s \in \Lambda} \hat{f}(L_0 = \lambda_s, L_{n_r} = x_r, r \in \mathcal{N}_s)} \\ &= \frac{\hat{f}(\mathbf{z} = \mathrm{Col}[x_s, x_r, r \in \mathcal{N}_s])}{\sum_{\lambda_s \in \Lambda} \hat{f}(L_0 = \lambda_s, L_{n_r} = x_r, r \in \mathcal{N}_s)} \\ &= \frac{\sum_{p \in S_y, \mathcal{N}_p \subset S_y} \exp\left[-\frac{1}{2h_{opt}^2} (\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}} (\mathbf{z} - \mathbf{Z}_p) \right]}{\sum_{x_s \in \Lambda} \sum_{p \in S_y, \mathcal{N}_p \subset S_y} \exp\left[-\frac{1}{2h_{opt}^2} (\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}} (\mathbf{z} - \mathbf{Z}_p) \right]}, \end{aligned} \tag{5.12}$$

where $\mathbf{z} = \mathrm{Col}[x_s, x_r, r \in \mathcal{N}_s]$.

In practice it is more convenient to estimate the LCPDF, as required, directly from the sample data rather than building a multi-dimensional histogram and convolving it with a multi-dimensional Gaussian. For large neighbourhood systems or for images with a large range of grey levels, the multi-dimensional histogram tends to be too large to store and too large to calculate every frequency estimate. Instead, as only the LCPDF is required, it is simpler to calculate the LCPDF directly from the sample data via Eq. (5.12), and compactly storing the sample data in the form of the training image.

5.3.1 Required sample size for given accuracy

Suppose that the true density f is unit multivariate normal, and that the kernel is also multivariate normal. Suppose that it is of interest to estimate f at the mean, and that the window parameter h has been chosen to minimise the mean square error at this point. Table 5.1 shows the sample size required to ensure that the relative mean square error at the mean is less than 0.1 [183].

Table 5.1: Sample size required to estimate a standard multivariate normal density at the mean for an error of less than 0.1, from Silverman [183]

Dimensionality	*Required sample size*
1	4
2	19
3	67
4	223
5	768
6	2 790
7	10 700
8	43 700
9	187 000
10	842 000

Furthermore, the results obtained in this case are likely to be optimistic since the point 0 is not in the tail of the distribution, and the normal is a smooth unimodal density. To highlight why the sample size increases so rapidly; given a ten-dimensional normal distribution, 99% of the mass of the distribution is at points whose distance from the origin is greater than 1.6 standard deviation. Compared to the one-dimensional case, where nearly 90% of the distribution lies between ± 1.6 standard deviation. This shows that it is likely to be difficult to estimate the density except from enormous samples, and that the density itself may give a superficially false impression of the likely behaviour of sample data sets. Similar behaviour is observed even when the tail of the density is eliminated altogether by approximating the density as a uniform distribution over a multidimensional box [180].

5.4 Alternative nonparametric estimators

5.4.1 Adaptive kernel estimator

One practical drawback of the kernel method for density estimation is its inability to deal satisfactorily with the tails of distributions without over smoothing the main part of the density. The adaptive kernel estimator overcomes the problem by letting the window parameter h vary depending on the density of the sample data. Broader kernels are used in regions of low density. Thus an observation in the tail would have its mass "smudged" out over a wider range than one in the main part of the distribution.

The first stage of the method is to identify whether or not an observation is in a region of low density. An initial estimate is used to get a rough idea of the density; this estimate yields a pattern of window parameters h corresponding to various observations and these window parameters h are used to construct the adaptive estimator itself [183], Fig. 5.3.

5.4.2 Adaptive bin estimator

The adaptive bin estimator works on the same principal as the adaptive kernel estimator from Section 5.4.1. Both are designed to improve the estimate in the tails of the distributions. While the adaptive kernel estimator modifies the window parameter h – increasing h for areas where there are low bin counts – the adaptive bin estimator modifies the histogram bins themselves. The adaptive bin estimator is based on the idea that in order to record a statistically significant count in a bin that is located in a low density area, the bin size has to be increased until it captures enough sample data. The following procedure illustrates this idea for a one-dimensional histogram.

Consider a one-dimensional histogram denoted as $\mathbf{H} = \{H_\lambda = F(\lambda), \lambda \in \Lambda\}$, where H_λ is the histogram bin for $\lambda \in \Lambda$ and $F(\lambda)$ is the respective count (frequency). Again $\Lambda = \{0, 1, 2, \ldots, L-1\}$. Given an image $\mathbf{X} = \mathbf{x}$ defined on a set of sites $S = \{s_1, s_2, \ldots, s_N\}$, the histogram is then defined as,

$$\mathbf{H} = \left\{ H_\lambda = \sum_{s \in S} \delta(x_s - \lambda), \lambda \in \Lambda \right\} \tag{5.13}$$

Adaptive Kernel Estimator

1. Find a *pilot estimate* $\tilde{f}(\mathbf{z}),\ \mathbf{z} \in \Lambda^d$ that satisfies $\tilde{f}(\mathbf{Z}_p = \text{Col}[y_p, y_q, q \in \mathcal{N}_p]) > 0\ \forall p \in S_y, \mathcal{N}_p \subset S_y$, and let n equal the total number of data points $\mathbf{Z}_p,\ p \in S_y, \mathcal{N}_p \subset S_y$.

2. Define *local window factors* γ_p by

$$\gamma_p = \{\tilde{f}(\mathbf{Z}_p)/g\}^{-\alpha}$$

where g is the geometric mean of the $\tilde{f}(\mathbf{Z}_p)$ such that,

$$\log g = \frac{1}{n} \sum_{\substack{p \in S_y, \\ \mathcal{N}_p \subset S_y}} \log \tilde{f}(\mathbf{Z}_p)$$

and α is the *sensitivity parameter*, a number satisfying $0 \leq \alpha \leq 1$, but best set to $\frac{1}{2}$.

3. Define the *adaptive kernel estimate* $\hat{f}(\mathbf{z})$ by

$$\hat{f}(\mathbf{z}) = \frac{1}{n} \sum_{\substack{p \in S_y, \\ \mathcal{N}_p \subset S_y}} \frac{1}{(h\gamma_p)^d} K\left\{\frac{\mathbf{z} - \mathbf{Z}_p}{h\gamma_p}\right\}$$

where K is the kernel function and h is the window parameter. As in the ordinary kernel method, K is a symmetric function integrating to unity

Figure 5.3: Adaptive Kernel Estimator

where δ is the Kronecker function. Note that $N = \sum_{\lambda \in \Lambda} F(\lambda)$.

Before we can combine histogram bins together we need to know the expected mean and variance of each bin count. Two bins should only be combined if both their counts adequately fall within the variance of the mean of their counts.

The binomial distribution gives the probability of obtaining a specified number of successes when sampling from a finite population. Since $F(\lambda)$ represents the number of successes for $x_s = \lambda$ from a finite population S, the probability $P(H_\lambda = F(\lambda)),\ \lambda \in \Lambda$ can be calculate using the binomial distribution function if the true

probability distribution $p(\lambda),\ \forall\lambda \in \Lambda$ is known,

$$P(H_\lambda = F(\lambda); p(\lambda), N) = \begin{pmatrix} N \\ F(\lambda) \end{pmatrix} p(\lambda)^{F(\lambda)}(1 - p(\lambda))^{N-F(\lambda)}, \qquad \lambda \in \Lambda \quad (5.14)$$

where,

$$\begin{pmatrix} N \\ F(\lambda) \end{pmatrix} = \frac{N!}{F(\lambda)!(n - F(\lambda))!} \quad (5.15)$$

We are interested in the reverse situation. From the given counts $F(\lambda),\ \lambda \in \Lambda$ we wish to determine the expected value of $p(\lambda)$ and the confidence bounds on the estimate. The expected value is quite simply defined as,

$$\hat{p}(\lambda) = \frac{F(\lambda)}{N} \quad (5.16)$$

The confidence bounds on $p(\lambda)$ may be found by approximating the binomial to the normal distribution whereby $\mu = p(\lambda)$ and $\sigma^2 = p(\lambda)(1 - p(\lambda))$. This approximation gives reasonable results if $Np(\lambda)$ and $Np(\lambda)(1 - p(\lambda))$ are both at least 5. Alternatively, we may use the curves from Hahn and Shapiro [94] for 95 and 99 percent confidence. Neither are quite acceptable, as it is the low density areas of the histogram that we are mostly interested in, where either $Np(\lambda)$ or $Np(\lambda)(1 - p(\lambda))$ is less than 5. Also the sample sets are usually much greater than 1000, which is the limit of their curves. An alternative approach, based on the idea that $p(\lambda)$ can be bounded, is described in Fig. 5.4.

We now have a method for calculating the adaptive bin sizes in one-dimension, now a complementary method for multi-dimensional histograms is required. Unfortunately for a multi-dimensional histogram it is not obvious how the histogram bins should be ordered from the smallest deviation to largest, as in the one-dimensional case, Fig. 5.4 Step 3.

One method would be not to do the adaptive bin estimation in multi-dimensions, but to do the bin estimation on each of the one-dimensional marginal histograms. The multi-dimensional adaptive bins can then be formed by the "splitting" of the histogram space with respect to each of the one-dimensional adaptive bin estimates. Using the new bin sizes, a more efficient multi-dimensional histogram can be produced. However as the dimensions increase there will be a rapid increase of bins that are under filled, therefore the problem is not solved.

Adaptive Bin Estimator

1. Define individual sets $S(\lambda) = \lambda,\ \forall\lambda \in \Lambda$.

2. Define $p(\lambda) = F(\lambda)/N,\ \forall\lambda \in \Lambda$.

3. Define an index $\lambda(i),\ \lambda, i \in \Lambda$ such that,

$$|F(\lambda(i)) - F(\lambda(i)-1)| \leq |F(\lambda(i+1)) - F(\lambda(i+1)-1)|,\ \forall 0 < i < L-1$$

so λ is ordered with respect to i from smallest deviation in $F(\lambda)$ to largest.

4. Choose a minimum probability $P_{\min}$ such that,

$$P_{\min} \leq \min_{\lambda\in\Lambda}\ \{P(H_\lambda = F(\lambda); p(\lambda), n)\}\,.$$

5. **For** $i = 1$ to $L-1$ **do**

5.1. Let $S = S(\lambda(i)) \cup S(\lambda(i)-1)$.

5.2. Let $p = \underset{j\in S}{\text{avg}}\ \{p(\lambda(j))\}$.

5.3. **If** $P(H_{\lambda(j)} = F(\lambda(j)); p, n) \geq P_{\min},\ \forall j \in S$ **Then**
bins $H_{\lambda(j)},\ \forall j \in S$ can be acceptably represented as a combined bin with an associated probability equal to p, such that,

$$\begin{aligned} S(\lambda(j)) &= S \qquad \forall j \in S \\ p(\lambda(j)) &= p \qquad \forall j \in S \end{aligned}$$

6. The new histogram is then defined as, $\mathbf{H}' = \{H'_\lambda = \{H_i,\ \forall i \in S(\lambda)\}\}$.

Figure 5.4: Adaptive Bin Estimator

A second method is not to use any spatial ordering of the bins, but to order the bins with respect to their individual counts $F(\mathbf{z}),\ \mathbf{z} \in \Lambda^d$, where d is the dimensionality of the histogram. First all the bins with the same counts need to be grouped together, such that Fig. 5.4 Step 1 is replaced by,

$$S(\mathbf{z}) = \left\{i : F(i) = F(\mathbf{z}),\ i \in \Lambda^d\right\}, \qquad \forall \mathbf{z} \in \Lambda^d. \tag{5.17}$$

Then the ordering of the bins need only be compiled for the set of unique counts $\{F(\mathbf{z})\}$ such that no two counts in the set $\{F(\mathbf{z})\}$ are the same. However in high multi-dimensional histograms, what one usually finds is that the only unique counts are $F(\mathbf{z}) = 0$ and $F(\mathbf{z}) = 1$. This makes adaptive bin estimator unworkable for high

multi-dimensional histograms, and therefore it is probably easier to abide with the adaptive kernel estimator.

5.5 Multi-dimensional histogram compression

For $\Lambda = \{0, 1, \ldots, L-1\}$ the domain of the d-dimensional histogram is equal to L^d which, for even moderately sized d, is far too large to store in computer memory. Even for $L = 4$ the space required becomes prohibitively large for $d > 9$. On the other hand, if we were to store only those histogram bins that have counts greater than zero, and regenerate the LCPDF as required from these counts, then the space required to store the histogram would be reduced. But if the average bin counts were less than d, then the memory space required to store the histogram would be larger than that required to store the image itself.

The algorithms we used to analyse texture were designed to run on the massively parallel processor DECmpp 12000 Sx (MasPar ®). The MasPar provided the necessary decrease in run time, but it had a limited amount of available memory. Therefore the algorithms were designed to use the least amount of memory space. For this reason we chose not to store the d-dimensional histogram, but to calculate the LCPDF as required directly from the image itself. However at any one time the LCPDF only needs to be calculated in one dimension as the values on which it is conditional are fixed. This means the LCPDF requires less memory space than a d-dimensional histogram.

Alternatively a method of histogram compression may be used whereby the density associated with the histogram is approximated by a small set of densities. This is accomplished by clustering the histogram data (see Section 5.5.1) and then constructing a unimodal density over each cluster (see Section 5.5.2), possibly in the form of a standard multi-variate Gaussian. Popat and Picard [164] used precisely this method with great success to produce a non-parametric causal model for texture synthesis. Their method was tried, but found to be too computationally and memory expensive for the needs of the MasPar. Popat and Picard [164] also found that although they could synthesise textures with neighbourhood systems of the order $o = 8$, the method did not extrapolate well to higher order neighbourhoods. On the other hand, the model presented in this thesis has been used to synthesise textures using neighbourhood systems of up to an order of $o = 32$ with gratifying results (see Chapter 7). In fact with our model, we believe that there is no limit to

the size of the neighbourhood that may be used for texture synthesis.

The clustering algorithm employed by Popat and Picard [164] was the vector quantisation algorithm LBG [133]. The LBG clustering algorithm comes from a family of algorithms that have made refinements on the standard ISODATA clustering algorithm [60].

Basic ISODATA clustering algorithm

Step 1 Assume data comprises of c clusters.

Step 2 Choose some initial values for the means $\hat{\mu}_1, \ldots, \hat{\mu}_c$.

Step 3 Classify the n samples by assigning them to the class of the closest mean.

Step 4 Recompute the means as the average of the samples in their class.

Step 5 If any mean has changed value, go to **Step 3**; otherwise stop.

The ISODATA clustering algorithm and its compatriots (*i.e.,* LBG algorithm [133]) rely on an initial guess of the means $\hat{\mu}_1, \ldots, \hat{\mu}_c$. Generally these means are randomly scattered over the sample domain. However this may be inappropriate for sample data in large sample domains like our multi-dimensional histogram. The problem is that if the sample data is not indicative of randomly scattered Gaussian distributions, then it is unlikely that the sample data will be evenly distributed among the means. If the distribution is too skewed the resulting density estimate may not be optimal with respect to the number of means. To better place the means amongst the sample data, unsupervised clustering would probably be more appropriate. The advantage of unsupervised clustering is that it uses the sample data to form the means instead of trying to conform means to the sample data.

5.5.1 Unsupervised clustering

The aim of cluster analysis is to divide a given population into a number of clusters or classes. The aim of unsupervised cluster analysis is to do this without the aid of prior information concerning the properties — or even the existence — of the various classes; the number of classes or the rules of assignment into these classes. All these properties have to be discerned solely from the given data, without reference to a training set. The following nonparametric unsupervised clustering methods have been taken from Silverman [183].

Hierarchical clustering

The hierarchical clustering algorithm was presented by Koontz, Narendra, and Fukunaga [124]. The basis of the algorithm is to assign a parent to each data point in the histogram space. If no parent can be assigned, then that point becomes a root. Each root is given a different class, and those points with parents take on the class of their parent. The algorithm is called hierarchical because no point can have more than one parent, but a parent can have many children.

Density estimates can be used to define a hierarchical structure on a set of points $\{\mathbf{Z}_p,\ p \in S_y, \mathcal{N}_p \subset S_y\}$ in d-dimensional space. Any density estimate $\hat{f}$ can be used [183], but Koontz, Narendra, and Fukunaga [124] used the following. Let d_{ij} be the Euclidean distance between $\mathbf{Z}_i$ and $\mathbf{Z}_j$. Define a neighbourhood $\eta_i(\omega)$ of X_i as

$$\eta_i(\omega) = \{j : d_{ij} \leq \omega, j \neq i\} \tag{5.18}$$

where ω is the threshold. Then the density estimate $\hat{f}$ may be defined as,

$$\hat{f}(\mathbf{Z}_i) = |\eta_i(\omega)| \tag{5.19}$$

Of course this density estimate is not normalised, but it does not need to be for this application.

For data points within distance ω of $\mathbf{Z}_i$, the *parent* of $\mathbf{Z}_i$ is chosen as that data point $\mathbf{Z}_j$ which is steepest uphill from $\mathbf{Z}_i$ with respect to the density estimate $\hat{f}$. In other words, the parent of $\mathbf{Z}_i$ is chosen as,

$$\text{parent}_i = \arg \max_{j; d_{ij} \leq \omega} \hat{f}(\mathbf{X}_j) \geq \hat{f}(\mathbf{X}_i) \left\{ \frac{\hat{f}(\mathbf{X}_j) - \hat{f}(\mathbf{X}_i)}{d_{ij}} \right\} \tag{5.20}$$

For cases where there is more than one suitable parent, a tie breaking rule is applied so only one parent exists. Nodes that have not been assigned a parent are labelled as roots. These roots form the nucleus of the new clusters. If there are a large number of data points $\mathbf{Z}_i$, then the effect of moving up the hierarchical structure is to "hill-climb" on the density estimate $\hat{f}$ towards a local maximum. Thus the divisions between clusters will tend to occur along the "valleys" in the density estimate.

A liability of the hierarchical clustering algorithm is that the number of clusters formed is largely dependent on the chosen threshold ω. There is an intuitive link between the threshold ω and the window parameter h used to determine the size

of the kernel in a nonparametric density estimator [183]. The larger the kernel, the smoother the estimated density. In comparison, as the threshold is increased there is more likelihood that the number of clusters will decrease. An algorithm that has the ability to make the clustering more dependent on the sample data rather than the threshold ω is the mean–shift clustering algorithm.

Mean–shift clustering

The mean–shift clustering algorithm in itself is still very much dependent on the threshold ω. However the basis of the mean–shift clustering algorithm is to iteratively *move* the sample data into clusters. Therefore the algorithm lends itself more readily to progressive clustering, by which after each full application of the algorithm the threshold ω may be increased and the algorithm applied again. Wilson and Spann [207] used precisely this method to decrease the influence of ω on the formation of the clusters. They suggested that by adopting such an approach, a stopping criteria could be used that was alternatively dependent on the movement of the clusters.

The mean–shift clustering algorithm is a special case of Fukunaga and Hostetler's [74] *gradient clustering algorithm*. In their approach, the sample data is iteratively moved up the gradient to become concentrated into a number of tight clumps. Denote $\mathbf{Z}_i^m$ as the position of object i at stage m of the procedure; initially $\mathbf{Z}_i^0 = \mathbf{Z}_i$. A density estimate $\hat{f}_{(m)}$ is constructed from the $\mathbf{Z}_i^m$, and each point is moved in an uphill direction an amount proportional to the gradient of $\log \hat{f}_{(m)}$ at that point. The constant of proportionality is a control parameter α.

$$\begin{aligned} \mathbf{Z}_i^{m+1} &= \mathbf{Z}_i^m + \alpha \nabla \log \hat{f}_{(m)}(\mathbf{Z}_i^m) \\ &= \mathbf{Z}_i^m + \frac{\alpha \nabla \hat{f}_{(m)}(\mathbf{Z}_i^m)}{\hat{f}_{(m)}(\mathbf{Z}_i^m)} \end{aligned} \tag{5.21}$$

The quotient in Eq. (5.21) has several desirable properties. The incremental step of $\mathbf{Z}_i^m$ is dependent on the local density. Small densities induce large steps, whilst large densities induce small steps. Thereby the objects $\mathbf{Z}_i^m$ move quickly towards high density regions. The step size is also governed by the control parameter α. It is advisable to choose α conservatively; if α is chosen too large then objects are likely to 'overshoot' clusters and the algorithm may not converge.

A special case of the gradient clustering algorithm arises when the density esti-

mate $\hat{f}_{(m)}$ is constructed from the kernel method using the multivariate Epanechnikov kernel [183],

$$K_e(\mathbf{z}) = \begin{cases} \frac{1}{2c_d}(d+2)(1-\mathbf{z}^{\mathrm{T}}\mathbf{z}) & \text{if } \mathbf{z}^{\mathrm{T}}\mathbf{z} < 1 \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

where c_d is the volume of the unit d-dimensional sphere: $c_1 = 2, c_2 = \pi, c_3 = 4\pi/3$, *etc.* . Now suppose that the control parameter α is set to the value

$$\alpha = \frac{\omega^2}{d+2} \tag{5.23}$$

Then Eq. (5.21) reduces to the form

$$\begin{aligned} \mathbf{Z}_i^{m+1} &= \text{mean position of all points } \mathbf{Z}_j^m \text{ lying within Euclidean} \\ &\quad \text{distance } \omega \text{ of } \mathbf{Z}_i^m \end{aligned} \tag{5.24}$$

The algorithm given by Eq. (5.24) is called the *mean–shift algorithm.*

As mentioned at the beginning of this subsection, Wilson and Spann [207] used the mean–shift algorithm to reduce the influence of the threshold ω on the formation of the clusters. They iteratively performed the mean–shift algorithm on a set of data, increasing the threshold ω after each implementation of the algorithm. For a stopping mechanism, they waited until after one such implementation of the algorithm where the means did not move. However for unsupervised segmentation purposes we found that their method was still inadequate [157]. Consider the case where we have two independent clusters close to each other in feature space (each represent their own class), also included in this feature space are outlying points that are going to cluster. As the algorithm stands, the stopping mechanism is dependent on all points in the feature space whether they are forming significant clusters or not. Therefore it is quite possible that the two independent clusters will be merged into one just so as to allow the other points to cluster; useful information will have been lost for the sake of outlying points. To overcome just such a problem we proposed a *hierarchical clustering algorithm with significance* [157].

5.5.2 Basis functions

There are various means for compressing the data in the multi-dimensional histogram. All involve trying to fit a particular model to the data. Some common types of models are;

- Radial Basis Functions
- Sum of Gaussian distributions
- Multi-dimensional Boxes

The Radial Basis Functions usually requires a neural network to optimally size and place a number of the functions in the data space (histogram) [9, 145, 163, 64]. Sum of Gaussian distributions is more statistically sound, and can be calculated very easily provided clusters have been identified within the data space [183]. The third method using multi-dimensional boxes, also requires the clusters to be first identified. It models these clusters as multi-dimensional boxes centred around each cluster. The probability density function formed by this model is uniform within each box and equal to the number of data points contained with in the box divided by the total number of data points N. Even though the box estimate is generally not as sound as an estimate by normal density functions, when the quantity of data is very low compared to the histogram space, there will be an equivalent amount of error associated with either estimate.

The important difference between the Gaussian model and the box model, is that the Gaussian model can easily produce a positive definite PDF over the whole histogram space. The box model could also, but it would require defining a default box for the remainder of the data space and assigning it a probability significantly lower than the smallest probability from any other box.

5.5.3 Adaptation for histogram compression

Histogram compression methods described in the previous subsection require clusters to be found that can be described from a common mean. The description can be in the form of a Gaussian function or a box, but both have to be centred on the mean of the cluster and encapsulate all of its data points. The clusters formed DO NOT have to give the best segmentation for the data, but do have to allow the kernel

estimator to derive a respectable distribution $\hat{f}$. Therefore the chosen clustering algorithm for histogram compression was the *mean–shift algorithm*, and the chosen kernel estimator was the Gaussian kernel estimator.

Recalling Section 5.2, an example of a multi-dimensional histogram was presented for a second order neighbourhood system $o = 2$, with the eight nearest neighbours of a pixel. As sample data, a 1000×1000 pixel image was used with a grey level range of 0–15. This gave a histogram space of $16^9 \approx 6.9 \times 10^{10}$ pixels. However, with a million sample points only about $1.5 \times 10^{-3}\%$ of the histogram space would be filled. To extrapolate a useful distribution from this sparse data set, density estimation is required. However even with a density estimate of the histogram space, an LCPDF which represents a vector through this space (as the neighbours are constant) will most likely approximate a uniform distribution due to the low density of the sample data.

At this juncture it might be considered that estimating the LCPDF from a nonparametric density estimate of a multi-dimensional histogram may not be all that useful. Some alternative approaches are listed below.

1. Use the adaptive kernel estimator to create the effect of annealing when using the LCPDF. This is done be defining the kernel of the estimator as a multi-dimensional Gaussian function with an increased variance. Then as the LCPDF is iteratively used, the variance is slowly reduced.

2. Instead of finding the grey level LCPDF of a pixel given its neighbours, find the most likely grey level through some nearest neighbour point method. This equates to something like the ICM method.

3. Produce a pseudo LCPDF as a function of distance to the sample data points.

Alternatively we might like to consider the application of the Gibbs-Markov equivalence theorem. The significance of the Gibbs-Markov equivalence is that the LCPDF $P(x_s|x_r, r \in \mathcal{N}_s)$ does not need to be contrived from a large multi-dimensional histogram. Instead the LCPDF can be reduced to the form,

$$P(x_s|x_r, r \in \mathcal{N}_s) = \frac{\exp\left\{\sum_{C \in \mathcal{C}_s} V_C(\mathbf{x})\right\}}{\sum_{\lambda_s \in \Lambda} \exp\left\{\sum_{C \in \mathcal{C}_s} V_C(\lambda_s, \mathbf{x}_{(s)})\right\}} \tag{5.25}$$

The potential functions $V_C(\mathbf{x})$ are defined over cliques $C \in \mathcal{C}$ which are subsets of the neighbourhood set $\mathcal{N}$. The aim would be to determine a way of deriving the

LCPDF, not from the multi-dimensional histogram of the neighbourhood, but from lower dimensional histograms that represent the cliques of the neighbourhood.

5.6 Goodness-of-fit testing

A method for validating the nonparametric LCPDF estimate is via a goodness-of-fit test [169], *e.g.,* Pearson's X^2 test. However, there are a few problems in obtaining a valid result. One reason for this is due to the high-dimensionality of the multi-dimensional histogram, most frequency counts will be less than the recommended 5 counts [67, 169]. Also the high-dimensionality means that Pearson's X^2 statistic tends to be no longer χ^2 distributed but more normally distributed with mean and variance as specified in [169].

The main question that needs to be asked when determining the goodness-of-fit of the nonparametric LCPDF is whether the neighbourhood size of the LCPDF is correct. We can never be sure that increasing the neighbourhood size will uncover a lower entropy distribution. Consider, for example, a parity bit added to a random bit stream. The PDF landscape will be 1/N featureless (high entropy) until the neighbourhood is large enough to include the parity bit. At that step the PDF becomes highly featured 0 or 2/N.

Presumably, in theory, we could expand the neighbourhood size incrementally and test to see if the actual counts in the histogram bins are significantly different than the high entropy expectations. However, the significance test would rapidly run out of power as number of samples got lost in the high-dimensional space. Thus this test will terminate at some large neighbourhood size.

Dubes and Jain [59] suggest goodness-of-fit testing is largely ignored in the MRF literature due to the computational difficulty in obtaining the test statistic and identifying the significance of it. In addition the relevance of the goodness-of-fit test may be of little consequence in identifying whether the LCPDF has correctly modelled the texture.

Although there is no optimal solution, if the required goodness-of-fit test is to determine whether the LCPDF has correctly modelled a texture by capturing all of its unique characteristics, the "chi-by-eye" test may be the only practical alternative. In which case, the goodness-of-fit test is accomplished by synthesising some textures via the LCPDF and subjectively analysing the visual similarity of the syn-

thetic textures to the training texture. This may indeed be acceptable for many applications of texture analysis. Correct neighbourhood size could then be determined by reducing the neighbourhood size to the minimum required for 'adequate' reproduction.

Chapter 6

Strong Nonparametric MRF Model

In this chapter we incorporate the theory from the MRF-Gibbs distribution equivalence, from Chapter 3, into our own theory that demonstrates an elegant equivalence between the strong MRF model and the ANOVA log-linear construction. From this proof we are able to derive the general ANOVA log-linear construction formula. To our knowledge, this relationship has not been demonstrated before.

6.1 Introduction

The underlying problem with determining the LCPDF is that the domain over which the estimation process is performed is very large and only sparsely populated with sample data. This makes reasonable estimation of the LCPDF very hard to achieve. The model presented in this chapter has been constructed so as to reduce the domain over which the estimation process is performed, while maintaining the integrity of the LCPDF.

A model that is suitable for synthesising texture is not necessarily appropriate for segmenting and classifying texture. Although the nonparametric MRF model can capture all the unique characteristics of the texture it can also be easily overtrained. If the neighbourhood size is increased unnecessarily, then there is a big danger of the LCPDF being overtrained. One obvious limit to this is if the neighbourhood size is increased to the size of the training image. In this case, if the image lattice was not toroidal, we would just have the one sample point (representing the whole image)

by which to estimate the extremely multi-dimensional LCPDF. Consequently, any estimate of the LCPDF would not be representative of other like texture, and therefore the LCPDF could not be used to model that texture class. The other limit in obtaining the correct neighbourhood size is if the neighbourhood size was reduced to just one pixel. In this case we would obtain a very robust estimate of the grey level histogram, but we would not have any spatial information. Clearly a compromise needs to be sought between obtaining enough information in the LCPDF to represent the texture, while still allowing the LCPDF to be generalised for other like texture. This can be achieved via judicious selection of the neighbourhood size.

Finding the correct neighbourhood size is not the only method for obtaining a generalised LCPDF of a texture. Using the same philosophy as Zhu, Wu, and Mumford [211], a texture model used for segmentation and classification should maximise its entropy while retaining the unique characteristics of the texture. The principle of this philosophy is that the texture model should only model known characteristics of a texture and no more. The model should remain completely noncommittal towards any characteristics that are not part of the observed texture. Zhu, Wu, and Mumford [211] use this philosophy to build their minimax model, which was designed to obtain low entropy for characteristics seen in the texture while maintaining high entropy for the rest, thereby attaining a model that infers little information about unseen characteristics. This minimax entropy philosophy is equivalent to reducing the statistical order of a model while retaining the integrity of the respective synthesised textures.

Here we present our new version of the nonparametric model. It is still estimated over the same neighbourhood as its nonparametric MRF counterpart, but the Parzen-window estimation is performed over a set of smaller domains. We estimate the LCPDF as a function of its marginal distributions which reduces the statistical order of the LCPDF. We are able to do this by assuming that there is unconditional independence between non-neighbouring sites for any subset of S. This is a much stronger assumption than is made for a normal MRF which defines a site as being unconditionally independent upon its non-neighbouring sites given all of the neighbouring sites. The difference between the two models can be seen in their mathematical definitions given by Eqs. (6.1) and (6.2). We show that the strong MRF model is equivalent to the Analysis-of-variance (ANOVA) construction [22, 67]. This equivalence allows us to use the theorems from the ANOVA construction to estimate the LCPDF of the strong MRF model.

MRF condition, Eq. (3.8)

$$\Pi_s(x_s|x_r, r \neq s) = P(x_s|x_r, r \in \mathcal{N}_s), \qquad \forall\, \mathbf{x} \in \Omega, s \in S \quad (6.1)$$

Strong MRF condition

$$\Pi_s(x_s|x_r, r \neq s, r \in A \subseteq S) = P(x_s|x_r, r \in \mathcal{N}_s \cap A),\ \forall\, \mathbf{x} \in \Omega, s \in S \quad (6.2)$$

The strong MRF condition implies that the strong LCPDF, $\Pi_s(x_s|x_r, r \neq s, r \in A \subseteq S)$, is only dependent on those states $\{x_r, r \in \mathcal{N}_s \cap A\}$ regardless of whether or not those states are from all the sites $\{r \in \mathcal{N}_s\}$. This is contrary to the standard MRF, for which when some of the states $\{x_r, r \in \mathcal{N}_s\}$ are not given, the conditional probability $\Pi_s(x_s|\mathbf{x}_{(s)})$ will in general no longer be conditionally on just those sites in the neighbourhood $\mathcal{N}_s$ [148].

It was Moussouris [148] who first proposed that the Markovian system could be simplified by imposing stronger conditions on the LCPDF, but it does not necessarily follow that if an image can be modelled as an MRF that it can also be modelled as a strong MRF. However, a common approach to simplifying complex mathematical problems is to assume a degree of independence even when there is no basis for the assumption. Consequently, we shall also assume an extra degree of unconditional independence so as to simplify the MRF model to a strong MRF model.

6.2 Strong MRF theory

The theory presented in this section is our own theory on the strong MRF. The main claim is Proposition 6.1, which we prove via two separate mathematical constructions. The first proof, Section 6.3, is based on the similar proof by Grimmett [92] and Moussouris [148] for the equivalence theorem of a standard MRF and a Gibbs distribution. The second proof, Section 6.4, is based on the ANOVA construction [22, 67] for testing independence in a distribution. As both mathematical constructions are used to prove Proposition 6.1, they are equivalent in terms of the strong MRF. The beauty of this statement is that ANOVA estimation can be used for the strong MRF and vise versa.

Denote the marginal probability $P(\mathbf{x}_A) = P(x_s, s \in A)$, where $A \subseteq S$. For

$\mathbf{x}_A = \{x_s, s \in A\}$, denote the corresponding *configuration space* as Ω_A such that,

$$\mathbf{x}_A \in \Omega_A, \qquad \text{where} \qquad \Omega_A = \Lambda^{|A|}. \tag{6.3}$$

Then the marginal probability $P(\mathbf{x}_A)$ is defined as,

$$P(\mathbf{x}_A) = \sum_{\mathbf{y}_{S-A} \in \Omega_{S-A}} \Pi(\mathbf{x}_A \mathbf{y}_{S-A}), \qquad A \subseteq S. \tag{6.4}$$

The null probability $P(\mathbf{x}_\emptyset)$, defined by Eq. (6.4), is $P(\mathbf{x}_\emptyset) = 1$. Note, however, $P(\mathbf{x}_\emptyset)$ has no physical meaning. For any $s \notin A \subseteq S$, denote,

$$P(x_s|\mathbf{x}_A) = P(\mathbf{x}_{A+s})/P(\mathbf{x}_A) = P(x_s|x_r, r \in A). \tag{6.5}$$

The notation $A + s$ is used to denote a set of sites A plus the site s, or alternatively $A - s$ denotes the same set A excluding the site s.

The strong MRF condition may be expressed in the form of the following identity. Given two sites $s, t \in S$ for which neither is a neighbour of the other, *i.e.*, $t \notin \mathcal{N}_s \Leftrightarrow s \notin \mathcal{N}_t$, and given $s, t \notin B \subseteq S$, then the strong MRF condition of Eq. (6.2) can be expressed as,

$$\begin{aligned} P(x_s|x_t, x_B) &= P(x_s|x_B) \\ \frac{P(\mathbf{x}_{B+s+t})}{P(\mathbf{x}_{B+t})} &= \frac{P(\mathbf{x}_{B+s})}{P(\mathbf{x}_B)}. \end{aligned} \tag{6.6}$$

Proposition 6.1 *Given a neighbourhood system $\mathcal{N}$, the LCPDF of a strong MRF may be decomposed as,*

$$\log P(x_s|x_r, r \in \mathcal{N}_s) = \sum_{C \in \mathcal{C}_s} \sum_{s \in C' \subseteq C} (-1)^{|C|-|C'|} \log P(x_s|\mathbf{x}_{C'-s}) \tag{6.7}$$

or,

$$\log P(x_s, x_r, r \in \mathcal{N}_s) = \sum_{C \in \mathcal{C}_s} \sum_{s \in C' \subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \tag{6.8}$$

These equations may be further decomposed through Moussouris's [148] decomposition to be expressed as,

$$P(x_s|x_r, r \in \mathcal{N}_s) = \prod_{C \in \mathcal{C}_s} P(x_s|\mathbf{x}_{C-s})^{n_{\mathcal{C}_s C}}, \tag{6.9}$$

and,

$$P(x_s, x_r, r \in \mathcal{N}_s) = \prod_{C \in \mathcal{C}_s} P(\mathbf{x}_C)^{n_{\mathcal{C}_s C}}, \tag{6.10}$$

respectively where,

$$n_{\mathcal{C}_s C} = (-1)^{|C|} \sum_{C \subseteq C' \in \mathcal{C}_s} (-1)^{|C'|}, \tag{6.11}$$

and C, C' are cliques from the local clique set $\mathcal{C}_s = \{C \in \mathcal{C} : s \in C\}$.

6.3 Proof 1 of Proposition 6.1

Here we present our first proof of Proposition 6.1. This proof relies on the Möbius inversion formula Eq. (3.16), and is based on the Grimmett's [92] and Moussouris's [148] equivalence proof for a standard MRF and a Gibbs distribution. However we follow the layout presented by Geman [78]. For any sets $A, B \subseteq S$, we show that for a strong MRF, Π is a Gibbs distribution with respect to the strong $\mathcal{N}$-potential,

$$V_A(\mathbf{x}_A) = \sum_{B \subseteq A} (-1)^{|A|-|B|} \log P(\mathbf{x}_B), \qquad \forall\ \mathbf{x} \in \Omega. \tag{6.12}$$

Moreover, for any element $s \in A$,

$$V_A(\mathbf{x}_A) = \sum_{s \in B \subseteq A} (-1)^{|A|-|B|} \log P(x_s | \mathbf{x}_{B-s}), \qquad \forall\ \mathbf{x} \in \Omega. \tag{6.13}$$

This representation is unique among normalised potentials.

1. Π *is Gibbs w.r.t.* V*:* Assuming Eq. (6.12) and using the Möbius inversion formula Eq. (3.16) for sets $B, C \subseteq S$,

$$\log P(\mathbf{x}_B) = \sum_{C \subseteq B} V_C(\mathbf{x}_C), \tag{6.14}$$

This is the second condition imposed on the two functions P and V. The first condition is implied from Eq. (6.4) for which, given sets $A, B, C \subset S$ such that $B \subseteq A \subseteq S$, we have,

$$\log P(\mathbf{x}_B) = \log \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} P(\mathbf{x}_B \mathbf{y}_{A-B}) \tag{6.15}$$

Then by applying the second condition of Eq. (6.14), we obtain,

$$\begin{aligned}\sum_{C\subseteq B} V_C(\mathbf{x}_C) &= \log \sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{C\subseteq A} V_C(\mathbf{x}_B\mathbf{y}_{A-B}) \\ &= \log \left[\left(\exp \sum_{C\subseteq B} V(\mathbf{x}_C) \right) \left(\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{\substack{C\subseteq A,\\ C\not\subseteq B}} V_C(\mathbf{x}_B\mathbf{y}_{A-B}) \right) \right] \\ &= \sum_{C\subseteq B} V(\mathbf{x}_C) + \log \sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{\substack{C\subseteq A,\\ C\not\subseteq B}} V_C(\mathbf{x}_B\mathbf{y}_{A-B}) \qquad (6.16)\end{aligned}$$

Therefore, from Eq. (6.16) we arrive at the implicit condition that;

$$\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{\substack{C\subseteq A,\\ C\not\subseteq B}} V_C(\mathbf{x}_B\mathbf{y}_{A-B}) = 1 \qquad \forall\ \mathbf{x}\in\Omega,\ B\subseteq A\subseteq S \qquad (6.17)$$

2. V is normalised: The potential V, defined by Eq. (6.12), is not normalised in the conventional manner. In the original proof for a standard MRF, the potentials were said to be normalised if $V_A(\mathbf{x}) = 0$ if $x_s = 0$ for any $s \in A$. For the potential defined by Eq. (6.12), this is not the case. However,

$$V_\emptyset(\mathbf{x}_\emptyset) = \sum_{B\subseteq\emptyset} (-1)^{|\emptyset|-|B|} \log P(\mathbf{x}_B) = \log P(\mathbf{x}_\emptyset) = \log 1 = 0. \qquad (6.18)$$

Therefore, although the potential V is not normalised in the conventional manner, it is bound by the criteria that $V_\emptyset(\mathbf{x}) = 0$, and condition Eq. (6.17).

3. Eq. (6.12) $\Leftrightarrow$ Eq. (6.13): For any $s \in A$,

$$\begin{aligned}V_A(\mathbf{x}_A) &= \sum_{s\in B\subseteq A} (-1)^{|A|-|B|} \log P(\mathbf{x}_B) + \sum_{s\notin B\subseteq A} (-1)^{|A|-|B|} \log P(\mathbf{x}_B) \\ &= \sum_{s\in B\subseteq A} (-1)^{|A|-|B|} (\log P(\mathbf{x}_B) - \log P(\mathbf{x}_{B-s})) \\ &= \sum_{s\in B\subseteq A} (-1)^{|A|-|B|} \log P(x_s|\mathbf{x}_{B-s}). \qquad (6.19)\end{aligned}$$

4. V is a strong $\mathcal{N}$-potential: Given that $\mathbf{x}$ is defined on a strong MRF with respect to $\mathcal{N}$, then V is a strong $\mathcal{N}$-potential if $V_A(\mathbf{x}_A) = 0\ \forall\ A \notin \mathcal{C}$.

Choose $A \notin \mathcal{C}$, then $\exists s, t \in A$ such that $t \notin \mathcal{N}_s \Leftrightarrow s \notin \mathcal{N}_t$.

$$\begin{aligned} V_A(\mathbf{x}_A) &= \sum_{B \subseteq A} (-1)^{|A|-|B|} \log P(\mathbf{x}_B) \\ &= \sum_{B \subseteq A-s-t} (-1)^{|A|-|B|} \log P(\mathbf{x}_B) + \sum_{B \subseteq A-s-t} (-1)^{|A|-|B+s|} \log P(\mathbf{x}_{B+s}) + \\ &\quad \sum_{B \subseteq A-s-t} (-1)^{|A|-|B+t|} \log P(\mathbf{x}_{B+t}) + \sum_{B \subseteq A-s-t} (-1)^{|A|-|B+s+t|} \log P(\mathbf{x}_{B+s+t}) \\ &= \sum_{B \subseteq A-s-t} (-1)^{|A|-|B|} \log \left[\frac{P(\mathbf{x}_B) P(\mathbf{x}_{B+s+t})}{P(\mathbf{x}_{B+s}) P(\mathbf{x}_{B+t})} \right] \\ &= 0. \end{aligned} \tag{6.20}$$

Note identity Eq. (6.6) for a strong MRF was used. This modifies Eq. (6.14) to,

$$\log P(\mathbf{x}_B) = \sum_{\substack{C \subseteq B, \\ C \in \mathcal{C}}} V_C(\mathbf{x}_C). \tag{6.21}$$

Therefore a strong MRF, with Gibbs distribution Π, may be expressed with respect to the $\mathcal{N}$-potentials Eq. (6.12) or Eq. (6.13).

□

As Eq. (6.12) and Eq. (6.13) have been shown to be $\mathcal{N}$-potentials, we can now use them to prove Proposition 6.1. Consider a site $s \in S$, then the strong LCPDF may be expressed as;

$$\begin{aligned} P(x_s | x_r, r \in \mathcal{N}_s) &= \frac{P(\mathbf{x}_S)}{P(\mathbf{x}_{S-s})} = \exp\left[\sum_{C \in S} V_C(\mathbf{x}_C) - \sum_{C \in S-s} V_C(\mathbf{x}_C) \right] \\ &= \exp\left[\sum_{s \in C \in S} V_C(\mathbf{x}_C) \right] = \exp\left[\sum_{C \in \mathcal{C}_s} V_C(\mathbf{x}_C) \right] \\ \log P(x_s | x_r, r \in \mathcal{N}_s) &= \sum_{C \in \mathcal{C}_s} \sum_{s \in C' \subseteq C} (-1)^{|C|-|C'|} \log P(x_s | \mathbf{x}_{C'-s}) \end{aligned} \tag{6.22}$$

Where in the above equation, the $\mathcal{N}$-potential Eq. (6.13) was used since all the cliques $C \in \mathcal{C}_s$ contain the site s. This proves the first part of Proposition 6.1, Eq. (6.7).

The second part of Proposition 6.1, Eq. (6.8) is proved by applying Möbius

inversion formula Eq. (3.17) to Eq. (6.12) for the set of sites $\mathcal{N}_s + s \subset S$;

$$\begin{aligned}
\log P(x_s, x_r, r \in \mathcal{N}_s) &= \sum_{C \subseteq \mathcal{N}_s + s} \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \\
&= \sum_{C \subseteq \mathcal{N}_s} \left\{ \sum_{C' \subseteq C+s} (-1)^{|C+s|-|C'|} \log P(\mathbf{x}_{C'}) + \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \right\} \\
&= \sum_{C \subseteq \mathcal{N}_s} \left\{ \sum_{C' \subseteq C} (-1)^{|C+s|-|C'+s|} \log P(\mathbf{x}_{C'+s}) + \sum_{C' \subseteq C} (-1)^{|C+s|-|C'|} \log P(\mathbf{x}_{C'}) + \right. \\
&\qquad \left. \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \right\} \\
&= \sum_{C \subseteq \mathcal{N}_s} \sum_{C' \subseteq C} (-1)^{|C+s|-|C'+s|} \log P(\mathbf{x}_{C'+s}) \\
&= \sum_{C \subseteq \mathcal{C}_s} \sum_{s \in C' \subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \qquad (6.23)
\end{aligned}$$

Finally, to obtain Eq. (6.9) and Eq. (6.10) of Proposition 6.1 we may observe that both Eq. (6.22) and Eq. (6.23) have the correct Möbius set decomposition with respect to the set $\mathcal{N}_s$. Even though the site s is included in the decomposition, it is included in all cliques and therefore does not compromise the decomposition over the set $\mathcal{N}_s$. Therefore we can apply the Moussouris [148] conversion that gave Eq. (3.43) $\Leftrightarrow$ Eq. (3.44), to both Eq. (6.9) and Eq. (6.10) over the set of sites $\mathcal{N}_s$ to obtain,

$$P(x_s | x_r, r \in \mathcal{N}_s) = \prod_{C \in \mathcal{C}_s} P(x_s | \mathbf{x}_{C-s})^{n_{\mathcal{C}_s C}}, \qquad (6.24)$$

and,

$$P(x_s, x_r, r \in \mathcal{N}_s) = \prod_{C \in \mathcal{C}_s} P(\mathbf{x}_C)^{n_{\mathcal{C}_s C}}, \qquad (6.25)$$

respectively where,

$$n_{\mathcal{C}_s C} = (-1)^{|C|} \sum_{C \subseteq C' \in \mathcal{C}_s} (-1)^{|C'|}, \qquad (6.26)$$

and C, C' are cliques from the local clique set $\mathcal{C}_s = \{C \in \mathcal{C} : s \in C\}$.

□

6.4 Proof 2 of Proposition 6.1

Here we present our second proof of Proposition 6.1. This proof is based on the ANOVA construction [22, 67] for testing independence in a distribution. In ANOVA-type notation, the probability $P(x_s, x_r, r \in \mathcal{N}_s)$ is decomposed into its marginal distributions. This is expressed in terms of the *general log-linear model* [22]:

$$\log P(x_s, x_r, r \in \mathcal{N}_s) = \log P(\mathbf{x}_A) = \sum_{B \subseteq A} U_B(\mathbf{x}_B). \tag{6.27}$$

where A is explicitly denoted as $A = \mathcal{N}_s + s = \{s, r \in \mathcal{N}_s\}$. Neither Bishop *et al.* [22] nor Fienberg [67] describe the ANOVA log-linear model in the general terms of sets. It has been done here to show the generalisation of the model.

In the ANOVA log-linear model, $U_\emptyset$ is the grand mean of the logarithmic probabilities $\log P(\mathbf{y}_A),\ \mathbf{y}_A \in \Omega_A$:

$$U_\emptyset(\mathbf{x}_\emptyset) = \frac{1}{|\Omega_A|} \sum_{\mathbf{y}_A \in \Omega_A} \log P(\mathbf{y}_A). \tag{6.28}$$

Note that the value of $U_\emptyset \neq V_\emptyset$ from Eq. (6.18).

The ANOVA log-linear model is extrapolated as $\sum_{C \subseteq B} U_C(\mathbf{x}_C)$ being equal to the mean of the logarithmic probabilities $\log P(\mathbf{x}_B \mathbf{y}_{A-B}),\ \mathbf{y}_{A-B} \in \Omega_{A-B}$:

$$\sum_{C \subseteq B} U_C(\mathbf{x}_C) = \frac{1}{|\Omega_{A-B}|} \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \log P(\mathbf{x}_B \mathbf{y}_{A-B}). \tag{6.29}$$

From Eq. (6.29), consider the summation of both sides over $x_s \in \Lambda$ for some $s \in B$, that is:

$$\begin{aligned}
\sum_{x_s \in \Lambda} \sum_{C \subseteq B} U_C(\mathbf{x}_C) \;&=\; \sum_{x_s \in \Lambda} \frac{1}{|\Omega_{A-B}|} \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \log P(\mathbf{x}_B \mathbf{y}_{A-B}) \qquad s \in B \\
\sum_{x_s \in \Lambda} \sum_{C \subseteq B-s} U_{C+s}(\mathbf{x}_{C+s}) + \sum_{x_s \in \Lambda} \sum_{C \subseteq B-s} U_C(\mathbf{x}_C) &= \\
&\frac{1}{|\Omega_{A-B}|} \sum_{\mathbf{y}_{A-B+s} \in \Omega_{A-B+s}} \log P(\mathbf{x}_{B-s} \mathbf{y}_{A-B+s}) \\
\sum_{x_s \in \Lambda} \sum_{C \subseteq B-s} U_{C+s}(\mathbf{x}_{C+s}) + \frac{|\Omega_s|}{|\Omega_{A-B+s}|} \sum_{\mathbf{y}_{A-B+s} \in \Omega_{A-B+s}} &\log P(\mathbf{x}_{B-s} \mathbf{y}_{A-B+s}) =
\end{aligned}$$

$$\frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B+s}\in\Omega_{A-B+s}}\log P(\mathbf{x}_{B-s}\mathbf{y}_{A-B+s})$$

$$\sum_{x_s\in\Lambda}\sum_{C\subseteq B-s}U_{C+s}(\mathbf{x}_{C+s})=0 \qquad \forall\ s\in B \tag{6.30}$$

Since $B\subseteq A$ can be any sized set, by the principle of mathematical induction,

$$\sum_{x_s\in\Lambda}U_B(x_s,\mathbf{x}_{B-s})=0 \qquad \forall\ s\in B. \tag{6.31}$$

This is the general form of the condition that was stated in [22] and [67].

The general log-linear model Eq. (6.27) imposes no restrictions on the probability $P(\mathbf{x}_A)$. However, if the set of sites $s\in A$ are independent such that,

$$P(\mathbf{x}_A)=\prod_{s\in A}P(x_s), \tag{6.32}$$

then the ANOVA construction of the logarithmic probability $\log P(\mathbf{x}_A)$ is given by Eq. (6.27) with $U_B(\mathbf{x}_B)=0,\ \forall\ |B|>1$. This implies that $P(\mathbf{x}_A)$ is not made up of any interacting components between sites. In reference to the strong MRF model, it is like saying $P(\mathbf{x}_A)$ is only constructed from potentials defined on single sites.

Given that $\mathbf{x}$ is defined on a strong MRF with respect to $\mathcal{N}$, then U is a strong $\mathcal{N}$-potential if $U_B(\mathbf{x}_B)=0\ \forall\ B\notin\mathcal{C}$. Lets denote $B\notin\mathcal{C}$. Then $\exists s,t\in B$ such that $t\notin\mathcal{N}_s\Leftrightarrow s\notin\mathcal{N}_t$. From Eq. (6.29),

$$\begin{aligned}
\sum_{C\subseteq B}U_C(\mathbf{x}_C) &= \frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}}\log P(\mathbf{x}_B\mathbf{y}_{A-B})\\
U_B(\mathbf{x}_B) &= \frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}}\log P(\mathbf{x}_B\mathbf{y}_{A-B})-\sum_{C\subset B}U_C(\mathbf{x}_C)\\
&= \frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}}\log P(\mathbf{x}_B\mathbf{y}_{A-B})-\sum_{C\subseteq B-s}U_C(\mathbf{x}_C)-\\
&\quad\sum_{C\subseteq B-t}U_C(\mathbf{x}_C)+\sum_{C\subseteq B-s-t}U_C(\mathbf{x}_C)-\sum_{C\subset B-s-t}U_{C+s+t}(\mathbf{x}_{C+s+t})\\
\sum_{C\subseteq B-s-t}U_{C+s+t}(\mathbf{x}_{C+s+t}) &= \frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}}\log P(\mathbf{x}_B\mathbf{y}_{A-B})-\sum_{C\subseteq B-s}U_C(\mathbf{x}_C)-\\
&\quad\sum_{C\subseteq B-t}U_C(\mathbf{x}_C)+\sum_{C\subseteq B-s-t}U_C(\mathbf{x}_C))
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{|\Omega_{A-B}|}\sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}}\log P(\mathbf{x}_B\mathbf{y}_{A-B}) - \\
&\quad \frac{1}{|\Omega_{A-B+s}|}\sum_{\mathbf{y}_{A-B+s}\in\Omega_{A-B+s}}\log P(\mathbf{x}_{B-s}\mathbf{y}_{A-B+s}) - \\
&\quad \frac{1}{|\Omega_{A-B+t}|}\sum_{\mathbf{y}_{A-B+t}\in\Omega_{A-B+t}}\log P(\mathbf{x}_{B-t}\mathbf{y}_{A-B+t}) + \\
&\quad \frac{1}{|\Omega_{A-B+s+t}|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log P(\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+s+t}) \\
&= \frac{1}{|\Omega_{A-B}|}\frac{1}{|\Omega_{s+t}|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log P(\mathbf{x}_B\mathbf{y}_{A-B}) - \\
&\quad \frac{1}{|\Omega_{A-B+s}|}\frac{1}{|\Omega_t|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log P(\mathbf{x}_{B-s}\mathbf{y}_{A-B+s}) - \\
&\quad \frac{1}{|\Omega_{A-B+t}|}\frac{1}{|\Omega_s|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log P(\mathbf{x}_{B-t}\mathbf{y}_{A-B+t}) + \\
&\quad \frac{1}{|\Omega_{A-B+s+t}|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log P(\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+s+t})
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{|\Omega_{A-B+s+t}|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log\left[\frac{P(\mathbf{x}_B\mathbf{y}_{A-B})P(\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+s+t})}{P(\mathbf{x}_{B-s}\mathbf{y}_{A-B+s})P(\mathbf{x}_{B-t}\mathbf{y}_{A-B+t})}\right] \\
&= \frac{1}{|\Omega_{A-B+s+t}|}\sum_{\mathbf{y}_{A-B+s+t}\in\Omega_{A-B+s+t}}\log\left[\frac{P(x_s|\mathbf{x}_{B-s}\mathbf{y}_{A-B})P(y_s|\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+t})}{P(y_s|\mathbf{x}_{B-s}\mathbf{y}_{A-B})P(x_s|\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+t})}\right] \\
&= 0 \qquad (6.33)
\end{aligned}
$$

From Eq. (6.33), $\sum_{C\subseteq B-s-t}U_{C+s+t}(\mathbf{x}_{C+s+t}) = 0\ \forall\ B \notin \mathcal{C}$ since, from the strong MRF identity Eq. (6.6), $P(x_s|\mathbf{x}_{B-s}\mathbf{y}_{A-B}) = P(x_s|\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+t})$ and, similarly, $P(y_s|\mathbf{x}_{B-s}\mathbf{y}_{A-B}) = P(y_s|\mathbf{x}_{B-s-t}\mathbf{y}_{A-B+t})$.

A supplementary result to Eq. (6.33), is that $U_B(\mathbf{x}_B) = 0\ \forall\ B \notin \mathcal{C}$. This can be proved by the principle of mathematical induction, since B is an arbitrary set contained within S, whereby $\exists s,t \in B$, such that $t \notin \mathcal{N}_s \Leftrightarrow s \notin \mathcal{N}_t$. For $B = \emptyset$ we have from Eq. (6.33), $U_{s+t}(\mathbf{x}_{s+t}) = 0$. Then given $U_B(\mathbf{x}_B) = 0$ for $|B| < n$, *i.e.,* where the number of sites in B is less then n, we have from Eq. (6.33) for $|B| = n$,

$$
0 = \sum_{C\subseteq B-s-t}U_{C+s+t}(\mathbf{x}_{C+s+t}) = U_B(\mathbf{x}_B) + \sum_{C\subset B-s-t}U_{C+s+t}(\mathbf{x}_{C+s+t}) = U_B(\mathbf{x}_B) \quad (6.34)
$$

Therefore by the principle of mathematical induction, $U_B(\mathbf{x}_B) = 0 \; \forall \; B \notin \mathcal{C}$.

For a strong MRF, the ANOVA log-linear model may now be rewritten as,

$$\log P(x_s, x_r, r \in \mathcal{N}_s) = \log P(\mathbf{x}_A) = \sum_{\substack{C \subseteq A, \\ C \in \mathcal{C}}} U_C(\mathbf{x}_C). \tag{6.35}$$

Other identities were used to obtain Eq. (6.33). One was the fact that given $\Omega_{A-B} = \Lambda^{A-B}$ and $\Omega_{s+t} = \Lambda^{s+t}$, then,

$$|\Omega_{A-B}||\Omega_{s+t}| = |\Lambda^{A-B}||\Lambda^{s+t}| = |\Lambda^{A-B}\Lambda^{s+t}| = |\Lambda^{A-B+s+t}| = |\Omega_{A-B+s+t}| \tag{6.36}$$

Another was the fact that $P(\mathbf{x}_B\mathbf{y}_{A-B})$ does not vary for $y_s \in \Lambda, \; s \in B$, giving,

$$\log P(\mathbf{x}_B\mathbf{y}_{A-B}) = \frac{1}{|\Omega_s|} \sum_{\mathbf{y}_s \in \Omega_s} P(\mathbf{x}_B\mathbf{y}_{A-B}) \qquad s \in B \tag{6.37}$$

The ANOVA log-linear model, like the strong MRF model, specifies the probability $P(\mathbf{x}_A)$ as being constructed from functions defined on interacting subsets $C \subseteq A$. Comparing Eq. (6.21) and Eq. (6.35), both the strong MRF model and the ANOVA log-linear model restrict these subsets $C \subseteq A$ to cliques $C \in \mathcal{C}$. However, in the ANOVA log-linear model, the functions $U_C(\mathbf{x}_C)$ are not potentials, but represent successive deviations from the mean. This difference is evident when the Möbius inversion formula, Eq. (3.16), is applied to Eq. (6.29) to obtain,

$$U_C(\mathbf{x}_C) = \sum_{C' \subseteq C} (-1)^{|C|-|C'|} \frac{1}{|\Omega_{A-C'}|} \sum_{\mathbf{y}_{A-C'} \in \Omega_{A-C'}} \log P(\mathbf{x}_{C'}\mathbf{y}_{A-C'}). \tag{6.38}$$

Even though the functions $U_C(\mathbf{x}_C) \neq V_C(\mathbf{x}_C), \; \forall \; C \in \mathcal{C}$, the marginal probabilities $P(\mathbf{x}_B), \; B \subseteq A$ are the same. Therefore we should still be able to derive the general formula, Eq. (6.8), for $P(\mathbf{x}_A)$ in terms of $P(\mathbf{x}_B), \; B \subseteq A$. First we need the general equation for $P(\mathbf{x}_B), \; B \subseteq A$. In reference to Bishop *et al.* [22], marginal $P(\mathbf{x}_B)$ may be expressed as,

$$\begin{aligned} P(\mathbf{x}_B) &= \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} P(\mathbf{x}_B\mathbf{y}_{A-B}) \\ &= \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \exp \sum_{C \subseteq A} U_C(\mathbf{x}_B\mathbf{y}_{A-B}) \end{aligned}$$

$$= \left\{ \exp \sum_{C \subseteq B} U_C(\mathbf{x}_B) \right\} \left\{ \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \exp \sum_{\substack{C \subseteq A, \\ C \not\subseteq B}} U_C(\mathbf{x}_B \mathbf{y}_{A-B}) \right\}$$

$$\log P(\mathbf{x}_B) = \sum_{C \subseteq B} U_C(\mathbf{x}_B) + \log \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \exp \sum_{\substack{C \subseteq A, \\ C \not\subseteq B}} U_C(\mathbf{x}_B \mathbf{y}_{A-B}) \tag{6.39}$$

This gives an equation for any $P(\mathbf{x}_B),\ B \subseteq A$. Note that Eq. (6.27) could not have been used for any set $B \subseteq A$, but Eq. (6.27) can be derived from Eq. (6.39),

$$\begin{aligned} \log P(\mathbf{x}_A) &= \sum_{C \subseteq A} U_C(\mathbf{x}_A) + \log \sum_{\mathbf{y}_{A-A} \in \Omega_{A-A}} \exp \sum_{\substack{C \subseteq A, \\ C \not\subseteq A}} U_C(\mathbf{x}_A \mathbf{y}_{A-A}) \\ &= \sum_{C \subseteq A} U_C(\mathbf{x}_A) + \log \exp 0 \\ &= \sum_{C \subseteq A} U_C(\mathbf{x}_A) \end{aligned} \tag{6.40}$$

Rearranging Eq. (6.39) we obtain,

$$\sum_{C \subseteq B} U_C(\mathbf{x}_B) = \log P(\mathbf{x}_B) - \log \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \exp \sum_{\substack{D \subseteq A, \\ D \not\subseteq B}} U_D(\mathbf{x}_B \mathbf{y}_{A-B}) \tag{6.41}$$

If the right hand side of Eq. (6.41) is regarded as just a function for the set B, then it is clear we may apply the Möbius inversion formula, Eq. (3.16), to Eq. (6.41) and obtain $U_C(\mathbf{x}_C)$ such that,

$$U_C(\mathbf{x}_C) = \sum_{B \subset C} (-1)^{|C|-|B|} \left[\log P(\mathbf{x}_B) - \log \sum_{\mathbf{y}_{A-B} \in \Omega_{A-B}} \exp \sum_{\substack{D \subseteq A, \\ D \not\subseteq B}} U_D(\mathbf{x}_B \mathbf{y}_{A-B}) \right] \tag{6.42}$$

We now have an equation for $U_C(\mathbf{x}_C)$ in terms of the marginal probabilities $P(\mathbf{x}_B),\ B \subseteq A$ which we can substitute into Eq. (6.35), giving,

$$\log P(\mathbf{x}_A) = \sum_{\substack{C \subseteq A, \\ C \in \mathcal{C}}} U_C(\mathbf{x}_C)$$

$$
\begin{aligned}
&= \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \left[\log P(\mathbf{x}_B) - \log \sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{\substack{D\subseteq A,\\ D\not\subseteq B}} U_D(\mathbf{x}_B\mathbf{y}_{A-B}) \right] \\
&= \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \log P(\mathbf{x}_B) - \\
&\quad \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \log \sum_{\mathbf{y}_{A-B}\in\Omega_{A-B}} \exp \sum_{\substack{D\subseteq A,\\ D\not\subseteq B}} U_D(\mathbf{x}_B\mathbf{y}_{A-B}) \\
&= \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \log P(\mathbf{x}_B) - \log \sum_{\mathbf{y}_{A-A}\in\Omega_{A-A}} \exp \sum_{\substack{D\subseteq A,\\ D\not\subseteq A}} U_D(\mathbf{x}_A\mathbf{y}_{A-A}) \\
&= \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \log P(\mathbf{x}_B) - \log \exp 0 \\
&= \sum_{\substack{C\subseteq A,\\ C\in\mathcal{C}}} \sum_{B\subset C} (-1)^{|C|-|B|} \log P(\mathbf{x}_B) \qquad (6.43)
\end{aligned}
$$

Again the Möbius inversion formula, Eq. (3.16), was applied. This time it was used to simplify the sum for $U_D(\mathbf{x}_B\mathbf{y}_{A-B})$. Although the summation was over the set A, the correct Möbius set decomposition did occur for the sites $A-s$. Even though the site s is included in the decomposition, it is included in all the sets and therefore does not compromise the decomposition.

From Eq. (6.43) we have,

$$\log P(x_s, x_r, r\in\mathcal{N}_s) = \sum_{\substack{C\subseteq\mathcal{N}_s+s,\\ C\in\mathcal{C}}} \sum_{C'\subset C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \qquad (6.44)$$

As in the derivation for Eq. (6.23), Eq. (6.44) can be re-expressed as,

$$\log P(x_s, x_r, r\in\mathcal{N}_s) = \sum_{C\subseteq\mathcal{C}_s} \sum_{s\in C'\subseteq C} (-1)^{|C|-|C'|} \log P(\mathbf{x}_{C'}) \qquad (6.45)$$

Therefore via the ANOVA log-linear model, Eq. (6.8) of Proposition 6.1 is proved. The rest of Proposition 6.1 is subsequently proved, since the first proof of Proposition 6.1 showed a connection between the rest of the equations.

□

As a side note, if $\mathbf{x}$ is not defined on a strong MRF, then this means that

$U_B(\mathbf{x}_B) \neq 0 \; \forall \; B \notin \mathcal{C}$. The outcome of this is that instead of obtaining Eq. (6.45), we obtain the general formula for the ANOVA construction,

$$\log P(\mathbf{x}_A) = \sum_{B \subseteq A} \sum_{C \subseteq B} (-1)^{|B|-|C|} \log P(\mathbf{x}_C) \tag{6.46}$$

or equivalently, through Moussouris's [148] decomposition,

$$P(\mathbf{x}_A) = \prod_{C \subseteq A} P(\mathbf{x}_C)^{n_{AC}}, \qquad n_{AC} = (-1)^{|C|} \sum_{C \subseteq B \subseteq A} (-1)^{|B|} \tag{6.47}$$

Although this formula was proposed by Moussouris for the strong MRF, the formula can also be applied in the study of the ANOVA for contingency tables. It is not known whether this has been made apparent to the contingency tables community.

6.5 Equivalence with ANOVA model

We presented two proofs of Proposition 6.1. The first method relied on the Möbius inversion formula Eq. (3.16), and followed Grimmett's [92] and Moussouris's [148] construction for the $\mathcal{N}$-potential V, Section 6.3. The second proof was based on the ANOVA log-linear model [22, 67] for testing variable independence in a distribution, Section 6.4. Our proofs showed that the strong MRF model is equivalent to the ANOVA construction.

Even though Eq. (6.10) from Proposition 6.1 represents the general clique decomposition formula for $P(x_s, x_r, r \in \mathcal{N}_s)$, it is subject to condition Eq. (6.17). Bishop *et al.* [22] did not derive the general formulas Eq. (6.10) or Eq. (6.47) for the ANOVA construction, but did suggest under what conditions they exist. Given a set of cliques over which Eq. (6.10) is calculated, Bishop *et al.* [22] outlined steps for determining when Eq. (6.10) is valid.

Step 1 If for any $s \in A$ such that $s \in B$, $\forall \; P(\mathbf{x}_B)$, then relabel all marginal probabilities $P(\mathbf{x}_{B-s})$.

Step 2 If for any $s \in A$ such that $s \in B$, $\forall \; P(\mathbf{x}_B)$ except for $P(\mathbf{x}_C)$, then relabel $P(\mathbf{x}_C)$ to $P(\mathbf{x}_{C-s})$.

Step 3 If for any marginal probability $P(\mathbf{x}_B)$ such that $B \cap C = \emptyset$, $\forall \; C \subset A, C \neq B$, then remove $P(\mathbf{x}_B)$ from the set of marginal probabilities.

Step 4 Repeat Steps 1 – 3 until

1. No more than two marginal probabilities remain. This means that a closed-form estimate of $P(\mathbf{x}_A)$ exists, *i.e.,* Eq. (6.10) is valid.
2. No further steps can be taken. This is an indication that no closed-form estimate of $P(\mathbf{x}_A)$ exists.

Basically the closed-form estimate of $P(\mathbf{x}_A)$ exists, *i.e.,* Eq. (6.10) is valid, if the sets $B \subseteq A$ of the marginal probabilities $P(\mathbf{x}_B)$ do not form a loop of three or more sets.

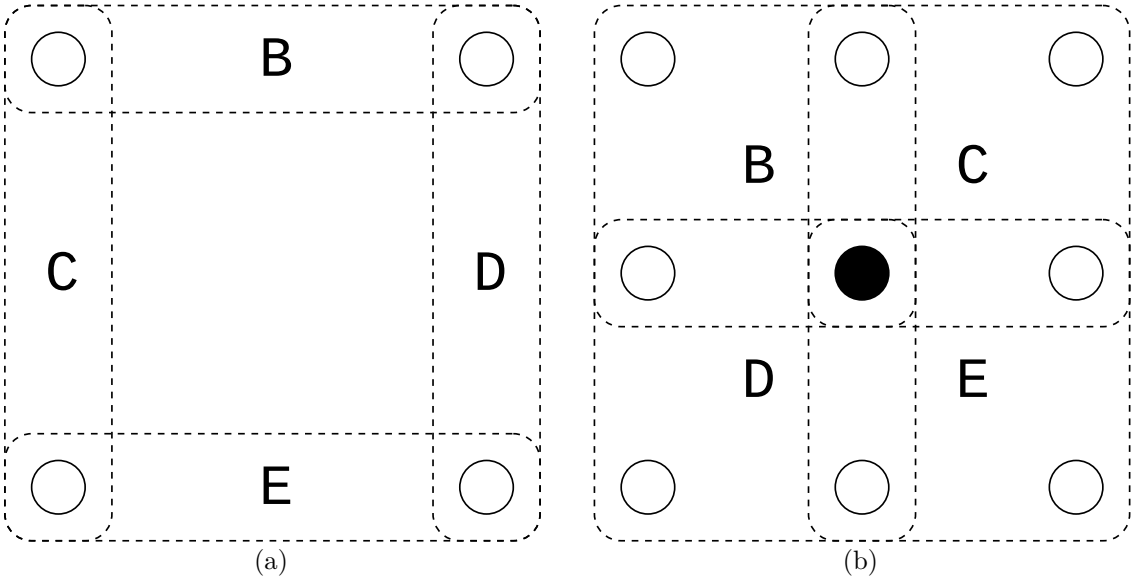


Figure 6.1: Examples of three or more sets that form a loop. These types of sets can not be represented by the strong MRF formula.

In Fig. 6.1 shows two ways in which a loop of three or more sets may be formed. Fig. 6.1 (a) shows a basic four site set A with pairwise subsets B, C, D, E. This configuration does not reduce further via the steps outlined above. Therefore a closed-form estimate of $P(\mathbf{x}_A)$ does not exist. In the case of the strong MRF model, a more common scenario is the construction of the LCPDF from the marginal probabilities $P(\mathbf{x}_C)$ from the set of cliques $C \in \mathcal{C}_s$. Fig. 6.1 (b) shows an example of the cliques (with four sites) for the neighbourhood $\mathcal{N}^2$. As can be seen from Fig. 6.1 (b) a loop is formed with respect to the outer sites. In fact for any neighbourhood with cliques of three or more sites, a loop will be formed with respect to the outer sites. Therefore the closed-form estimate of $P(\mathbf{x}_A)$ only exists when the cliques are

restricted to the pairwise. That is Eq. (6.10) is only valid for the auto-models of the strong MRF. Proofs that these steps used to determine when Eq. (6.10) is valid are also presented by Bishop *et al.* [22].

6.6 Estimation of the strong LCPDF

The estimation of the strong LCPDF, $P(x_s|x_r, r \in \mathcal{N}_s)$, is performed by first estimating each marginal probability $P(\mathbf{x}_C),\ \forall\ C \in \mathcal{C}_s$ and then combining them to obtain $P(x_s|x_r, r \in \mathcal{N}_s)$. Each marginal probability $P(\mathbf{x}_C),\ \forall\ C \in \mathcal{C}_s$ is estimated in the same way as the nonparametric LCPDF estimate Eq. (5.12). However, in the case of the marginal estimates, Eq. (5.12) is calculated with respect to C rather than $\mathcal{N}_s$. The set of sample vectors, $\{Z_p, p \in S_y, \mathcal{N}_p \subset S_y\}$, are therefore modified to $\{Z_p, p \in S_y, C \subset S_y\}$. The dimensionality of z is modified to $d = |C|$, and the optimal window parameter h_{opt} is recalculated for each $C \in \mathcal{C}_s$, Eq. (5.11).

$$P(\mathbf{x}_C) = \frac{1}{nh_{opt}^d} \sum_{\substack{p \in S_y, \\ \mathcal{N}_p \subset S_y}} \exp\left[-\frac{1}{2h_{opt}^2}(\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}}(\mathbf{z} - \mathbf{Z}_p)\right] \tag{6.48}$$

where $\mathbf{z} = [x_C]^{\mathrm{T}},\ C \in \mathcal{C}_s$, and $n = |\{p \in S_y, \mathcal{N}_p \subset S_y\}|$. Given each marginal probability estimate $P(\mathbf{x}_C),\ \forall\ C \in \mathcal{C}_s$ the strong LCPDF $P(x_s, x_r, r \in \mathcal{N}_s)$ may then be estimated.

The *direct estimate technique* for determining $P(x_s, x_r, r \in \mathcal{N}_s)$ from its marginal probabilities is given by Eq. (6.10). However, this direct estimate technique is only valid for particular clique decompositions of Eq. (6.10) [21, 67, 88]. The steps for determining the existence of the direct estimate are given by Bishop *et al.* [22]. For the neighbourhood systems defined by Eq. (3.9), the steps identify that the direct estimate only exists for pairwise clique decompositions. That is, Eq. (6.10) is only valid for cliques restricted to no more than two sites, $|C| \leq 2$.

Although the direct estimate technique is limited to cases when only pairwise cliques are used, the *iterative proportional fitting technique* may be used for all clique decompositions of Eq. (6.10). Fienberg [67] and Bishop *et al.* [22] describe the iterative proportional fitting technique for a distribution defined in three dimensions. The technique is easily generalised to estimating $P(x_s, x_r, r \in \mathcal{N}_s)$ in multi-dimensions given the marginal probabilities $P(\mathbf{x}_C),\ \forall\ C \in \mathcal{C}_s$, Fig. 6.2.

Iterative proportional fitting technique

1. Given $P(\mathbf{x}_C),\ \forall\ C \in \mathcal{C}_s$ and $\delta \geq 0$.

2. Set $P(x_s, x_r, r \in \mathcal{N}_s)^{(0)} = P(\mathbf{x}_A)^{(0)} = \frac{1}{|\Omega_A|},\ \forall\ \mathbf{x}_A \in \Omega_A$

3. **For** $k = 0$ to ∞ **do**

3.1. **For** $\forall\ C \in \mathcal{C}_s$ **set**

$$P(\mathbf{x}_A)^{(k+1)} = \frac{P(\mathbf{x}_C)P(\mathbf{x}_A)^{(k)}}{\sum_{\mathbf{y}_{A-C} \in \Omega_{A-C}} P(\mathbf{x}_C \mathbf{y}_{A-C})^{(k)}},\ \forall\ \mathbf{x}_A \in \Omega_A$$

3.2. **If** $|P(\mathbf{x}_A)^{(k)} - P(\mathbf{x}_A)^{(k-1)}| < \delta,\ \forall\ \mathbf{x}_A \in \Omega_A$ **then** break

4. **Done**

Figure 6.2: Iterative proportional fitting technique

Regardless of the number of grey levels, $|\Lambda|$, and which cliques are used $C \in \mathcal{C}_s$, the minimum amount of memory space required to iteratively calculate the LCPDF is $2^{|\mathcal{N}_s|+1}$. This is because in order to calculate $P(x_s, x_r, r \in \mathcal{N}_s)$ via the iterative proportional fitting technique, Fig. 6.2, it is possible to reduce the configuration space Ω_A down to the binary space $\{(y_s == x_s), (y_r == x_r), r \in \mathcal{N}_s\}$. This gives $|\Omega_A| = 2^{|\mathcal{N}_s|+1}$. Even so, in our experiments run on the MasPar ®, we found that the memory requirements only allowed the iterative proportional fitting technique of $P(x_s|x_r, r \in \mathcal{N}_s)$ for $(|\mathcal{N}_s| + 1) < 10$. This meant that the iterative proportional fitting technique could only be used for neighbourhood systems $\mathcal{N}^1$ and $\mathcal{N}^2$. Although, we obtained the necessary decrease in run time in using the MasPar ®, it was at the expense of a comparatively reduced amount of available memory. Because of this we found that we could not estimate the LCPDF for neighbourhoods larger than ten sites.

A variation on the direct estimate technique Eq. (6.10) is the *simple estimate*,

$$\log P(\mathbf{x}_A) = \sum_{\substack{C \in \mathcal{C}_s, \\ C \not\subset C' \in \mathcal{C}_s}} \log P(\mathbf{x}_C), \tag{6.49}$$

where C, C' are cliques. The simple estimate yields an estimate that can be calculated on the MasPar ® for large neighbourhood systems and various clique decompositions. The simple estimate is like the direct estimate except for the fact

that it only incorporates those marginal probabilities $P(\mathbf{x}_C)$ defined on cliques $\{C \in \mathcal{C}_s, C \not\subset C' \in \mathcal{C}_s\}$. These are the major cliques contained in the local clique set. The clique decomposition summation of Eq. (6.49) is then only performed over those cliques contained in the local clique set which are not subsets of other cliques contained in the local clique set.

The simple estimate Eq. (6.49) does not give the correct estimate, but it does give an estimate that may be used in the synthesis process. As only the major marginals are used in the summation of Eq. (6.49) and are not counterbalanced by smaller marginals as in Eq. (6.10), the estimate Eq. (6.49) of $P(x_s, x_r, r \in \mathcal{N}_s)$ is biased towards those sites $\{s, r, r \in \mathcal{N}_s\}$ contained in multiple major cliques. Intuitively this means the LCPDF will be more "peaked." For texture synthesis purposes a "peaked" LCPDF means that the *Gibbs sampler* will behave more like the *ICM algorithm* [78]. Both algorithms may be used for synthesising textures and are explained in Chapter 4.

6.7 Goodness-of-fit testing

As the strong MRF is equivalent to the ANOVA model it is reasonable to consider if the goodness-of-fit tests used for the ANOVA model are applicable for the strong MRF model.

The following test statistics are generally used for ANOVA:

$$X^2 = \sum \frac{(\text{Observed - Expected})^2}{\text{Expected}}, \tag{6.50}$$

$$G^2 = 2\sum(\text{Observed}) \log\left(\frac{\text{Observed}}{\text{Expected}}\right), \tag{6.51}$$

where the summation in both cases is over all cells in the table, or in our case bin's in the multi-dimensional histogram. Eq. (6.50) is Pearson's chi-square statistic, and Eq. (6.51) is the log-likelihood ratio statistic [67]. Both statistics are special cases of the power divergence statistic [169].

If the fitted model is correct and the total sample size is large (*i.e.,* the sample size is at least ten times the number of cells in the table), both X^2 and G^2 have approximate χ^2 distributions with degrees of freedom equal to the number of bins minus the number of parameters in the model. However Larntz [129] suggests that for testing a nominal 0.05 level of significance, a minimum expected bin count of

approximately 1.0 is sufficient. Under such conditions when there are relatively few samples with respect to the number of bins, Larntz [129] found that Pearson's chi-square statistic X^2 performed better than the log-likelihood ratio statistic G^2 [67].

Under the sparseness assumption (when the number of bins tends to infinity, but the expected count in each bin remains constant) the X^2 and G^2 statistic are asymptotically normal with a mean and variance given by Morris [146] and qualified by Read and Cressie [169]. This result was later extended by Dale [51] for product-multinomial sampling (*i.e.,* when one marginal constraint is given). However we are interested in the case where there are multiple constraints, ones that are imposed by our marginal distributions for each clique. Koehler [123] provided a Monte Carlo study of sparse contingency tables requiring parameter estimation, which is equivalent to having multiple constraints. Koehler [123] observed that the limiting normal distribution was more accurate for the G^2 statistic than for the X^2 statistic.

However the limiting normal distribution for the X^2 and G^2 statistic is really only technically correct when given large amounts of sample data. The accuracy of the goodness-of-fit test for small amount of sample data can be improved with a modified version of Fisher's exact test used for 2×2 contingency tables [22, 68, 202]. Fishers exact goodness-of-fit test is given by the sum of all hypergeometric probabilities of 2×2 contingency tables that fit the marginal constraints and for which the probabilities are less than the hypergeometric probability of the observed table. This type of analysis can be extended to larger contingency tables [169]. Mehta and Patel [141] present an algorithm that calculates only those tables whose probabilities are less than the probability of the observed table. This is supposed to be very efficient for large tables. Alternatively, Agresti *et al.* [2] takes a sample set of possible tables to estimate the exact goodness-of-fit. This is analogous to a Monte Carlo approach [84]. Verbeek and Kroonenberg [202] provide a useful survey of algorithms designed to calculate the exact goodness-of-fit in large contingency tables with a small amount of sample data.

Unfortunately we found that these tests were not able to successfully discriminate the goodness-of-fit of different orders of the strong MRF model due to the high correlation between neighbouring pixel values and therefore the marginal constraints. This problem was brought to light when we tried to use entropy measurements to determine the optimal set of cliques to use for the strong MRF. We found that it was virtually impossible to discriminate the cliques on the basis of the entropy of their marginal distributions, Appendix B.3. We therefore conclude that, although

not optimal, the best practical approach for performing a goodness-of-fit test for the strong MRF model is to again compare synthetic textures derived from the model with the original texture. If the synthetic textures are found to be subjectively similar to the original, we conclude that order of the strong MRF is amenable to the texture. Again "chi-by-eye" is better than no estimate if no other goodness-of-fit test is available.

Chapter 7

Synthesising Texture

This chapter details our own multiscale texture synthesis algorithm incorporating our novel pixel temperature function. Other variations of this synthesis algorithm are also outlined. From this chapter we have published four papers showing the merits of the nonparametric and strong nonparametric MRF models [152, 155, 153, 156]. Of particular interest is the texture synthesis by the strong nonparametric MRF model, which identified textures that could be completely represented by just third order statistics [153].

7.1 Introduction

Texture synthesis is a means of testing whether the LCPDF has captured the textural characteristics required to model a particular texture. How specific the required texture characteristics need to be is governed by the intended application of the texture model. The aim of this thesis is to develop a texture model suitable for open-ended texture classification, therefore, able to capture *all* textural characteristics unique to a training texture. We propose that if the model is capable of synthesising texture that is visually indistinguishable from the training texture, then it has captured *all* the visual characteristics of that texture.

There have been quite a few attempts at synthesising textures, but none of the conventional techniques have produced a general model for natural textures [95]. However new methods based on stochastic modelling of various multi-resolution filter responses have produced impressive results [23, 211, 103, 149]. Popat and Picard [164] successfully used a high order causal nonparametric multiscale MRF

model to synthesise structured natural textures. In fact our approach is indicative of theirs, but where they suffered from *phase discontinuity* we used our method of *local annealing* to synthesise highly representative examples of natural textures.

We perform texture synthesis via a multiscale synthesis algorithm incorporating our novel pixel temperature function. In the synthesis process, the pixel temperature function is responsible for "compressing" various dimensions of the multi-dimensional histogram (or the nonparametric LCPDF). The degree of compression for each dimension is judged by how much we want a pixel to be conditionally dependent on its corresponding neighbourhood pixel. If all the dimensions associated with the neighbourhood pixels were completely compressed, the resultant LCPDF would amount to a simple normalised histogram, and each pixel would no longer be conditionally dependent on its neighbouring pixels. We use this approach to gently "grow" the multi-dimensional histogram from a simple histogram to the desired dimensionality as the texture undergoes synthesis. This is analogous to annealing [82], but in this case it allows us to use large multi-dimensional histograms (or LCPDFs with large neighbourhood systems) to model texture.

The multiscale synthesis algorithm uses stochastic relaxation (SR) [50, 80, 82], Section 4.4, to synthesise an MRF at each scale. SR starts with an image and iteratively updates pixels in the image with respect to the LCPDF. This generates a sequence of images $\{\mathbf{X}(0), \mathbf{X}(1), \ldots, \mathbf{X}(n)\}$ with the property,

$$\lim_{n\to\infty} P(\mathbf{X}(n) = \mathbf{x} | \mathbf{X}(0) = \mathbf{x}(0)) = \Pi(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega, \tag{7.1}$$

Two well-known SR algorithms are the Metropolis algorithm [142], Fig. 4.4, and the Gibbs sampler [82], Fig. 4.5. Besag [19] introduced deterministic relaxation algorithm called the *Iterative conditional modes (ICM)* algorithm, Fig. 4.6. Synthesis by the Gibbs sampler tends to converge to a texture defined by the equilibrium condition Eq. (7.1), whereas for the ICM algorithm, the synthesis tends to a texture more conditional on the starting image $\mathbf{X}(0) = \mathbf{x}(0)$.

In the following notation $(\mathbf{Y} = \mathbf{y} \in \Omega)$ will refer to the training texture and $(\mathbf{X} = \mathbf{x} \in \Omega)$ will refer to the synthesised texture. That is the LCPDF is estimated from the image $(\mathbf{Y} = \mathbf{y} \in \Omega)$, and used to synthesise images $(\mathbf{X} = \mathbf{x} \in \Omega)$. The set of sites over which $\mathbf{Y}$ is defined will be denoted as S_y. While S will denote those sites on which $\mathbf{X}$ is defined.

7.2 Multiscale relaxation

The basic concept of Multiscale Relaxation (MR) is to relax the field at various "resolutions." The advantage is that some characteristics of a field are more prominent at some resolutions than at others. For example, high frequency characteristics are prominent at high resolutions, but are hidden at low resolutions. In contrast, low frequency (global) characteristics are more easily resolved at low resolutions [86].

A problem with the single-scale relaxation process is that the global image characteristics evolve indirectly in the relaxation process [26, 86, 193]. Typically these global image characteristics only propagated across the image lattice through local interactions. This results in a slow evolution process which is easily disrupted by phase discontinuities, see Section 7.3. Long relaxation times are required to obtain an equilibrium, as defined by Eq. (7.1).

MR attempts to overcome the problem of the global image characteristics evolving slowly and indirectly by implementing SR at various resolutions; first at a low resolution and then progressively at higher resolutions [86, 23, 134, 27, 8]. The information obtained from SR at one resolution is used to constrain the SR at the next highest resolution. By this method, global image characteristics that have been resolved at a low resolution are infused into the relaxation process at the higher resolution. This helps reduce the number of iterations required to obtain equilibrium [193].

As a consequence of implementing MR, Bouman and Liu [26] and Derin and Won [55] found, by experimentation, that it helped the ICM algorithm to find convergence to the global maximum of the joint distribution Π. Under SR the ICM algorithm tended to cause convergence to a local maximum over Π that was dependent on the initial image. This MR observation, by Bouman and Liu [26] and Derin and Won [55] for the ICM algorithm, was suggested to be due to the various image characteristics being better infused into the synthesised image over multiple resolutions. Coarse image characteristics were found to be better infused into the image at low resolutions, while fine image characteristics were found to be better infused into the image at high resolutions. Basically, the relaxation processes at a high resolution was found to be less susceptible to being trapped in a local maxima if the desired coarse image characteristics had been pre-infused into the image at a lower resolution.

As the multiscale relaxation process first infuses the coarse image characteristics

into the synthetic image, and then at each higher resolution successively adds more detail, the LCPDF does not need to be correctly defined for all resolutions. The LCPDF only needs to contain enough information to add the extra detail. This is an advantage when the LCPDF is defined over a large neighbourhood. Under such circumstances the domain of the LCPDF usually only contains a small area where there is relevant information. Most of the domain could be quite easily modelled as a uniform distribution. This is due to the relatively small amount of data available to construct the LCPDF over the large domain. This means that if the LCPDF was used in single-scale relaxation, the relaxation process would tend to languish in the domain of the LCPDF where it was relatively uniform and therefore non convergent. The multiscale relaxation process overcomes this liability by restricting the domain over which the LCPDF is sampled. As mentioned before, this is done by pre-infusing course image characteristics into the synthetic image.

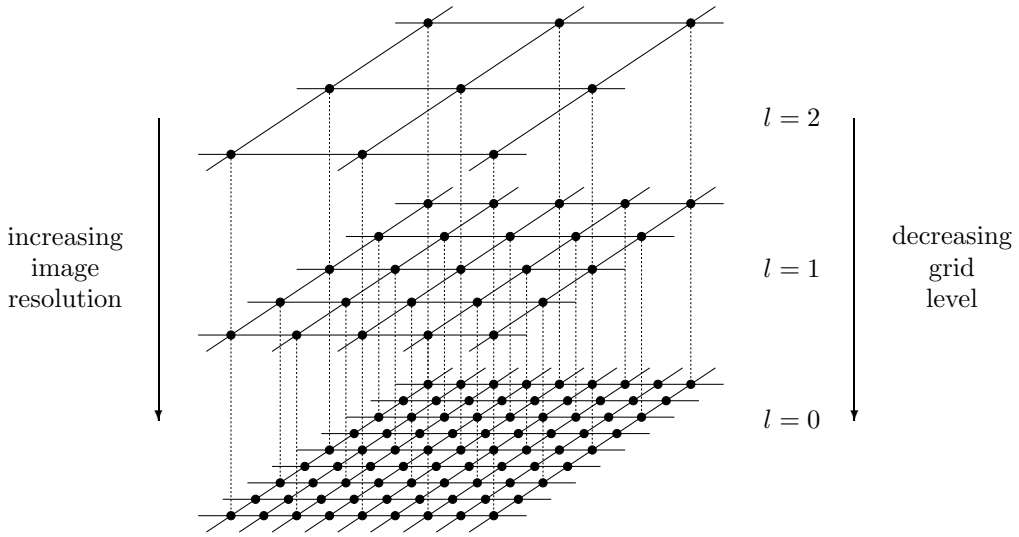


Figure 7.1: Possible grid organisation for multiscale modelling of an MRF. A small portion of three levels of a 2:1 multigrid hierarchy is shown. Only connections representing nearest-neighbour interactions are included.

MR may be best described through the use of a multigrid representation of the image as shown in Fig. 7.1. The grid at level $l = 0$ represents the image at the original resolution, where each intersection point '•' is a site $s \in S$. The lower resolutions, or higher grid levels $l > 0$, are decimated versions of the image at level $l = 0$. This multigrid representation is also used by Popat and Picard [164].

Given an $N \times M$ image, $\mathbf{X}$, we define the rectangular lattice on which to represent

this image at grid level $l = 0$ as,

$$S = \{s = (i, j) : 0 \leq i < N, 0 \leq j < M\}, \tag{7.2}$$

The multigrid representation of the image, $\mathbf{X}$, is then the set of images, $\mathbf{X}^l$, for grid levels $l \geq 0$. The image, $\mathbf{X}^l$, is defined on a lattice $S^l \subset S$, where,

$$S^l = \left\{ s = (2^l i, 2^l j) : 0 \leq i < \frac{N}{2^l}, 0 \leq j < \frac{M}{2^l} \right\}. \tag{7.3}$$

The set of sites S^l at level l, represents a decimation of the previous set of sites S^{l-1} at the lower grid level $l - 1$. On this multiscale lattice representation, we need to redefine the neighbourhood system for each grid level $l \geq 0$. Therefore, we define the neighbourhood $\mathcal{N}_s^l,\ s \in S^l$, with respect to order o as,

$$\mathcal{N}^l_{s=(2^l i, 2^l j)} = \{r = (2^l p, 2^l q) \in S^l : 0 < (i - p)^2 + (j - q)^2 \leq o\}. \tag{7.4}$$

The image $\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$ is defined with respect to the image $\mathbf{X}^l = \mathbf{x}^l$ by the function q whereby,

$$X^{l+1}_{s=(i,j)} = q\left(x^l_{(i,j)}, x^l_{(i+2^l,j)}, x^l_{(i,j+2^l)}, x^l_{(i+2^l,j+2^l)}\right) \tag{7.5}$$

Two typical forms of the function q are

1. **local averaging**

$$q(x_1, x_2, x_3, x_4) = \frac{x_1 + x_2 + x_3 + x_4}{4} \tag{7.6}$$

2. **local decimation**

$$q(x_1, x_2, x_3, x_4) = x_1 \tag{7.7}$$

The multigrid displayed in Fig. 7.1 shows the connections involved in the decimation approach.

Other typical approaches to the multiscale representation of an image involve Gaussian Filters, Laplacian of Gaussian Filters [30, 211], Gabor Filters [149, 206, 211], and wavelets [8, 25, 24, 103, 134]. The advantage of the Laplacian, Gabor, and wavelet filters is that they can be orthogonalised. When constructing a multiscale representation of an image, they allow each resolution to be derived from an

orthonormal filter. The advantage of such an approach is that each resolution is then independent of the others [44]. This multiscale representation supports data compression. However the disadvantage of this representation is that the higher resolutions are no longer dependent on the lower resolutions. On the other hand, the advantage of using decimation in the multiscale representation is that not only is it easy to constrain the relaxation process from one resolution to another, but there is also no loss of information in the representation. In the synthesis process, this is more crucial than finding a compact representation. If, on the other hand, the aim was to model texture for classification purposes, then maybe using orthonormal wavelets would be an advantage.

The MR algorithm is formed by constraining an SR processes at each level, l, by the result of the SR process at the previous level, $l+1$. In [26] the relaxation process was constrained by initialising the image $\mathbf{X}^l = \mathbf{x}^l$ with respect to the previous image $\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$ and then using the ICM relaxation algorithm. We, on the other hand, have chosen the method proposed by Gidas [86], where the constraint imposed by the previous image, $\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$, is maintained through the entire SR process at level l and successive levels $k < l$. The constraint is such that at any point through the SR process at level l, the image $\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$ may still be obtained from $\mathbf{X}^l$, by the multigrid representation.

The MR algorithm starts at the lowest resolution (highest grid level) with a random image, $\mathbf{X}^{L-1} = \mathbf{x}^{L-1} \in \Omega$, defined on an appropriate lattice S^{L-1}, where L is the number of grid levels. The random image $\mathbf{X}^{L-1} = \mathbf{x}^{L-1}$ then undergoes SR until an equilibrium state is reached. The relaxed image $\mathbf{X}^{L-1} = \mathbf{x}^{L-1}$ is then used to constrain further SR at lower grid levels. The MR algorithm moves sequentially down from one level to the next. At each level, SR of image, $\mathbf{X}^l$, is constrained by the image $\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$. This continues until the final SR is performed at level $l = 0$. The final image $\mathbf{X}^0 = \mathbf{x}^0$ is the new synthetic texture.

At each grid level $0 \leq l < L$, the SR algorithm requires a valid LCPDF. Gidas [86] showed how to calculate the LCPDF at level l given a parametric LCPDF at level $l-1$. For a nonparametric LCPDF, as we use, the LCPDF at level l has to be calculated directly from a training image, $\mathbf{Y}$, of homogeneous texture. To calculate the LCPDF at level l, the image, $\mathbf{Y}$, is first scaled to the corresponding resolution to obtain an image, $\mathbf{Y}^l$. From this image the neighbourhood function $\mathcal{N}^l$ is used to draw the sample data from which to construct the LCPDF with the appropriate density estimation. Although the computation time required to calcu-

late the LCPDF at each level decreases as the level increases, the reliability of the estimation also decreases due to the decreasing amount of available data. This effect is accounted for in the density estimation scheme with an increase in the window parameter h_{opt} which in turn "smoothes" the LCPDF.

The decrease in sampling data can be counteracted by resampling the multigrid at shifted pixel locations. That is, if the multigrid structure reduces the image size by say four at each ascending grid level, then the multigrid can be resampled at four shifted pixel locations to produce four separate representations of the image at that grid level. This would maintain the same amount of data for each grid level, while still performing the necessary subsampling. However, if such a scheme was implemented, then consideration would need to be given to two possible problems. First, if the subsampling scheme of the multigrid structure incorporated filtering, then this would result in an increase in correlation between data samples at each ascending grid level. Secondly, if no filtering was incorporated into the subsampling scheme of the multigrid structure, then there would lie the distinct possibility that the image representations at the same grid level could become statistically dissimilar. This problem would be especially apparent for a highly structured texture, *e.g.,* consider the effect on a checkerboard pattern when using the decimation subsampling scheme (see Section 7.2.2). There is high potential that these problems will have a detrimental effect on the simulation algorithm. However, for the classification algorithm, they are likely to be inconsequential.

As mentioned previously we use the Gidas [86] MR algorithm whereby the constraint imposed by the previous image $\mathbf{X}^{l+1}$ is maintained through the entire relaxation process at level l. This is accomplished by multiplying the joint distribution Π by the probability $p^l(\mathbf{X}^{l+1} = \mathbf{x}^{l+1} | \mathbf{X}^l = \mathbf{x}^l)$. For notational purposes the probability p^l will be rewritten as $p^l(\mathbf{x}^{l+1}|\mathbf{x}^l)$.

The probability $p^l(\mathbf{x}^{l+1}|\mathbf{x}^l)$ is usually chosen to be a product of local conditional probabilities. Given a site $r = (i, j) \in S^{l+1}$ whose pixel is directly dependent on the lower grid pixels at the sites $S^l(r = (i, j)) = \{(i, j), (i+2^l, j), (i, j+2^l), (i+2^l, j+2^l)\}$, then

$$p^l(\mathbf{x}^{l+1}|\mathbf{x}^l) = \prod_{r \in S^{l+1}} p_r^l(x_r^{l+1} | x_t^l, t \in S^l(r)), \tag{7.8}$$

When a pixel at site s is updated in MR, it is updated with respect to the LCPDF and the local conditional probability $p^l(\mathbf{x}^{l+1}|\mathbf{x}^l)$. Therefore the LCPDF

$P^l(x_s^l|x_r^l, r \in \mathcal{N}_s)$ may be redefined as,

$$P^l(x_s^l|.) = P^l(x_s^l|x_r^l, r \in \mathcal{N}_s^l)p_r^l(x_r^{l+1}|x_t^l, s, t \in S^l(r)). \tag{7.9}$$

A special case of the probability p_r^l is when it is equated to the Kronecker function such that,

$$p_r^l(x_r^{l+1}|x_t^l, t \in S^l(r)) = \delta(x_r^{l+1}, q(x_t^l, t \in S^l(r))), \tag{7.10}$$

where $\delta(\alpha, \beta) = 0$ if $\alpha \neq \beta$, and $\delta(\alpha, \beta) = 1$ if $\alpha = \beta$. The function q is the scaling function discussed previously. The constraint imposed by the Kronecker function δ effectively restricts the sampling of the LCPDF to those pixel values $\lambda_s \in \Lambda$ for which

$$x_r^{l+1} = q(\lambda_s, x_t^l, t \neq s, t \in S^l(r)), \qquad s \in S^l(r) \tag{7.11}$$

There are two general approaches to the scaling function q: decimation or averaging. It may seem that the local averaging approach would be the more intuitive choice for representing the image at lower resolutions. However when applying MR there are distinct advantages in using local decimation. For one, MR requires synthesising an image at one level and propagating the results of the synthesis to lower levels. The propagation of information down the levels involves the inverse of q which is easier under local decimation rather than local averaging [86]. Another less obvious advantage is that the state space Λ in the decimation approach is the same at all resolutions.

7.2.1 Averaging approach

In the averaging approach the scaling function is defined over a 2×2 set of pixels whereby,

$$x_r^{l+1} = q(x_t^l, t \in S^l(r)) = \frac{1}{4} \sum_{t \in S^l(r)} x_t^l \tag{7.12}$$

where the result of the division is rounded to the nearest integer. Given the pixel value x_r^{l+1} from the image X^{l+1} and the current pixel values $\{x_t^l, t \in S^l(r)\}$ from the image X^l, a pixel $x_s^l, s \in S^l(r)$ can be updated to a value $\lambda_s \in \Lambda$ with respect to the LCPDF under the constraint imposed by Eq. (7.12). If $\Lambda = \mathcal{Z}$, then there are just four consecutive values $\lambda_s \in \Lambda$ for which $x_s^l = \lambda_s$ is a solution to Eq. (7.12). Therefore the LCPDF is at most only sampled over four consecutive values $\lambda_s \in \Lambda$ imposed by the constraint of Eq. (7.12). This means that the information contained

in the LCPDF is under-utilised. To overcome this sampling deficiency but still constrain the SR process, the sampling process can be expanded to included two or more pixels $\{x_s^l, s \in S_{sub}^l(r) \subseteq S^l(r)\}$. The image $\mathbf{X}^l$ is then relaxed over two or more pixels at the same time with respect to the LCPDF defined as,

$$\begin{aligned} P^l(x_s^l, s \in S_{sub}^l(r)|.) &= P^l(x_s^l, s \in S_{sub}^l(r)|x_t^l, t \in \mathcal{N}_s) p_r^l(x_r^{l+1}|x_t^l, t \in S^l(r)) \\ &\approx p_r^l(x_r^{l+1}|x_t^l, t \in S^l(r)) \prod_{s \in S_{sub}^l(r)} P^l(x_s^l|x_t^l, t \in \mathcal{N}_s). \end{aligned} \quad (7.13)$$

This means that instead of the state space being restricted to one $\lambda_s \in \Lambda$, the state space is expanded to $\lambda_s^n = \{\lambda_s, s \in S_{sub}^l(r)\} \in \Lambda^n$ with $2 \leq n \leq 4$. This new state space is restricted by the same condition of Eq. (7.12) whereby,

$$x_r^{l+1} = q(\lambda_s^n, x_t^l : t \notin S_{sub}^l(r), t \in S^l(r)), \qquad \lambda_s^n = \{\lambda_s, s \in S_{sub}^l(r)\} \quad (7.14)$$

and therefore the size of the new restricted state space $\leq 4|\Lambda|^{n-1}$.

Unfortunately in experiments when texture was synthesised via the averaging approach, it was found that the large size of the state space impeded the speed of the relaxation algorithm. The time taken to complete one iteration of the relaxation process was found to be so long that it made the process impractical.

7.2.2 Decimation approach

In the decimation approach the scaling function is defined over a 2×2 set of pixels whereby

$$x_r^{l+1} = q(x_t^l, t \in S^l(r)) = x_r^l. \quad (7.15)$$

In this case a direct mapping of the pixel values exist from one level down to the next, which means that the inverse function of q exists.

$$\begin{aligned} q^{-1}(x_{r=(i,j)}^{l+1}) &= \{x_t^l, t \in S^l(r)\} \\ &= \{x_{(i,j)}^l = x_{(i,j)}^{l+1}; \lambda_{(i+2^l,j)}^l, \lambda_{(i,j+2^l)}^l, \lambda_{(i+2^l,j+2^l)}^l\} \qquad \lambda \in \Lambda. \end{aligned} \quad (7.16)$$

Sampling the LCPDF with respect to $s \in S^l(r)$ results in two possible cases. If the site $s = r$ then the value of x_s^l is fixed by the inverse function of Eq. (7.16). If on the other hand $s \neq r, s \in S^l(r)$ then the LCPDF is sampled over the complete state space Λ. Therefore the constraint imposed on the SR at level l by the image

$\mathbf{X}^{l+1} = \mathbf{x}^{l+1}$ fixes the pixels

$$X_r^l = x_r^{l+1}, \qquad \forall r \in S^{l+1} \tag{7.17}$$

and the rest of the pixels undergo non-restricted SR. For the ease of implementing the MR constraint, we have chosen the decimation approach for multiscale representation of an image $\mathbf{X}$.

7.3 Pixel temperature function

To better incorporate the SR process with the MR pre-infused synthetic image, we introduce our own novel pixel temperature function. With the pixel temperature function, we can directly define how the MR constraint is imposed on the SR process. As a consequence of using the pixel temperature function, an equilibrium state exists which may be used to determine when the SR process at one level may be terminated to be proceeded by the SR process at the next level. We have also found that the multiscale relaxation algorithm proposed in this thesis, which incorporates our novel pixel temperature function, produces synthetic textures with minimal phase discontinuities [152]. An example of a texture phase discontinuity is shown in Fig. 7.2.

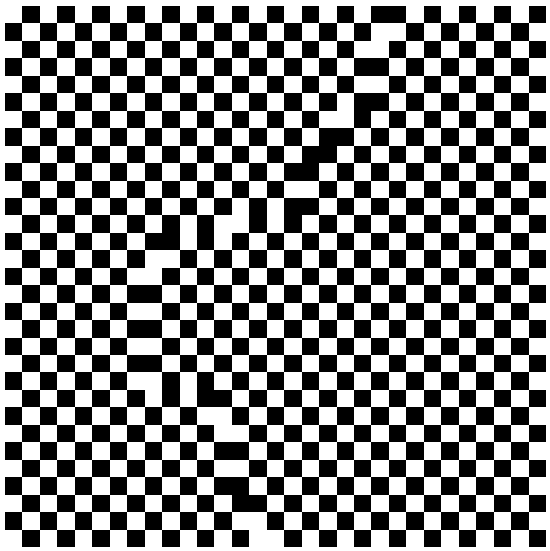


Figure 7.2: Example of phase discontinuity. There exists a visual separation between two realizations of the same texture.

A phase discontinuity occurs when there exists more than one maximum for Π, be it a local or global maximum. Under relaxation, the value of a pixel in the image will tend to the maximum of the LCPDF. A set of pixels within a local area of the image will tend to a maximum given by the joint distribution Π. It is the set of pixels that exhibit a maximum in Π that determines the global image characteristics of a texture. The relaxation of the image via the LCPDF only indirectly influences these global image characteristics. Two different maxima in Π correspond to two textures with different global image characteristics. Therefore if two different local areas within the image are propagating different global image characteristics via the relaxation process, a phase discontinuity will occur at the border between the two areas as seen in Fig. 7.2. Multiscale relaxation helps reduce the occurrence of phase discontinuities in synthetic textures. This is because the global image characteristics that are resolved at the lower resolutions, with long interconnections, represents phase continuity over larger distances than at higher resolutions.

The aim of our pixel temperature function is to define a "confidence" associated with a pixel, that its value has been sampled from an unconstrained LCPDF. Total confidence in a pixel value will occur when it has been sampled from an unconstrained LCPDF. Initially in the MR algorithm, the SR algorithm is constrained by the realisation attained at the previous grid level. Pixels that are from the previous grid level do not undergo further stochastic relaxation, and therefore can be regarded as having full "confidence" in their pixel values. New pixels that are introduced at the current grid level must undergo a complete course of stochastic relaxation, and therefore must have no "confidence" in their initial pixel values. If the confidence associated with a pixel is used to constrain the LCPDF so that it is less conditional on those pixels with lower confidence, then it follows that the LCPDF will only become unconstrained when all its neighbouring pixels have full confidence. Also, the MRF will not reach an equilibrium state until all LCPDFs are unconstrained, and this will only occur when all pixels at the current grid level attain full confidence.

Basically an LCPDF should not be fully conditioned on a value until that value has been sampled from a fully conditioned LCPDF. Note, if the LCPDF was perfectly defined, then this would not be necessary, but because we have limited data and a huge domain for the LCPDF, it is better to control the access to where viable data exists within the domain of the LCPDF.

Each pixel is given its individual temperature t_s, representing the confidence associated with the pixel x_s. The confidence is expressed as $(1 - t_s)$ for $0 \leq t_s \leq 1$,

such that $t_s = 0$ represents complete confidence, and $t_s = 1$ none at all. In the MR algorithm, the confidence or temperature associated with each pixel is used to modify the dimensionality of the LCPDF. This is done so that the conditional dependence of the LCPDF is strongest on those pixels with $t_s \to 0$, and weakest for those with $t_s \to 1$. The pixel temperature is incorporated into the LCPDF by modifying the form of $(\mathbf{z} - \mathbf{Z}_s)$ in Eq. (5.12).

Given an image $\mathbf{X}$ on which to define the LCPDF, and a training image $\mathbf{Y}$ from which to take the data samples, the estimate of the LCPDF, with respect to a neighbourhood system $\mathcal{N}^l$, is given by Eq. (5.12) as,

$$\hat{P}(x_s|x_r, r \in \mathcal{N}_s^l) = \frac{1}{Q(\mathbf{z})} \sum_{\substack{p \in S_y^l, \\ \mathcal{N}_p^l \subset S_y^l}} \exp\left[-\frac{1}{2h_{opt}^2}(\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}}(\mathbf{z} - \mathbf{Z}_p)\right], \tag{7.18}$$

where $Q(\mathbf{z})$ is the normalising function, and constant with respect to s,

$$Q(\mathbf{z}) = \sum_{\lambda_s \in \Lambda} \sum_{p \in S_y^l, \mathcal{N}_p^l \subset S_y^l} \exp\left[-\frac{1}{2h_{opt}^2}(\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}}(\mathbf{z} - \mathbf{Z}_p)\right]. \tag{7.19}$$

The estimate of the LCPDF is for the image $\mathbf{X}^l$ at the site $s \in S^l$. The d-dimensional column vector $\mathbf{z} = \mathrm{Col}[x_s, x_r, r \in \mathcal{N}_s^l]$. The sample data $\mathbf{Z}_p$ is taken from the image $\mathbf{Y}$ defined on the lattice S_y^l, where $\mathbf{Z}_p = \mathrm{Col}[y_p, y_q, q \in \mathcal{N}_p^l]$ for which $\mathcal{N}_p^l \subset S_y^l$. The vector $\mathbf{Z}_p$ may be redefined with respect to the neighbourhood $\mathcal{N}_s^l$ defined on the lattice S^l, such that,

$$\mathbf{Z}_p = \mathrm{Col}[y_p, y_q, q \in \mathcal{N}_p^l] = \mathrm{Col}[y_p, y_{r-s+p}, r \in \mathcal{N}_s^l]. \tag{7.20}$$

The pixel temperature is then incorporated into the LCPDF by modifying the form of $(\mathbf{z} - \mathbf{Z}_p)$ in Eq. (7.18) to,

$$(\mathbf{z} - \mathbf{Z}_p) = \mathrm{Col}[x_s - y_p, (x_r - y_{r-s+p})(1 - t_r), r \in \mathcal{N}_s^l], \tag{7.21}$$

where the pixel temperature t_r is from the same site as the pixel value $x_r, r \in S^l$ on the image $\mathbf{X}^l$.

Before the SR algorithm starts at level l, those pixels which were relaxed at the previous level, $l + 1$, are given a pixel temperature $t_s = 0 \ \forall s \in S^{l+1}$ *i.e.*, complete confidence. The other pixels have their pixel temperatures initialised to

$t_s = 1 \; \forall s \notin S^{l+1}$ *i.e.,* no confidence. This is illustrated in Fig. 7.3, where those pixels represented as '•'s are given a pixel temperature $t_s = 0$, and the other pixels represented as '∘'s are given a pixel temperature $t_s = 1$, whose pixel values undergo SR.

$$t_s = \begin{cases} 0 & s \in S^{l+1} \\ 1 & \text{otherwise} \end{cases} \tag{7.22}$$

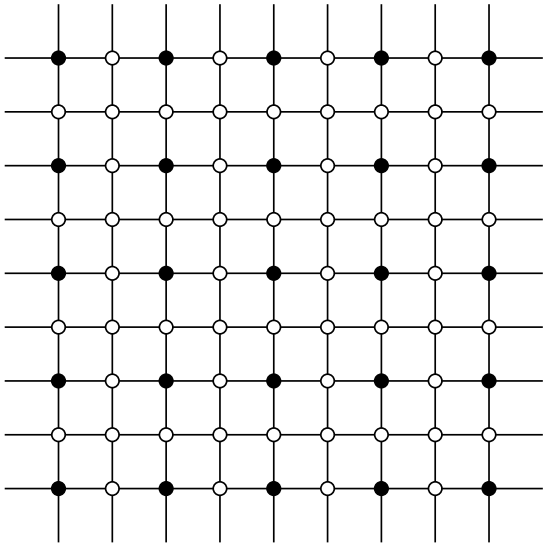


Figure 7.3: A grid at level l. A • represents a site $s \in S^{l+1}$. A ∘ represents a site $s \notin S^{l+1}$ whose pixel is to undergo SR.

If the pixel value at site $s \in S^l$ were to undergo relaxation and the pixel temperature $t_r = 1$ for a site $r \in \mathcal{N}_s^l$, then the LCPDF for site s would not be conditional on that pixel x_r. Therefore when $t_r = 1$ the pixel x_r has no effect on the LCPDF for the site s, but as $t_r \searrow 0$ the pixel value x_r has an increased effect on the LCPDF estimate. This has similar connotations to Geman and Geman's temperature parameter T [82] for an LCPDF. When $T \sim \infty$ the pixel values $x_r, r \in \mathcal{N}_s^l$ have no effect on the LCPDF for a site s, but as $T \searrow 0$ the pixel values $x_r, r \in \mathcal{N}_s^l$ have an increasing effect.

Geman and Geman [82] defined a temperature parameter T in their Gibbs sampler which was used to control the degree of "peaking" in the LCPDF. The temperature parameter ranged from $0 \leq T < \infty$. For $T \sim \infty$ the LCPDF was "flat" and uniform. For $T = 1$ the LCPDF was in its normal state, and for $T = 0$ the LCPDF was "peaked" at the mode and zero everywhere else. The temperature parameter was used as part of an annealing algorithm, whereby the temperature started at

a high value and gradually decreased to one (or zero). This was done in order to gradually relax the image to its global maximum defined by the joint probability distribution Π. We can obtain a similar temperature function based on our pixel temperature function by defining a "local" temperature function T_s such that,

$$(1 - t_s) = \frac{1}{\sqrt{T_s}},\ 0 \leq T_s < \infty,\ \forall s \in S. \tag{7.23}$$

The local temperature T_s is then like a local version of the temperature parameter T as defined in [82]. In this way T_s varies from $\infty \searrow 1$ to obtain t_s varying from $1 \searrow 0$. In any case, the function of our local pixel temperature may be regarded as an implementation of *local annealing* in the relaxation process.

Just as the Gibbs sampler requires a cooling schedule to specify the rate at which the temperature parameter T reduces to zero, so does the local temperature $T_s,\ s \notin S^{l+1}$. As each site has its own local temperature T_s, so too can each site have its own cooling schedule. This means that the rate at which a pixel is cooled can be made dependent on its neighbouring temperatures. In high temperature surroundings a pixel could be cooled fairly rapidly, but in cool surroundings a slower cooling rate would be advisable.

High temperatures produce images that are not spatially correlated, but as the temperatures decrease the correlations induced by the LCPDF become more prevalent. These spatial correlations represent the image characteristics. It is important that these characteristics evolve correctly and therefore the LCPDF should not change too rapidly after each iteration; as the surrounding pixel temperatures drop, the cooling rate should be slowed.

We relate pixel temperature to pixel confidence, where pixel confidence is associated with the probability that x_s is the correct pixel value for the site s. Full pixel confidence occurs when x_s is sampled from an LCPDF at equilibrium, or when the LCPDF is completely conditional on its neighbouring pixel values at sites $r \in \mathcal{N}_s^l$. An LCPDF at equilibrium means the pixel temperatures $t_r = 0,\ \forall r \in \mathcal{N}_s^l$, and therefore the estimate of the LCPDF in Eq. (7.18) is not modified by pixel temperature t_r in Eq. (7.21).

The confidence associated with the pixel value x_s is then dependent on the pixel temperatures $t_r, r \in \mathcal{N}_s^l$ which are used to modify the respective LCPDF. Therefore, the following formula has been chosen to describe the confidence associated with a pixel value, x_s, that has been sampled from an LCPDF modified by the pixel

temperatures $t_r, r \in \mathcal{N}_s^l$,

$$t_s = \frac{\sum_{r \in \mathcal{N}_s^l} t_r}{|\mathcal{N}_s^l|}. \tag{7.24}$$

The expression $|\mathcal{N}_s^l|$ is the total number of sites in $\mathcal{N}_s^l$.

From Eq. (7.24) the pixel temperature t_s will only equal zero if $t_r = 0,\ \forall r \in \mathcal{N}_s^l$. Therefore t_s will only converge to zero but never equal zero. This means with the cooling schedule in the present form, the SR will not reach a state of equilibrium. The following modification was used to ensure that the SR reached a state of equilibrium.

$$t_s = \max\left\{0, \frac{-\xi + \sum_{r \in \mathcal{N}_s^l} t_r}{|\mathcal{N}_s^l|}\right\}. \tag{7.25}$$

where ξ is a constant such that $\xi > 0$. We used $\xi = 1$. When using Eq. (7.25) to modify the pixel temperature after its site has been relaxed, eventually all $t_s, s \in S^l$ will equal zero. Alternatively, if the "local temperature function" is used, as defined in Eq. (7.23), then a cooling schedule that is related to Geman and Geman's [82] cooling schedule for global temperature T could be used.

At some point in the SR process $t_s = 0\ \forall s \in S^l$, then each pixel x_s would have been sampled from an LCPDF at equilibrium. If the cooling schedule of the pixel temperature was slow enough, then the joint distribution $\Pi(\mathbf{X}^l = \mathbf{x}^l)$ should also be at equilibrium. This gives a convenient indicator for when the SR process can be terminated. Therefore, for our MR process, each grid level l undergoes constrained SR until $t_s = 0,\ \forall s \in S^l$. At which point the constrained SR at level l is terminated and recommenced at level $l-1$ with respect to $\mathbf{X}^l = \mathbf{x}^l$.

In summary, the multiplication of $(1 - t_r)$ in Eq. (7.21) represents the confidence in using site $r \in \mathcal{N}_s^l$ to estimate the LCPDF. Initially only those sites $s \in S^{l+1}$ which have had their values relaxed at the previous level $(l+1)$ are used to estimate the LCPDF, but as the SR iterations progress other sites gain in confidence. When $t_s = 0,\ \forall s \in S^l$, the SR process could be said to have reached an equilibrium state and therefore indicate it is time to move on to the next lower grid level $(l-1)$ to repeat the SR process.

7.4 Site visitation sequence

Site visitation sequence determines which sites are updated and when. The generally accepted sequence is to visit each site randomly. However there are other methods.

7.4.1 Exchange

It may be desirable to keep the overall distribution of values $\lambda \in \Lambda$ fixed. One way is to choose at random a pair of sites $s, t \in S$ and use the Metropolis algorithm to decide whether or not to exchange the values of the corresponding variables. This was employed by Cross and Jain [50] for texture synthesis.

An alternative by Green [90] was to introduce a *penalty* to the energy function $U(\mathbf{x})$, Section 3.3. Fix a vector $\mu = (\mu_0, \ldots, \mu_L)$ (the target proportions of each grey level) and let $p(\mathbf{x}) = (p_0(\mathbf{x}), \ldots, p_L(\mathbf{x}))$ denote the grey level proportions in $\mathbf{X}$: $p_j(\mathbf{x}) = \frac{1}{N} \sum_{s \in S} 1_{x_s = j}$. Then the argumented energy is:

$$U_\mu(\mathbf{x}) = U(\mathbf{x}) + \sigma^2 N |p(\mathbf{x}) - \mu|^2$$

7.4.2 Seeding

This scheme tries to mimic the physical phenomena of crystal growing. The idea is to start with a random subsample from the local joint probability distribution Π_s and map it randomly to S. This region is called the *seed* and is defined as $A \subset S$. The boundary to the seed is denoted as $\partial A = \bigcup_{t \in A} \mathcal{N}_t \setminus A$. There are then two parts to the algorithm:

1. procedure for updating the pixels $x_s,\ s \in \partial A$;
2. procedure for growing the seed A, *i.e.,* the criterion for the set A to be redefined as $A + \partial A$ and ∂A to be redefined as the boundary to this new set.

Consider a site $s \in \partial A$. The neighbouring sites $r \in \mathcal{N}_s$ may or may not have their states defined.

$$x_r = \begin{cases} \text{defined}, & r \in A \\ \text{defined/undefined}, & r \in \partial A \\ \text{undefined}, & r \in S \setminus (A \cup \partial A) \end{cases}$$

The problem is that for sites $s \in \partial A$ the LCPDF $\Pi_s(x_s|x_r, r \in \mathcal{N}_s)$ will be dependent on undefined values $x_r,\ r \in \mathcal{N}_s$. It is only when the site s is contained in A that the LCPDF is valid with respect to its neighbours. Derin and Elliott [54] recognised this problem and suggested that the boundary could be increased to $\partial A + \partial\partial A$ where $\partial\partial A$ was the boundary of $A + \partial A$. However when the set A was updated, it would still be redefined as $A + \partial A$ with the set of pixel values for $\partial\partial A$ being discarded.

Another aspect of Derin and Elliott's [54] algorithm was that they did not relax each site $s \in \partial A + \partial\partial A$ individually with respect to their LCPDFs, but updated the sites with respect to the partial joint distribution defined for all the sites $\partial A + \partial\partial A$. However this would be almost computationally impossible for our nonparametric model.

A possible solution to this seeding problem is the implementation of our own pixel temperature function. The seed can be defined by a pixel temperature $t_s = 0,\ \forall s \in A$ and $t_s = 1,\ \forall s \notin A$. The same local cooling function can be employed as Eq. (7.25), thereby slowly giving confidence to those pixels $x_s,\ s \in \partial A$. However in using this scheme directly, the influence of the confidence will extend out from the seed after each iteration. That is there is a steady progression of $t_s > 0$ for sites $s \in S$ beyond the implied boundary ∂A. This may not be exactly what is sought, and can in fact defeat the purpose of growing the field from a seed. Basically it may be preferable to control the spread of the pixel confidence.

The philosophy behind using a seed is to slowly grow it so that the internal structure of the field may be controlled. If the field is allowed to reach its own equilibrium during each stage of the seeds growth, then this will minimise phase discontinuities. This should also help the process find the global minimum of the whole field. However this all relies on a slow growth of the seed. In our experiments we found that we had to control the growth beyond that dictated by the propagation of the pixel temperature function. Therefore we only relaxed those sites within a certain radius of the seed A. This radius was only allowed to increase when all $t_s,\ s \in A$ reached a certain confidence level. However, unfortunately in our experimentation we were not able to reasonably reduce the amount of phase discontinuities for the amount of computation involved.

7.4.3 Parallelised relaxation

In our experiments, we synthesised textures on an SIMD (single instruction, multiple data) system. With SIMD systems, a single program instruction may be executed simultaneously on many relatively smaller processors that exist on a parallelised array. Therefore, although the instructions remain sequential, the data may be parallelised. For image processing applications, this is very useful as each processor in the parallelised array may be dedicated to a single pixel in the image. The SIMD system used was the massively parallel processor system, MasPar ®, with 16384 processors. For synthesising texture, a MasPar ® gives the ability to update up to 16384 pixels in one iteration. This is a significant advantage when applying SR with our nonparametric LCPDF, as the LCPDF has to be derived directly from the sample data for each iteration, which in itself is computationally intensive.

A relaxation algorithm may be parallelised if the relaxation of a single pixel is conditionally independent of other pixels undergoing simultaneous relaxation. A set of sites that may undergo simultaneous relaxation are those sites,

$$S_{i.i.d.} = \{s \in S, r \notin S_{i.i.d.} \; \forall r \in \mathcal{N}_s\} \,. \tag{7.26}$$

Basically, no two sites that are neighbours should be simultaneously relaxed. If all sites were simultaneously relaxed, Besag [19] suggests that oscillations in the site representation may result. In fact, we found simultaneous relaxation of all sites was detrimental in regard to the Ising model [121].

The set of sites that are allowed to undergo simultaneous relaxation are the same set of sites required to obtain independent and identically distributed (i.i.d.) data as specified by Besag's coding method [17]. The same coding method may therefore be used to identify each set of sites that may be simultaneously relaxed in one iteration in a parallelised relaxation algorithm.

7.5 Edge effects

When synthesising textures, there are two different problems caused by edge effects that need to be addressed in the estimation of the LCPDF. One is the problem of how to estimate the LCPDF near an edge of the synthetic image. The other, is the problem of how to incorporate sample data into the estimation process that has

been taken near an edge of the training image.

The first problem is solved by decreasing the neighbourhood associated with the pixel of concern, to just those pixels in the neighbourhood that exist in the synthetic image. Consider a site $s \in S$ that has at least one neighbouring site $r \in \mathcal{N}_s$ for which $r \notin S$, such that s is an edge site. At these sites the LCPDF cannot be estimated over the whole neighbourhood $\mathcal{N}_s$. Instead the neighbourhood for these edge pixels is reduced to be just over those pixels $r \in \mathcal{N}_s$ for which $r \in S$. This can easily be done by assigning those pixels $r \notin S$ with a pixel temperature $t_r = 1$. The LCPDF is then estimated via Eq. (7.18) with the vector $(\mathbf{z} - \mathbf{Z}_p)$ replaced by the vector $(\mathbf{z} - \mathbf{Z}_p)$ of Eq. (7.21). In Eq. (7.21) the pixel values x_s, for which $s \notin S$, are multiplied by zero and therefore can remain undefined.

The other problem is in regard to the edge effect associated with the training image $\mathbf{Y}$. In Eq. (7.21) the sample data $\mathbf{Z}_p$ is a column vector from the image $\mathbf{Y}$ representing the pixel y_p and the complete set of neighbours $y_q, q \in \mathcal{N}_p$ for a site $p \in S_y$. Previously in Eq. (7.18) the sample data $\mathbf{Z}_p$ was only obtained from those sites $p \in S_y$ which were not edge sites. However, for images $\mathbf{Y}^l$ at high grid levels, the images may be so small that all the sites $p \in S_y^l$ may be required to obtain an adequate sample distribution. Therefore in order to produce an adequate estimate of the LCPDF, the sample data from the edge sites may need to be incorporated into the estimation process. The solution to this problem is a little more complex. Instead of trying to incorporate lower dimensional sample data into a higher dimensional estimation, we keep the neighbourhood size the same for all sample data. This means that for sample data taken near the edge of the training image $\mathbf{Y}$, pixel values need to be surmised for those pixels in the neighbourhood but *outside* the image. A common approach is to set these pixel values to a common value [164], or to wrap the image as though it was on a toroidal lattice [116]. Without biasing the estimation process towards sample data collected from edge pixels, we label those pixels outside the image $y_r, r \notin S_y$ with values corresponding to their respective pixel values in $\mathbf{X}$ in the neighbourhood of the pixel for which the LCPDF is being estimated, plus some fixed offset. This makes the component of $(\mathbf{z} - \mathbf{Z}_p)$ at the position $r \notin S_y$ equal to that offset (unless $t_r = 1$). We use an offset equal to $|\Lambda|$, the number of possible grey levels.

Algorithm 1: *Nonparametric multiscale MRF texture synthesis*
Input:

$\mathbf{Y} \leftarrow$ training texture image.

$N_y \times M_y \leftarrow$ size of training image $\mathbf{Y}$

$N_x \times M_x \leftarrow$ size of synthetic image $\mathbf{X}$

$o \leftarrow$ the order of the neighbourhood system

Begin

1. Define number of grid levels as $M \leq 1 + \log_2(\min\{N_x, M_x, N_y, M_y\})$.

2. Define image $\mathbf{X}$ as being on a set of sites S as given by Eq. (7.2).

3. Define the multigrid representation of image $\mathbf{X}$ as the set of subset of sites $S^l \subseteq S$ for $0 \leq l < M$ as given by Eq. (7.3).

4. Similarly, define image $\mathbf{Y}$ as being on a set of sites, S_y Eq. (7.2), with a multigrid representation as the set of subset of sites $S_y^l \subseteq S_y$ for $0 \leq l < M$ as given by Eq. (7.3).

5. Initialise pixel temperatures $t_s = 1,\ \forall s \in S$.

6. **For** $l = M - 1$ to 0 **do**

6.1. Define neighbourhood $\mathcal{N}_s^l$ w.r.t. order o as given by Eq. (7.4).

6.2. **While** $t_s \neq 0,\ \forall s \in S^l$ **do**

6.2.1. choose a set of i.i.d.sites $S_{i.i.d.} \subset S^l$ from Eq. (7.26) for which $t_s > 0$.

6.2.2. **For all** $s \in S_{i.i.d.}$ **in parallel do**

6.2.2.1. Estimate the LCPDF for site s via Eq. (7.18), with $(\mathbf{z} - \mathbf{Z}_p)$ defined by Eq. (7.21).

6.2.2.2. Choose new x_s by sampling its LCPDF, as in the *Gibbs Sampler*, Fig. 4.5, or choose the new x_s via the *ICM algorithm*, Fig. 4.6.

6.2.2.3. Update t_s via Eq. (7.25).

6.2.3. **done**

6.3. **done**

7. **done**

End

Figure 7.4: Parallel implementation of our nonparametric multiscale MRF texture synthesis algorithm

7.6 Algorithm

An outline of our multiscale texture synthesis method for reproducing representative examples of a training texture using the nonparametric MRF model is presented in Fig. 7.4. Virtually the same algorithm is used for the strong nonparametric MRF model. The only difference is in how the LCPDF is estimated. The estimation

of the strong LCPDF is basically the same as for the nonparametric LCPDF, but instead of making the estimation with respect to a single set of neighbourhood values, separate estimates need to be compiled for each clique of the neighbourhood. The simple estimate of the strong LCPDF is then given by Eq. (6.49) as the product of the estimates obtained for each major clique.

7.6.1 Approximations made in the interest of speed

A faster version of Algorithm 1, Fig. 7.4, can be achieved by approximating the estimate of the LCPDF for $\lambda_s \in \Lambda$ by only using those data samples $\mathbf{Z}_p$ for which,

$$y_p = \lambda_s, p \in S_y^l. \tag{7.27}$$

This means that instead of obtaining the sample data $\mathbf{Z}_p$ from all $p \in S_y^l$, it is only obtained from those sites $p \in S_y^l$ for which $y_p = \lambda_s$. That is, if $y_p \neq \lambda_s \; \forall p \in S_y^l$ then $P(X_s = \lambda_s | x_r, r \in \mathcal{N}_s) = 0$. Therefore the LCPDF is then only sampled over those pixel values $\lambda_s \in \Lambda$ which occur in the image $\mathbf{Y}^l$.

A consequence of using this limited sample of $\mathbf{Z}_p$ for estimating the LCPDF $P(X_s = \lambda_s | x_r, r \in \mathcal{N}_s)$ is that now the variables used to calculate h_{opt} in Eq. (5.11) have to be updated. The first variable to be updated is σ, where σ^2 is the average marginal variance. But now σ is difficult to calculate. However, since h_{opt} is basically a guess anyway, there would probably not be any performance improvement if σ was correctly estimated. Also, it is unlikely that the correct estimate could be achieved with any great accuracy for the limited amount of sample data available. Therefore there seems to be no reason to update the original estimate of σ. Another variable to be updated is n, the number of sample data $\mathbf{Z}_p$. This is an important variable and is easy to calculate for each $P(X_s = \lambda_s | x_r, r \in \mathcal{N}_s)$. The other variable is d, the number of dimensions, which only has to be decremented by one.

In using this modification in Algorithm 1, Fig. 7.4, it was found that the texture synthesis was not hindered, but seemed to improve. This was attributed to the estimation of $P(X_s = \lambda_s | x_r, r \in \mathcal{N}_s)$ being independent of the estimations for $P(X_s \neq \lambda_s | x_r, r \in \mathcal{N}_s)$. It seems that dependence of $P(X_s = \lambda_s | x_r, r \in \mathcal{N}_s)$ on the sample data $\mathbf{Z}_p$ for which $y_p \neq \lambda_s$ might be a problem if the window parameter h_{opt} over-smoothed the distribution in the X_s dimension. Therefore it is suggested that no smoothing in the X_s dimension provides a better estimate of the LCPDF than too much smoothing.

7.7 Synthesised textures

The textures presented in Figs 7.5 and 7.6 were synthesised with the multiscale texture synthesis algorithm outlined in Fig. 7.4. In Fig. 7.5 we show the progressive realisations for the MR algorithm on a Brodatz texture at each grid level. This shows how the MR algorithm infuses the global to the local characteristics of a training texture into a synthetic texture. Fig. 7.6 demonstrates the wide range of textures – from the stochastic to the well structured – that we were able to synthesise. Best results for the structured textures were obtained with the higher order neighbourhood $\mathcal{N}^{18}$. In all cases the training texture images were of size 128×128 pixels which we used to estimate the LCPDF. The synthesised images were of size 256×256. In this way, we confirmed that the characteristics of the training texture had indeed been captured by the model. More results are present in Appendix B.

The synthesis of the textures was carried out with the images represented by 256 grey levels and neighbourhood systems of order $o = 8$ and $o = 18$. This meant that the LCPDF used to capture the characteristics of the textured training image, was very sparsely populated with sample data. Even so, the results from the texture synthesis algorithm show that the characteristics may be replicated over a larger area and with relatively little distortion by phase discontinuity. Therefore, the multiscale texture synthesis algorithm, Fig. 7.4, is very robust.

In Appendix B.1 we investigate the effect of varying the neighbourhood size with which to model the training texture. Figs. B.1–B.166 demonstrate how well the non-parametric MRF model is able to synthesise various texture types. The lack of phase discontinuity in the larger synthesised images suggest that the model is capable of capturing all the texture characteristics of a training texture. These results show that, although the multiscale texture synthesis algorithm can produce visually similar realisations of the training texture, there is still the possibility of the model being overtrained. This is evident when modelling with the larger neighbourhoods, as a degree of replication is present in the synthesised textures. However, this does give an indication of the upper bound to the neighbourhood size required for modelling a specific texture.

In Appendix B.1 we also explore the significance, if any, of choosing either the Gibbs sampling scheme or the ICM sampling scheme. From the results, Figs. B.1–B.166, it can be ascertained that on the whole the ICM sampling scheme performs

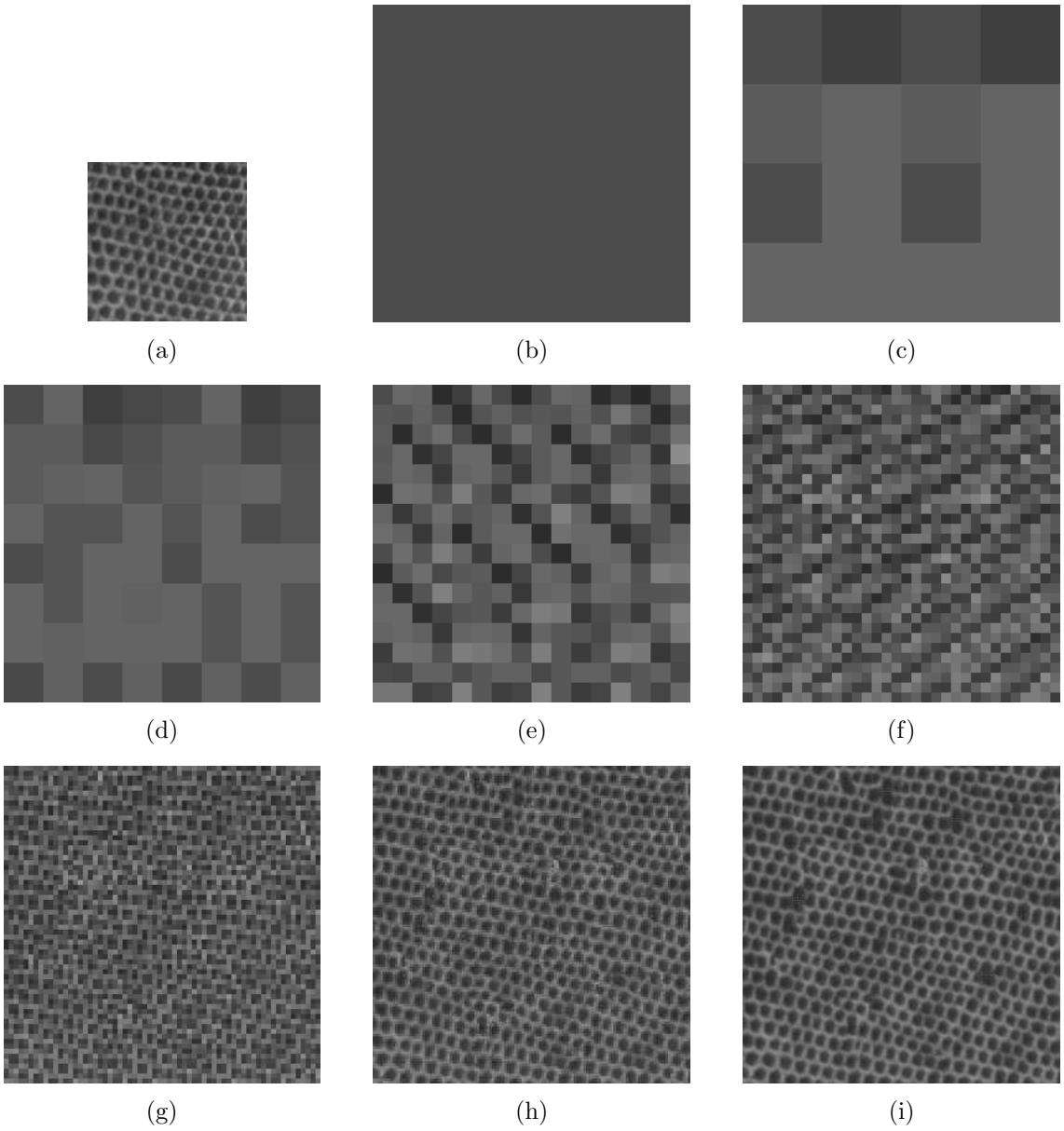
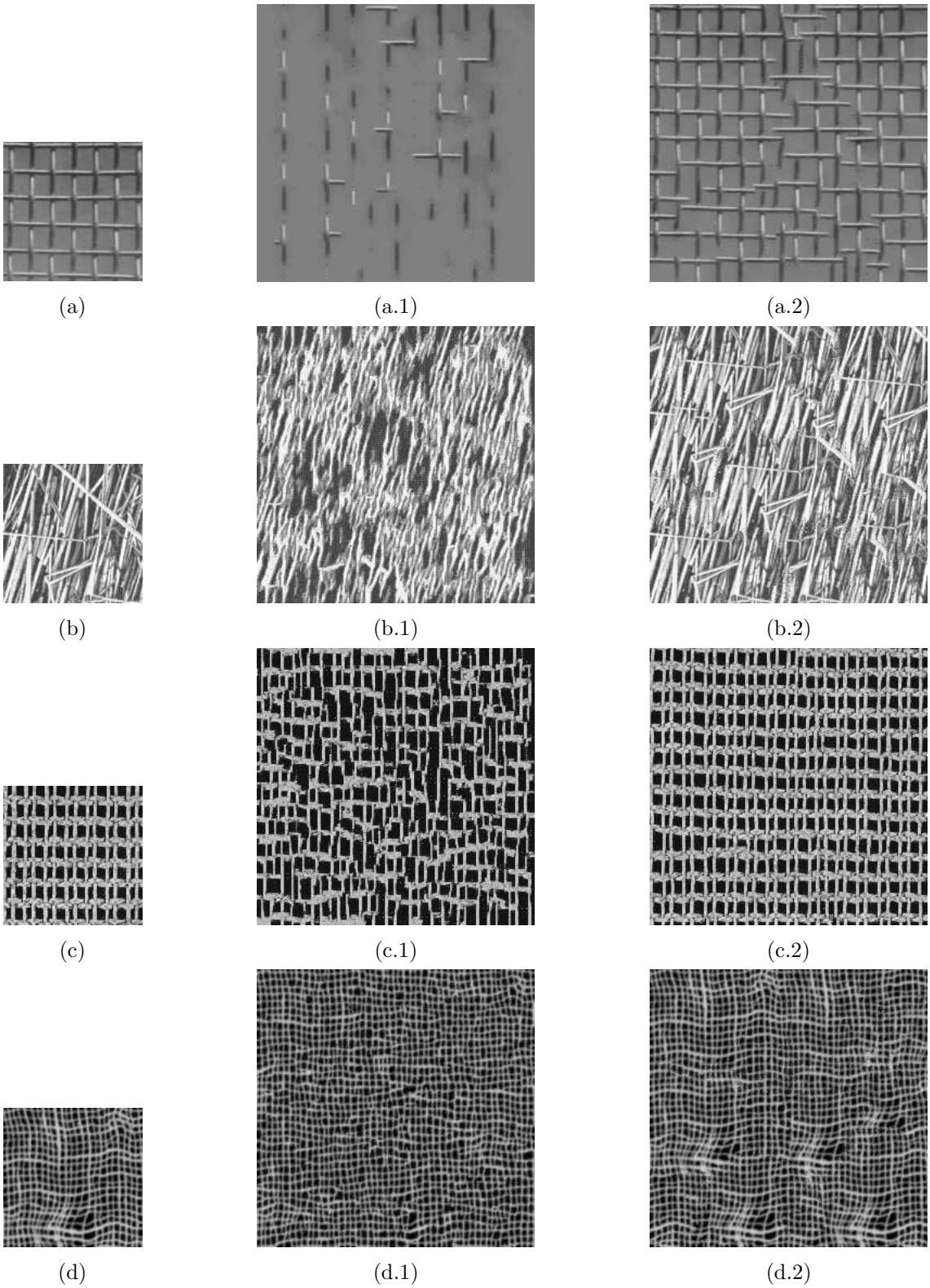


Figure 7.5: Multiscale texture synthesis of Brodatz D22 (reptile skin) with neighbourhood order $o = 8$. (a) Original textured image; (b) Level 7; (c) Level 6; (d) Level 5; (e) Level 4; (f) Level 3; (g) Level 2; (h) Level 1; (i) Level 0.

marginally better than the Gibbs sampling scheme, especially at small neighbourhood sizes. What can be deciphered from such an observational result is that the LCPDF is noisy. While the ICM algorithm samples from the mode of the distribution, the Gibbs algorithm samples from the whole distribution. Therefore, although the modes of our LCPDF seem to be fairly well estimated, the rest of the distribution does not fit the texture as well. This is to be expected when there is not



(a) (a.1) (a.2)

(b) (b.1) (b.2)

(c) (c.1) (c.2)

(d) (d.1) (d.2)

Figure 7.6: Brodatz textures: (a) D1 - aluminium wire mesh; (b) D15 - straw; (c) D20 - magnified French canvas; (d) D103 - loose burlap; (?.1) synthesised textures - using neighbourhood order $o = 8$; (?.2) synthesised textures - using neighbourhood order $o = 18$.

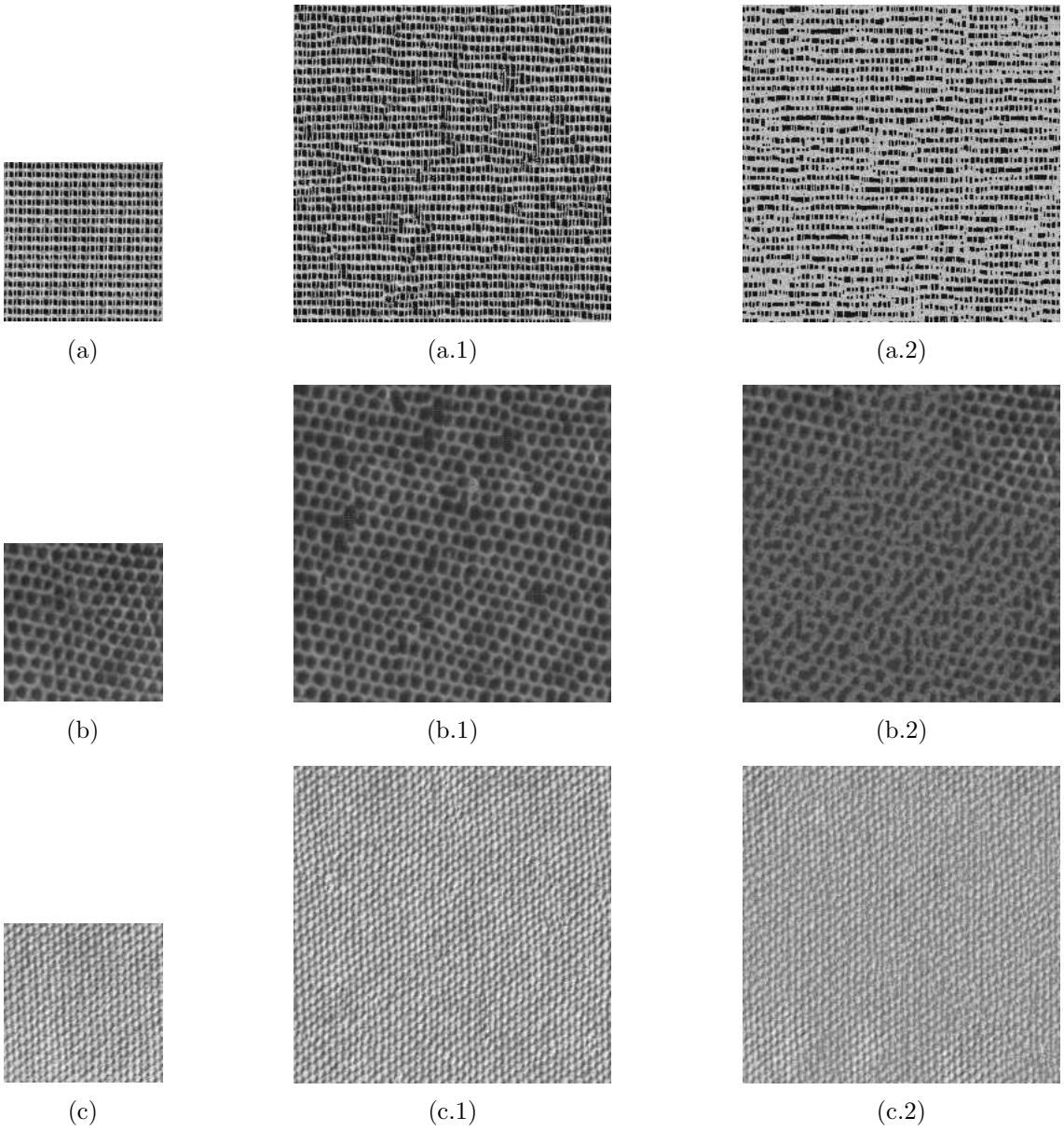


Figure 7.7: Brodatz textures: (a) D21 - French canvas; (b) D22 - Reptile skin; (c) D77 - Cotton canvas; (?.1) synthesised textures - Nonparametric MRF with neighbourhood order $o = 8$; (?.2) synthesised textures - strong nonparametric MRF with neighbourhood order $o = 8$ and only 3rd order cliques.

enough data to properly estimate an entire distribution. The reason why the Gibbs algorithm performs better with increasing neighbourhood size, is therefore due to the LCPDF becoming more modish.

One facet that comes out from the experimental comparison between the Gibbs and ICM sampling schemes is an idea of how the LCPDF could be most effectively

used for classification. Obviously, if it are the modes of the LCPDF that are the most reliable, then it are the modes which should form the basis of the classification algorithm. This concept will be further explored in Chapter 8.

In our multiscale texture synthesis algorithm, as outlined in Fig. 7.4, one of the variables is the multigrid height. In the presented algorithm we have set the variable to the maximum height possible. This however, is a default setting and is not a requirement. In Appendix B.2 we explore the possibility of synthesising various training textures with different settings of the multigrid height.

The setting of the multigrid height is equivalent to limiting the extent to which neighbouring pixels effect the LCPDF. The higher up the multigrid the height is set, the further away the pixels can be to still have a direct influence on the LCPDF. Under the MRF premise, we would like to show that textures can be represented by MRF models that have limited spatial extent, whereby all long rang correlations are derived from field interactions rather than by direct neighbourhood control. Therefore the lower the multigrid height can be set the better. The maximum multigrid height is also another parameter that can be set in the open-ended classification algorithm. In the interest of obtaining an "ideal" model for open-ended classification, we want to model just enough information to represent the texture. Therefore, this is another reason why we would want to seek the minimum multigrid height required to model a texture. The synthesis results, Figs. B.167–B.175, indicate that it is possible to limit the multigrid to just a couple of levels.

The results obtained for strong MRF model, Figs. 7.7 (a.2) (b.2) and (c.2), suggest that these textures may be successfully modelled with just third order statistics. Although the plain nonparametric MRF model clearly outperforms the strong MRF model in synthesising texture, as seen in Fig. 7.7 and Appendix B.3, the results do show that the strong MRF model is capable of capturing the characteristics of a training texture. It is the strong MRF model that will probably be the more successful for open-ended texture classification. This is because it uses lower order statistics, thereby increasing its entropy while retaining the unique characteristics of the texture [211]. Another advantage of the strong MRF model is that we can search for the maximum statistical order required to model a texture. This gives us an upper bound on the order of the significant statistical information contained within the texture. In Appendix B.3 we also look at reducing the number of cliques used in the strong MRF model through a process of entropy discrimination.

To perform open-ended classification, the ideal texture model should capture

enough information to be able to synthesise representative examples of the training texture. However the texture model should not be overtrained, otherwise the model will not be representative of other like texture in the texture class. To obtain the correct model, the minimum neighbourhood size should be chosen that allows adequate synthesise of the training texture. The model can then be further generalised by reducing the statistical order of the model through correct choice of cliques in the strong MRF model. Again the cliques should be chosen so as to allow adequate reproduction of the training texture. It is hoped that in a practical application neighbourhood size and clique sets could be predefined. That is, texture classes in a certain application may all be modelled from the one model structure. This quite possibly could be the case with terrain recognition in SAR images.

In trying to determine the optimal neighbourhood size for a training texture, an unusual observation was made. The minimum neighbourhood size required to synthesise a subjectively similar texture to a training texture did not seem to be dependent on the original resolution of the training texture. Rather, it seemed to be more dependent on the relative periodic nature of detail in the training texture. Textures that had fine detail periodically spaced at relatively large distances seemed to require relatively larger neighbourhoods (*e.g.,* Fig. 7.6(a)), whereas textures with large detail periodically spaced at relatively small distances (*e.g.,* Fig. 7.5(a)) required relatively smaller neighbourhoods. It was evident in Fig. 7.5 that the periodic nature of the detail became more pronounced at higher grid levels (lower resolutions), but the detail itself started to disappear. In fact there was an optimal resolution for the texture where the detail was evident as well as the periodic nature of that detail. It was this resolution that strongly dictated the optimal neighbourhood size (a neighbourhood size that was independent of the original resolution of the texture).

Chapter 8

Classifying Texture

This chapter outlines the common method for supervised classification. Because of the problems we see for supervised texture classification in the area of terrain recognition for SAR images, we introduce our own method of terrain recognition that uses an open-ended texture classifier. While supervised classification is based on a closed N class classifier that requires all textures class to be predefined in a set of training textures, the open-ended classifier requires just one training texture to be used to distinguish a texture class from all other unknown textures. From the results of this chapter we have published another three papers demonstrating the use of the open-ended texture classifier and in particular the advantage of the added functionality of the strong nonparametric MRF model [104, 155, 158].

8.1 Introduction

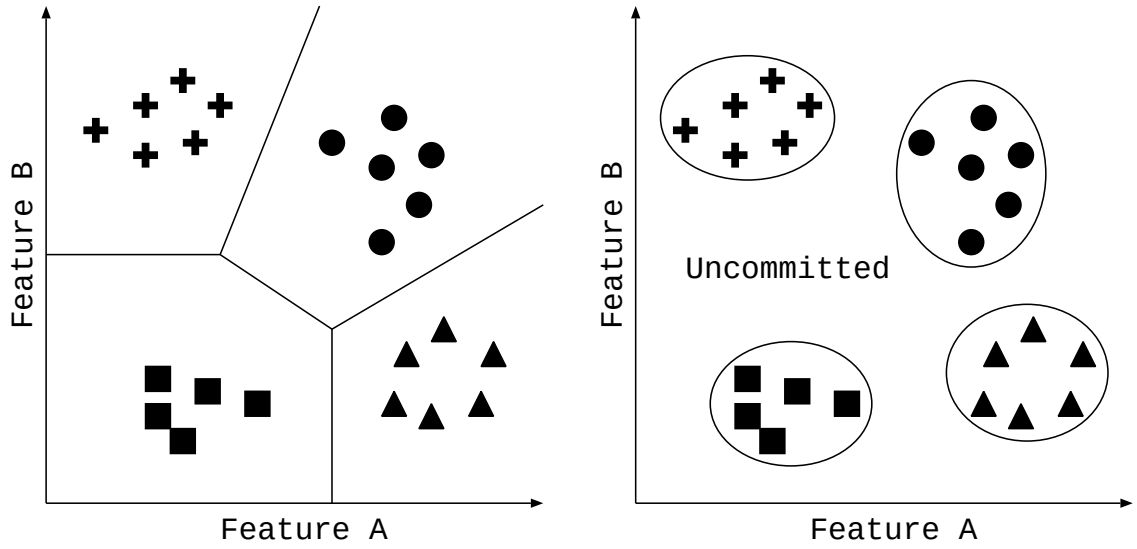
MRF models have mainly been used for *supervised* segmentation and/or classification, where the number and/or types of texture classes present are a prior known [34, 38, 40, 54, 81]. Under the assumption that all texture classes present in the image are contained within the training data, segmentation and classification is performed by finding the Maximum *a posterior* (MAP) estimate [81] or the Maximum Posterior Marginal (MPM) estimate [38]. In each case, whether the MAP or the MPM estimate is used, each pixel in the image is assigned a label, identifying it with a training texture class. The MAP estimate assigns these labels so as to limit the error (or maximise the likelihood) over the whole image (*i.e.,* it uses the associated joint distribution Π). The MPM estimate limits the error per pixel (*i.e.,* it uses

the LCPDF). Both use Markov models as texture measures to define a likelihood that a segment of texture is similar to a training texture. These texture measures need not be complete, they just need to be able to discriminate one training texture class from another. Therefore supervised classification is only applicable for discriminating the different training texture classes from an image. This is a hindrance for classifying Earth terrain types from SAR images, as there are a myriad of different terrain types (*i.e.,* textures), too many to build a library of training textures so that supervised classification may be performed on an arbitrary SAR image.

One approach around the problem stated above, is to build the texture models directly from the image to be segmented [106, 117, 138, 157]. The basic concept is to make an initial segmentation using a prior set of texture models, then to improve these models by training them on their own segmented areas. The image is repeatedly segmented, while successively improving the models after each iteration. This process continues until a steady state is reached. This process is sometimes referred to as *unsupervised classification* [117, 168], but a better term would be *unsupervised segmentation*. This is because only arbitrary labels are given to the homogeneous textured regions in the image scene.

We present a new approach to this classification problem, one that uses our multiscale nonparametric MRF model. The proposed solution is to just model one training texture, but well enough so that the captured texture characteristics can be used in an open-ended classifier. With a conventional N class classifier [60] the feature space is subdivided into N distinct domains or classes. The N class classifier uses the class boundaries to classify data into known classes. The advantage of using an open-ended classifier is that it has a default domain called "unknown", or "uncommitted," as illustrated in Fig. 8.1. The open-ended classifier uses an ideal model to determine the probability distribution for each class over the whole feature space. New data is then classified as being from the class with the highest likelihood according to the data specific features. However if this likelihood is low, then the data can be classified as being from an "unknown" class. The open-ended classifier is therefore "open" to data outside the known training classes.

The open-ended classifier is based on the assumption that the feature space is complete, in that any new training data of a separate class would occupy its own distinct region in this feature space. The classifier is also based on the assumption that the training data for each class can be represented by an ideal model, so that not only does each model fully characterise the training data, but the model may be used



(a) Conventional N class classifier (b) Open-ended classifier

Figure 8.1: Opened and closed classifiers: (a) The conventional N class classifier is a closed classifier since further input of training data would require a complete reconfiguration of the designated classes. (b) Open-ended classifier is open-ended because further input of training data would not destroy the overall configuration of the designated classes, but would instead enhance individual classes.

to give a precise likelihood that any new data is of the same class. Classification is not based on constructing class boundaries to describe the inter-relationships between classes, instead classification is based on an ideal model defining the intra-relationships of the training data for a class.

We have shown that we can model just about any arbitrary homogeneous texture via our nonparametric MRF model, such that the most likely texture synthesised from the model is one that is highly representative to the training texture, see Appendix B.1. Therefore we maintain that these models are ideal and the feature space on which they are based is complete. Given any set of these ideal models it is then possible to use the open-ended classifier for texture recognition.

Unlike a conventional N class classifier, an open-ended classifier does not possess defined boundaries between classes. Instead an open-ended classifier defines the probability of an image segment being of a modelled texture. This means instead of an image segment being labelled as a particular texture, it is given a probability for being that texture. If the open-ended classifier contains a set of ideal texture models, then the image segment is given a set of probabilities, one for each training

texture. Greenspan *et al.* [91] also used a probability map over an image to show the degree of similarity of texture to a training texture, but that was done through the use of a neural net performing supervised classification.

The open-ended classifier permits segmentation and classification of images with an unknown number and variety of texture classes. However in order to construct such a classifier we need to find some meaningful way of extracting the information required from the ideal models to perform the necessary classification.

8.2 Bayesian paradigm

First we look at a standard approach for classifying texture from an image. Geman and Graffigne [81] use the Bayesian paradigm to segment and classify texture via supervised classification. In this approach, the texture classification is based on two models; one modelling the interaction between the pixel labels, and the other modelling the pixels themselves. Following the notation of Geman and Graffigne [81], the image to be segmented and classified is denoted as $\mathbf{X} = \mathbf{x}$ and resides on the lattice $S = \{s_1, s_2, \ldots, s_N\}$. The label image is denoted as $\mathbf{X}^L = \mathbf{x}^L$ and resides on the same lattice as X. For each pixel variable there is a corresponding label variable. The classification is then made on the basis of maximising the combined probability via the *Bayes' rule* whereby,

$$P(\mathbf{x}^L|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{x}^L)P(\mathbf{x}^L)}{P(\mathbf{x})}, \tag{8.1}$$

In Geman and Graffigne's [81] interpretation of the label image, $\mathbf{X}^L$, each variable $\{X_s^L, s \in S\}$ may be assigned a discrete label $x_s^L \in \Lambda^L$, where Λ^L represents the discrete set of possible texture labels. In our case we wish to label a particular pixel with the probability that it is of the same texture class as the pre-modelled training texture. Therefore Λ^L no longer represents a discrete set of labels, but instead is a continuous set of real numbers bounded by zero and one.

The aim of the segmentation classification procedure is to find the image $\mathbf{x}^L$ that maximises the probability $P(\mathbf{x}^L|\mathbf{x})$. From Eq. (8.1), this probability is maximised by maximising both $P(\mathbf{x}|\mathbf{x}^L)$ and $P(\mathbf{x}^L)$. The probability $P(\mathbf{x})$ represents the probability of obtaining $\mathbf{x}$. As we are only concerned with the one $\mathbf{x}$, the probability $P(\mathbf{x})$ may be ignored. The probability $P(\mathbf{x}^L)$ includes factors relating to how the

pixels of the same (or in our case, similar) labels cluster together. One would expect that a pixel will have the same or similar labels as its neighbours. It is usual to define $P(\mathbf{x}^L)$ with respect to the Ising model, Section 4.2.1,

$$P(\mathbf{x}^L) = \frac{1}{Z^L} \exp\left\{ \beta \sum_{C=\{s,r\}\in\mathcal{C}} 1_{x_s^L = x_r^L} \right\}, \tag{8.2}$$

where, the greater the β, the more likely the neighbouring pixels will be of the same label. In our model, we will ignore this clustering effect and use $\beta = 0$. For each pixel, we will individually determine the probability that it is of the pre-modelled training texture. In a more sophisticated version of our classification algorithm, one would use the reasonable assumption that pixels close to each other would exhibit similar probabilities of classification. It may also be prudent to incorporate boundary detection as part of a constrained optimisation of the probability map as discussed in [80]. However these types of improvements to the general algorithm are more application driven, and so therefore we will limit ourselves to outlining the simple version of our classification algorithm, and discussing its own particular merits.

This leaves just the probability $P(\mathbf{x}|\mathbf{x}^L)$ from which to derive a classification algorithm. If the whole image is of the one texture and $\mathbf{X}$ is an MRF, then,

$$P(\mathbf{x}|\mathbf{x}^L) = \Pi(\mathbf{x}). \tag{8.3}$$

That is $P(\mathbf{x}|\mathbf{x}^L)$ equals the joint probability $\Pi(\mathbf{x})$ as uniquely determined by the LCPDFs $P(x_s|x_r, r \in \mathcal{N}_s)$ $s \in S$. For cases when $\mathbf{x}$ is not all of the one texture, Geman and Graffigne [81] classified small areas of the image assuming they were homogeneous. The classification was made on the basis that the product of the neighbourhood probabilities over the area of concern resembled the joint probability for that area, *i.e.,* ,

$$\Pi(x_r, r \in W_s) \simeq \prod_{r\in W_s} P_r(x_r, x_t, t \in \mathcal{N}_r), \tag{8.4}$$

where W_s is the window of sites, centred at s, which are to be used for the classification of x_s. Note that the probability $P_r(x_r, x_t, t \in \mathcal{N}_r)$ in Eq. (8.4) is not derived from the LCPDF, but rather the local probability density function (LPDF). The LPDF may be calculated from the nonparametric approximation of the LCPDF, Eq. (5.12), but with the normalising function omitted.

It is common to use a large window W_s for segmentation and classification. Geman and Graffigne [81] used a 5×5 window for an MRF modelled with respect to the nearest neighbour neighbourhood $\mathcal{N}^1$. Chellappa *et al.* [38] also used a large window from which to determine the classification of a pixel. An intuitive explanation as to why a window for segmentation/classification should be large, is found in the representation of Π by the LPDF. It is the joint distribution $\Pi(\mathbf{x})$ that will determine the probability that $\mathbf{x}$ is of a pre-modelled training texture. A window of one pixel will realise only one sample of the LCPDF by which to test the distribution. Therefore a large window is used to obtain a reasonable sample of LPDF realisations.

In order for the Bayesian classifier of Eq. (8.4) to be successful, all probabilities derived from the LPDF need to be well defined. This means that the estimate of the LPDF needs to be representative of the texture class over its whole domain, otherwise errors will creep into the classification algorithm, and under Eq. (8.4) these will have a multiplicative effect. This is exactly what we found in our preliminary experimentation. When dealing with large dimensional LPDFs, we were not able to accurately estimate the LPDF well enough to avoid contaminating the classification algorithm with a debilitating amount of noise in Eq. (8.4). Even alternative choices of h_{opt} were not able to rectify the problem.

8.3 Open-ended classification

We found the probability $\Pi(x_r, r \in W_s)$ as defined by Eq. (8.4) to be unwieldy. This was because our nonparametric LPDF tended to give low probabilities for the neighbourhood configurations in the classification window, which resulted in $\Pi(x_r, r \in W_s)$ being too susceptible to minor fluctuations from inaccurate estimates of the LPDF. Instead of trying to combat the inherent inaccuracies of the estimate of an LPDF over a large neighbourhood, we take an alternative approach of incorporating the uncertainty of the estimate into the classification scheme. We know we have captured the complete characteristics of the texture in the LPDF, we just need a better way of extracting that information out of the LPDF to perform texture classification. In our approach, we take the set of probabilities, as defined by the LPDF for the window W_s, and compared them directly to a set of probabilities obtained from the training texture. We use a hypothesis test to determine whether these two sets of probabilities come from the same population. This therefore side steps the

issue of trying to find a better estimate of the LPDF and partially indemnifies the classification procedure against minor fluctuations in the LPDF.

8.3.1 Probability measurement

The probability $P_r(x_r, x_t, t \in \mathcal{N}_r)$ is calculated from Eq. (5.12) with the normalising function omitted.

$$P_r(x_r, x_t, t \in \mathcal{N}_r) = \frac{1}{nh^d(2\pi)^{d/2}} \sum_{\substack{p \in S_y, \\ \mathcal{N}_p \subset S_y}} \exp\left[-\frac{1}{2h_{opt}^2}(\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}}(\mathbf{z} - \mathbf{Z}_p)\right]. \tag{8.5}$$

As defined previously, $\mathbf{z} = \mathrm{Col}[x_r, x_t, t \in \mathcal{N}_r]$ and $\mathbf{Z}_p$ are similar samples taken from the training image $\mathbf{Y}$ defined on the lattice S_y. The constant $\frac{1}{nh^d(2\pi)^{d/2}}$ is applied so that $P_r(x_r, x_t, t \in \mathcal{N}_r)$ remains a true probability, Section 5.3.

For the strong nonparametric MRF, we use the simple estimate of Eq. (6.49) giving,

$$P_r(x_r, x_t, t \in \mathcal{N}_r) = \prod_{\substack{C \in \mathcal{C}_r, \\ C \not\subset C' \in \mathcal{C}_r}} P_r(x_r, x_t, t \in C), \tag{8.6}$$

where each $P_r(x_r, x_t, t \in C)$ is calculated in a similar fashion to $P_r(x_r, x_t, t \in \mathcal{N}_r)$ in Eq. (8.5), except the calculation is based on the clique C rather than the neighbourhood $\mathcal{N}_r$, Section 6.6.

The samples of the LPDF, taken from the window $W_s \subset S$, are the set of probabilities $\{P_r(x_r, x_t, t \in \mathcal{N}_r), r \in W_s\}$. Instead of multiplying these probabilities together, as in Eq. (8.4), we will compare them directly to the set of probabilities obtainable from the training image $\mathbf{Y}$. For every site $q \in S_y$, $\mathcal{N}_q \subset S_y$ in the training image, it is possible to obtain a vector,

$$\mathbf{z} = \mathrm{Col}[y_q, y_t, t \in \mathcal{N}_q], \qquad q \in S_y,\ \mathcal{N}_q \subset S_y, \tag{8.7}$$

for which $P_q(y_q, y_t, t \in \mathcal{N}_q)$ may be calculated. This may be done via Eq. (8.5) with the sample vectors $\mathbf{Z}_p$ also coming from the training image $\mathbf{Y}$. However, this probability estimate would be biased since the sample vector $\mathbf{Z}_p = \mathbf{z}$ would also be included in the estimation. This bias is removed by excluding the site $p = q$ from the calculation of $P_q(y_q, y_t, t \in \mathcal{N}_q)$. The set of probabilities $\{P_q(y_q, y_t, t \in \mathcal{N}_q), q \in S_y, \mathcal{N}_q \subset S_y\}$ from the training image $\mathbf{Y}$, is therefore calculated using the modified

formula of Eq. (8.5), giving,

$$P_q(x_q, x_t, t \in \mathcal{N}_q) = \frac{1}{nh^d(2\pi)^{d/2}} \sum_{\substack{p \in S_y - q, \\ \mathcal{N}_p \subset S_y}} \exp\left[-\frac{1}{2h_{opt}^2}(\mathbf{z} - \mathbf{Z}_p)^{\mathrm{T}}(\mathbf{z} - \mathbf{Z}_p)\right]. \quad (8.8)$$

Given the set of probabilities $\{P_r(x_r, x_t, t \in \mathcal{N}_r), r \in W_s\}$ from the window to be classified, and the set of probabilities $\{P_q(y_q, y_t, t \in \mathcal{N}_q), q \in S_y, \mathcal{N}_q \subset S_y\}$ from the training image, we are now able to determine the recognition probability. The null hypothesis is that the distribution of probabilities from the window is the same as the distribution from the training image. We use the nonparametric Kruskal-Wallis test [128] to test this hypothesis.

The Kruskal-Wallis test is the nonparametric version of the F test [128]. We use the Kruskal-Wallis test to make inferences about treatment populations, accepting or rejecting the null hypothesis that the populations come from the same distribution, primarily by comparing the means. The Kruskal-Wallis statistic K is calculated in terms of the *ranks* of the observations rather than their nominal values.

Given c populations, each with n_j observations, combine them to form one set of n_T observations, where $n_T = \sum_{j=1}^{c} n_j$. Then rank these observations from smallest to largest value. The smallest valued observation is given a rank of 1, and the largest a rank of n_T. If more than one observation has the same value, then they are ranked arbitrarily and then given the same rank equal to the mean of their ranks. With this set of ranked observations, define T_j as the sum of the ranks for each population. Then the Kruskal-Wallis statistic K is expressed as,

$$K = \frac{12}{n_T(n_T + 1)} \sum_{j=1}^{c} \left(\frac{T_j^2}{n_j}\right) - 3(n_T + 1), \quad (8.9)$$

The sampling distribution of K is approximately chi-squared with $c - 1$ degrees of freedom. For two populations, the Kruskal-Wallis statistic K is equivalent to the square of the normalised Wilcoxon statistic [97, 128]. Given K, the accepted practice is to accept or reject the null hypothesis if K is greater than a particular *confidence* level α. Our approach is to instead calculate the *confidence* associated with accepting the null hypothesis and use that as a goodness-of-fit.

As we are just testing two populations – one a set of probabilities from the window to be classified, the other a set of probabilities from the training image

– then K is chi-squared distributed with one degree of freedom. The confidence associated with accepting the null hypothesis is equal to the area under the chi-squared distribution for values greater than K. This confidence, for a particular window W_s, will be denoted as P_{W_s} and refer to the probability,

$$P_{W_s} = P(k \geq K), \tag{8.10}$$

where K is calculated from Eq. (8.9) and k is chi-squared distributed with one degree of freedom. It is this confidence (or goodness-of-fit), P_{W_s}, with which we plot our probability map, such that,

$$X_s^L = P_{W_s}. \tag{8.11}$$

The convenience of this approach is the fact that Numerical Recipes [167] provides an efficient algorithm for calculating the area under a chi-squared distribution.

8.3.2 Multiscale probability measurement

There are various approaches for defining a multiscale probability measurement for classification. Krishnamachari and Chellappa [125] defined separate probability measurements for each grid level. At the top most grid level (*i.e.,* lowest resolution) they classified the image by using the ICM algorithm with respect to the corresponding probability measurement. This classified image was then propagated down a grid level where it was used as an initial classification for the implementation of again the ICM algorithm with respect to the next probability measurement. This was continued down the grid levels to the bottom grid level or highest resolution.

Bouman and Liu [26] defined a likelihood function that was dependent on a proposed classification of the lower grid level or higher resolution. They then used this likelihood function to obtain the maximum likelihood estimate at the top most grid level. This estimate or classified image was then propagated down to the next grid level where the ICM algorithm was used to obtain the maximum likelihood estimate at that level. This was continued down the grid levels to the bottom grid level or highest resolution. This algorithm was later improved upon by Bouman and Shapiro [27] whereby the likelihood function was updated with respect to the current classification.

Then there is the approach by De Bonet and Viola [24], who defined a probability measurement that transcended across all scales. This allowed them to classify the

pixels with respect to all multiscale features as one joint probability measurement.

Unfortunately none of these approaches are appropriate for our probability measurement. This is because we use the Kruskal-Wallis probability, Eq. (8.10), to classify a pixel in a test image. This probability does not require an iteration process like the ICM algorithm, nor is it dependent on the classification of pixels at other grid levels. It would in fact be very difficult to incorporate dependence between grid levels as the Kruskal-Wallis probability, Eq. (8.10), is a statistic comparing whether sets of samples are from the one population. A set of samples from each grid level would constitute separate and distinct populations. Therefore it is more convenient to calculate the Kruskal-Wallis probability for each grid level as a separate image and to regard these probabilities as unrelated.

The classification of the site $s \in S$ with the combined Kruskal-Wallis probabilities may now be explicitly expressed as,

$$X^L_{s=(i,j)} = \prod_{l \geq 0} P^l_{W_r}, \qquad r = \left(\frac{i}{2^l}, \frac{j}{2^l} \right) \in S^l, \tag{8.12}$$

where the probability map is given as $\mathbf{X}^L = \{X^L_s, s \in S\}$.

8.4 Boundaries and edges

Classification of a window that overlaps an edge of the image or a window that overlaps a boundary between two or more textures is an open problem. However there are segmentation methods for finding texture boundaries [79, 80, 204]. Here we present our own ad hoc approach to solving this boundary and edge problem.

First lets consider the classification of a site $s \in S$ whose window W_s centred at s is not wholly contained in S. That is, the window W_s overlaps the edge of the image. Previously we have designated the centre site of the window as the one to be labelled with the classification probability obtained for the window. However, this was only chosen for the reason that it was the most logical choice. In fact the Kruskal-Wallis probability, P_{W_s}, obtained for the window, W_s, is the probability that the patch covered by the window, W_s, is of the same texture as in the training texture. Therefore we may label any site within the window, W_s, with the Kruskal-Wallis probability P_{W_s}. For solving the edge problem, this means we may use an alternative window W_r that is wholly contained within S for which $s \in W_r \subset S$. An

example of an alternate window, W_r, for an edge site s is shown in Fig. 8.2.

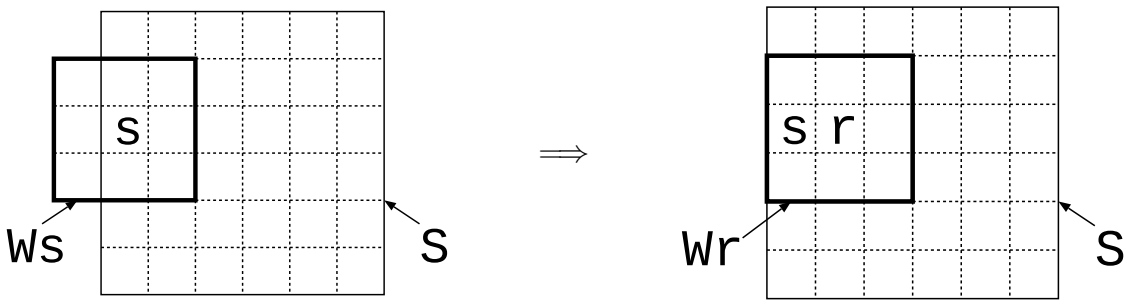


Figure 8.2: The modification of the window position for an edge site

The second problem, is the problem of a window overlapping a boundary between two or more textures. This is not so much a problem when the textures involved are not of the training texture, as a low Kruskal-Wallis probability will be obtained anyway. However, if the site being classified is on the boundary of a similar texture to the training texture, and its classification window encompasses other textures, then the Kruskal-Wallis probability will again be low. This means that the probability map will tend to show "boundary erosion" around textures that are deemed similar to the training image. To overcome this problem with the presentation of the probability map, we introduce our own ad hoc method for reducing the erosion around textural boundaries.

As with the edge problem, when a window overlapped the edge of a image, we may solve the boundary problem by choosing an alternate window W_r. Given a site s on the boundary of texture, we wish to choose a window W_r for s that is completely encompassed by that texture. Thereby maximising the probability that the boundary site will be classified along with the other sites of the texture. An example of such an alternate window, W_r, is shown in Fig. 8.3.

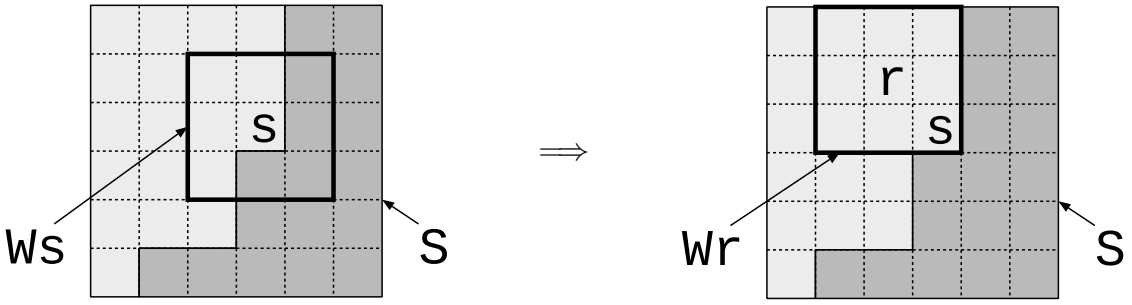


Figure 8.3: The modification of the window position for a boundary site

Even though the best alternate window is dependent on the unknown textural boundaries, it is still possible to choose a window, W_r, that gives the optimal probability of classification for the site $s \in S$, as,

$$W_{\text{optimal}} = \arg \max_{r \in W_s} P_{W_r} \tag{8.13}$$

This is tantamount to labelling X_s^L with,

$$X_s^L = \max_{r \in W_s} P_{W_r} \tag{8.14}$$

From experiments, this ad hoc approach produced probability maps with reduced boundary erosion, giving sharp clean borders between similar and dissimilar textures with respect to a training texture.

8.5 Algorithm

In the open-ended texture classification algorithm, Fig. 8.4, the multiscale approach used is not same as the Gidas [86] approach used in the multiscale synthesis algorithm, Fig. 7.4. Basically we do not constrain the classification at one grid level with respect to the classification from the higher grid level. Instead we label the site s with the combined Kruskal-Wallis probability under the assumption that the probabilities obtained at each grid level are independent.

Since the Gidas [86] constraint is no longer used, there is now no advantage in using *local decimation* for the multigrid representation. In fact there is a distinct disadvantage of using local decimation. As local decimation subsamples the image at grid l by four, this means that are four possibilities for the image at grid $l+1$. To be thorough, each image should be used. On the other hand, if *local averaging* is used, then is one distinct multigrid representation.

As with the synthesis algorithm, given an $N \times M$ image, $\mathbf{X}$, we will define the rectangular lattice on which to represent the image, $\mathbf{X}$, at grid level $l = 0$ as,

$$S = \{s = (i, j) : 0 \leq i < N, 0 \leq j < M\}. \tag{8.15}$$

However, unlike the multigrid representation of the image, $\mathbf{X}$, in the synthesis algorithm, we will base the multigrid representation on *local averaging* rather than

Algorithm 2: *Nonparametric multiscale MRF open-ended texture classification*
Input:

$\mathbf{X} \leftarrow$ the input image

$\mathbf{Y} \leftarrow$ the training image

$N_x \times M_x \leftarrow$ size of input image $\mathbf{X}$

$N_y \times M_y \leftarrow$ size of training image $\mathbf{Y}$

$o \leftarrow$ the order of the neighbourhood system

$W_s \leftarrow$ classification window

Begin

1. Define number of grid levels as $M \leq 1 + \log_2(\min\{N_x, M_x, N_y, M_y\})$.

2. Define probability map $\mathbf{X}^L$ and image $\mathbf{X}$ as being on a set of sites S as given by Eq. (8.15).

3. Define the multigrid representation of the image, $\mathbf{X}$, as the set of images, $\mathbf{X}^l$ Eq. (8.17), defined on respective lattices, S^l Eq. (8.16), for each grid level $0 \leq l < M$.

4. Similarly, define image $\mathbf{Y}$ as being on a set of sites, S_y Eq. (8.15), with a multigrid representation as the set of images, $\mathbf{Y}^l$ Eq. (8.17), defined on respective lattices, S_y^l Eq. (8.16), for each grid level $0 \leq l < M$.

5. Define neighbourhood $\mathcal{N}$ w.r.t. order o via Eq. (3.9).

6. Initialise probability map to $\mathbf{X}^L = \{X_s^L = 1, s \in S\}$.

7. **For** $l = M - 1$ to 0 **do**

7.1. Obtain the set of LPDF samples $\{P_q^l(y_q^l, y_t^l, t \in \mathcal{N}_q), q \in S_y^l, \mathcal{N}_q \subset S_y^l\}$ from the training image $\mathbf{Y}^l$ via Eq. (8.8).

7.2. **For all** $s \in S^l$ **in parallel do**

7.2.1. Calculate $P_s^l(x_s^l, x_r^l, r \in \mathcal{N}_s)$ via Eq. (8.5) or Eq. (8.18).

7.2.2. Calculate the Kruskal-Wallis Probability $P_{W_s}^l$ from Eq. (8.10) via Eq. (8.9).

7.2.3. Correct edge and boundary effects by equating $P_{W_s}^l = \max_{r \in W_s} P_{W_r}^l,\ W_r \subset S^l$.

7.2.4. Label all $X_r^L = x_r^L \times P_{W_s}^l$, for which $r = (p, q) \in S,\ s = (p/2^l, q/2^l)$.

7.3. **done**

8. **done**

End

Figure 8.4: Parallel implementation of our nonparametric multiscale MRF open-ended texture classification algorithm.

decimation for the open-ended texture classification algorithm. The multigrid representation of the image, $\mathbf{X}$, is then denoted as the set of images, $\mathbf{X}^l$, defined on

respective lattices S^l, for each grid level $l \geq 0$, for which,

$$S^l = \left\{ s = (i,j) : 0 \leq i < \frac{N_x}{2^l}, 0 \leq j < \frac{M_x}{2^l} \right\}, \tag{8.16}$$

and the image $\mathbf{X}^l$ is defined with respect to the image $\mathbf{X}^{l-1}$ as,

$$\mathbf{X}^l = \left\{ X_s^l = \frac{x_{2i,2j}^{l-1} + x_{2i+1,2j}^{l-1} + x_{2i,2j+1}^{l-1} + x_{2i+1,2j+1}^{l-1}}{4}, s = (i,j) \in S^l \right\} \tag{8.17}$$

8.5.1 Approximations made in the interest of speed

In the open-ended texture classification algorithm, Fig. 8.4, we require a set of LPDF samples from the training image and a set of LPDF samples from the test image. However since the texture classification is based on the similarity of the two sample populations with respect to the Kruskal-Wallis statistic Eq. (8.9), the samples need only reflect the underlying LPDF. Therefore we may substitute the LPDF samples with samples that have a relatively similar distribution.

As the LPDF is derived from a summation of Gaussian curves, Eq. (8.5), this summation may be approximated by the one Gaussian curve that is most significant. This is the Gaussian curve that is derived from the sample data $\mathbf{Z}_p = \mathrm{Col}[y_p, y_q, q \in \mathcal{N}_p]$, $p \in S_y$ closest to the test vector $\mathbf{z} = \mathrm{Col}[x_r, x_t, t \in \mathcal{N}_r]$. In fact, as we only need a relative LPDF distribution, we do not actually need the Gaussian curve, but may instead just use the nominal value $\|\mathbf{z} - \mathbf{Z}_p\|$ from this most significant Gaussian curve. That is, instead of using a distribution of LPDF samples, we use a reflection of this distribution, a distribution of minimum distances.

$$P_r^*(x_r, x_t, t \in \mathcal{N}_r) = \min_{p \in S_y, \mathcal{N}_p \subset S_y} \|\mathbf{z} - \mathbf{Z}_p\|. \tag{8.18}$$

The convenience is that such a distribution is a lot quicker to calculate than one that requires an exponential to be continually called.

8.6 Open-ended classification of textures

To demonstrate the performance of our open-ended texture classification algorithm, Fig. 8.4, we tested it on images containing a mosaic of sub-images with similar grey levels (see Fig. 8.5(a) (b) and Fig. 8.6(a) (b)). A conventional application of

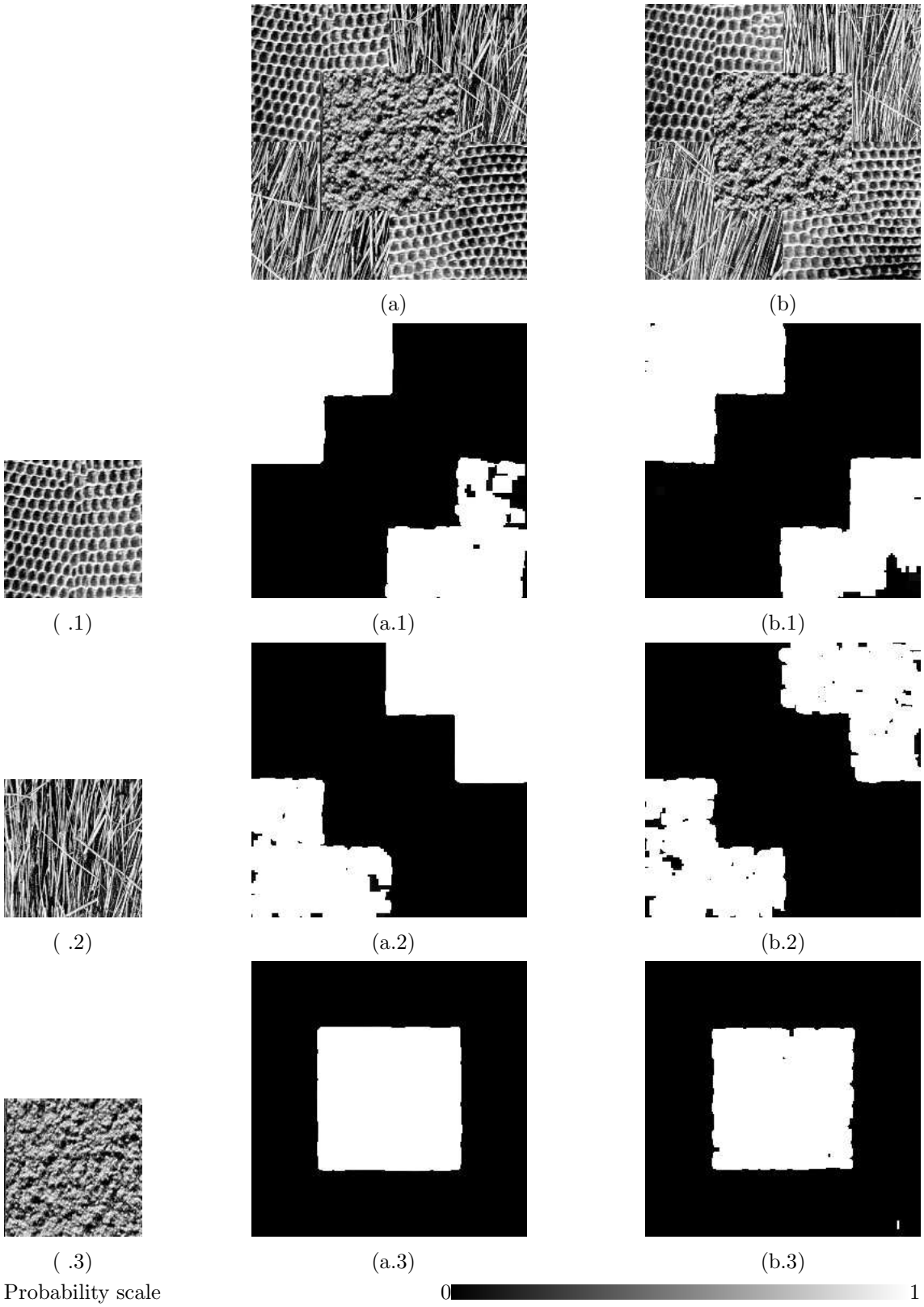


Figure 8.5: Probability maps of Brodatz texture mosaics (a) and (b) with respect to: (.1) D3 - Reptile skin; (.2) D15 - Straw; (.3) D57 - Handmade paper.

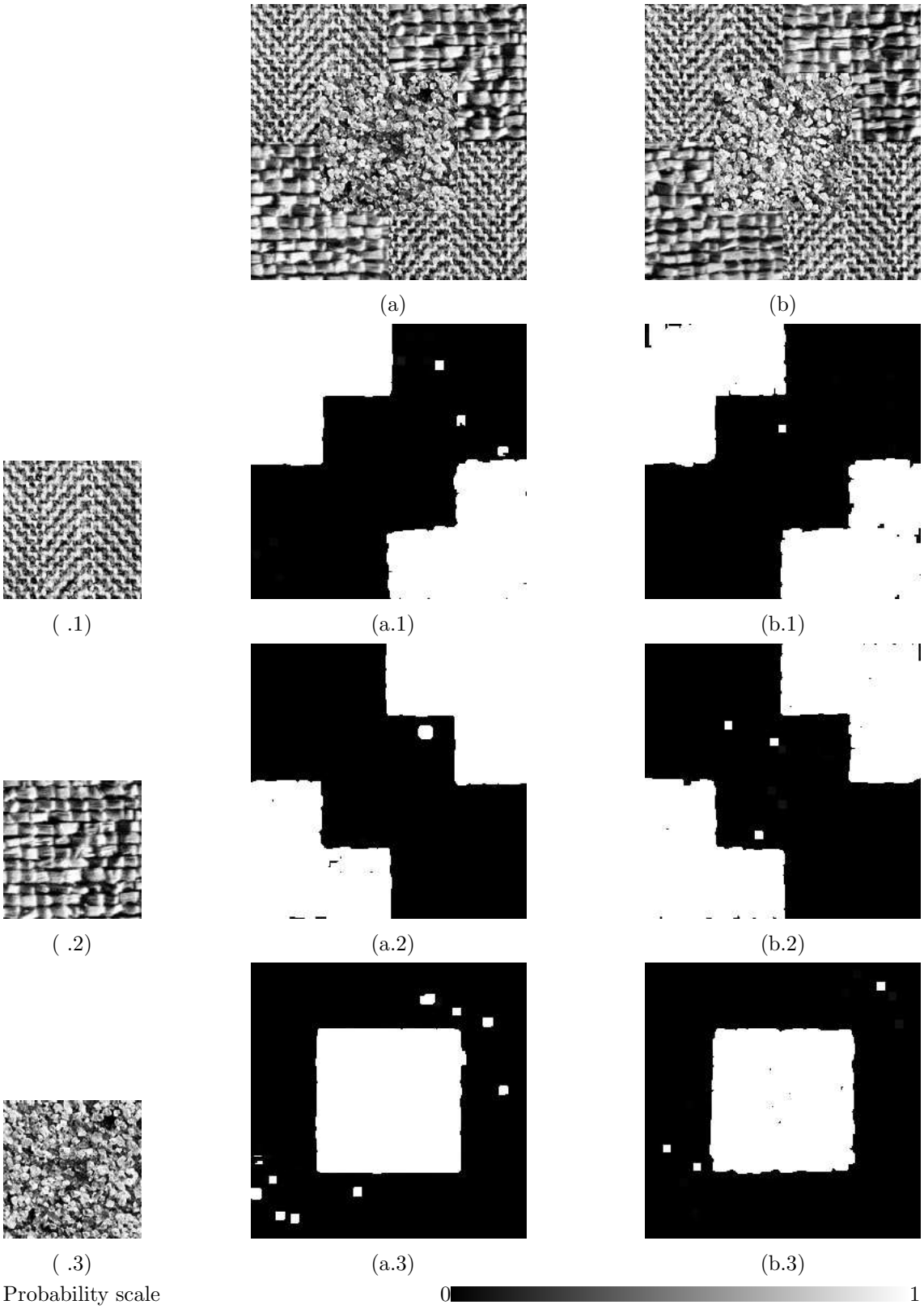


Figure 8.6: Probability maps of Brodatz texture mosaics (a) and (b) with respect to: (.1) D17 - Herringbone weave; (.2) D84 - Raffia; (.3) D29 - Beach sand.

a (first order) histogram technique would not be able to segment these. Also a mix of structured and stochastic sub images were chosen to illustrate how our non parametric technique is able to recognise all types of textures, and not limited to the stochastic textures as suggested by Greenspan *et al.* [91] for parametric MRF models.

The results of the open-ended texture classification algorithm are shown as probability maps representing the goodness-of-fit of a texture model to each region around each pixel, Figs. 8.5 and 8.6. These maps are displayed as grey scale images, with white (grey level 255) designating a probability of one, and black (grey level 0) designating zero probability. However, in the probability maps of Figs. 8.5 and 8.6 there is a distinct absence of varying grey levels. This was caused by a "sharp" goodness-of-fit function in combination with, the dilation function used for edges and boundaries, and Eq. (8.12) which multiplied together the resultant maps from each grid level. In our experiments it was found that the goodness-of-fit probability map at the first grid level contained a range of grey levels, but these were gradually "polarised" as each new probability map was multiplied to it.

The probability maps were each created with respect to just one training texture. For each image, Figs. 8.5(a) (b) and Figs. 8.6(a) (b), three probability maps were created. Figs. 8.5(a.1) (a.2) and (a.3) are the probability maps of Fig. 8.5(a), one for each training texture, Figs. 8.5(.1) (.2) and (.3), respectively. Similarly, the same training textures were used to create the respective probability maps for image, Fig. 8.5(b), as shown in Figs. 8.5(b.1) (b.2) and (b.3). The training textures shown in Figs. 8.5 and 8.6 are the same texture samples used in the top and centre of Fig. 8.5(a) and Fig. 8.6(a), respectively. The rest of the samples used in the texture mosaics of Figs. 8.5 and 8.6 are different texture samples.

The probability maps of Figs. 8.5 and 8.6, show that with the appropriate texture model it is possible to segment and recognise windows of texture with respect to just one training texture and without prior knowledge of the other types of textures in the image. To identify the optimal model, for which to segment and recognise windows of texture, we tested various configurations of the model on a set of 100 VisTex texture mosaics courtesy of Computer Vision Group at the University Bonn [49], and Vision Texture Archive of the MIT Media Lab [203]. The results of this experiment are highlighted in Table 8.1. From this table the optimal model, with an 87% classification accuracy, was the strong MRF model with a 3×3 neighbourhood, pairwise cliques and a maximum grid level of one. Some of the probability maps

Table 8.1: Percentage error for open-ended texture classification of 100 VisTex texture mosaics = percentage area of false negatives + percentage area of false positives. VisTex Texture mosaics courtesy of Computer Vision Group at the University Bonn [49], and Vision Texture Archive of the MIT Media Lab [203]

Neighbourhood Size	*Clique Size*	*Multigrid Height*	*Percentage Error*	*Rank*
3×3	2	0	15.67	6
3×3	2	1	12.94	1
3×3	2	2	13.85	3
3×3	2	3	18.33	8
3×3	3	0	23.70	18
3×3	3	1	18.58	10
3×3	3	2	17.62	7
3×3	3	3	21.80	17
3×3	-	0	24.04	20
3×3	-	1	19.45	12
3×3	-	2	18.40	9
3×3	-	3	21.79	16
5×5	2	0	14.69	4
5×5	2	1	13.48	2
5×5	2	2	15.22	5
5×5	2	3	21.55	15
5×5	3	0	21.45	14
5×5	3	1	18.74	11
5×5	3	2	19.46	13
5×5	3	3	25.48	22
5×5	-	0	25.54	23
5×5	-	1	24.38	21
5×5	-	2	23.98	19
5×5	-	3	30.33	24

obtained from this experiment for this model appear in Appendix C.1. This is also the model we used to obtain the results shown in Figs. 8.5 and 8.6.

It is worthwhile to note that from the results presented in Table 8.1, the nonparametric MRF model scored an average percentage error of 23.49 with a range of [18.40–30.33], while the strong nonparametric MRF model scored an average percentage error of 18.29 with a range of [12.94–25.48]. Therefore the strong nonparametric MRF model clearly outperforms the nonparametric MRF model as predetermined by the *minimax* philosophy [211] which is discussed in Chapter 6.

The power of the open-ended classification is dependent on how low the associated type I and II errors are. If the model achieves a very high goodness-of-fit measurement for all textures within its class, then there is a low likelihood that a texture of the class will be misclassified as not being of the class, and the type I

error associated with the model will be low. Then if also the model achieves a very low goodness-of-fit measurement for all textures outside its class, then there is a low likelihood that a texture not of the class will be misclassified as being of the class, and the type II error associated with the model will also be low. Keeping both type I and II errors low is a matter of finding the right model that gives a high goodness-of-fit to all textures within its class, while maintaining a low goodness-of-fit for all textures outside its class. From the results in Table 8.1, which was performed over 100 VisTex texture mosaics [49, 203], the overall best model for the job is the strong nonparametric MRF model with a 3×3 neighbourhood, just pairwise cliques, and a two tier multigrid. This model gave a type I plus type II error of 12.94%.

Given a set of goodness-of-fit probability maps for each texture model over an image, a classified image may be obtained by labelling each pixel in the image with the texture model class that achieved the highest goodness-of-fit. This is possible since the probability maps have been pre-normalised through the use of the Kruskal-Wallis statistic K. The Kruskal-Wallis statistic K is always chi-squared distributed, and independent with respect to the number of training texture samples. In a real application, the goodness-of-fit probability maps themselves are more likely to be the acceptable output of the open-ended classification algorithm, as given in Fig. 8.4. A goodness-of-fit probability map not only shows the segmentation and recognition of the image with respect to a single texture model, but also the confidence associated with that recognition.

8.6.1 Comparative assessment of performance

Texture models used in todays image analysis are designed for supervised classification. Therefore to make a comparison between the performance of our texture model and the standard models in todays literature, we must test the models under supervised classification. This can be easily done using the MeasTex image texture database and test suite [187]. In the following analysis we use four different test suites; *Grass*, *Material*, *OhanDube*, and *VisTex*. The average score from these test suites appears in the column titled *All*. Finally these scores are ranked from best to worst in each table.

In the first table, Table 8.2, a list of summary scores for a suite of nonparametric MRF models are presented. The key to the nonparametric MRF model names is given in Table 8.3. From first perusal of the table it is evident (by looking at the

relative ranks) that the nonparametric MRF model based on a 3×3 neighbourhood using just 3rd order cliques and a four tier multigrid gives the best performance with about 75% accuracy. This shows that the strong nonparametric MRF model does indeed add an advantageous functionality to the nonparametric MRF model.

The results in Table 8.2 for the nonparametric MRF models can be directly compared to the results in Table 8.4 for the; fractal, Gabor, GLCM, and Gaussian MRF models. The key to these model names is given in Table 8.5. Even the worst performing standard model (the Fractal model) does better than the best nonparametric MRF model (and is computationally more efficient). What this shows is that our method of open-ended texture classification does not translate well to supervised classification.

Table 8.2: MeasTex test suite summary scores – not normalised by priors

Model	*Test Suites*					*Rank*
	Grass	*Material*	*OhanDube*	*VisTex*	*All*	
MRF-n1t0	.732157	.767600	.680725	.680725	.723510	11
MRF-n1t1	.743578	.785322	.674175	.731708	.733695	8
MRF-n1t2	.764700	.784077	.677600	.747062	.743359	3
MRF-n1t3	.766828	.788995	.653425	.748470	.739429	4
MRF-n3c2t0	.638350	.687390	.604525	.650675	.645235	21
MRF-n3c2t1	.629728	.680813	.600075	.674262	.646219	19
MRF-n3c2t2	.621550	.678654	.589850	.692154	.645552	20
MRF-n3c2t3	.598307	.673072	.589975	.696625	.639494	22
MRF-n3c3t0	.720214	.776863	.691475	.709325	.724469	10
MRF-n3c3t1	.729285	.781795	.694400	.730533	.734003	7
MRF-n3c3t2	.747414	.789036	.690425	.749175	.744012	2
MRF-n3c3t3	.754221	.792018	.697400	.748270	.747977	1
MRF-n3t0	.733535	.761781	.668525	.705537	.717344	12
MRF-n3t1	.746742	.782454	.665350	.722929	.729368	9
MRF-n3t2	.766721	.788022	.650625	.742450	.736954	5
MRF-n3t3	.763900	.795795	.640075	.745591	.736340	6
MRF-n5c2t0	.659707	.681550	.601325	.668487	.652767	17
MRF-n5c2t1	.653392	.678340	.597475	.687891	.654274	16
MRF-n5c2t2	.643614	.677272	.586175	.689083	.649036	18
MRF-n5t0	.686642	.726740	.670875	.677470	.690431	14
MRF-n5t1	.678828	.737050	.649075	.699741	.691173	13
MRF-n5t2	.689757	.748400	.621250	.700987	.690098	15

While supervised classification uses a closed N class classifier to find definitive

Table 8.3: Nonparametric MRF model key

Model Name	*Neighbourhood Size*	*Clique Size*	*Multigrid Height*
MRF-n1t0	nearest 4	-	1
MRF-n1t1	nearest 4	-	2
MRF-n1t2	nearest 4	-	3
MRF-n1t3	nearest 4	-	4
MRF-n3c2t0	3×3	2	1
MRF-n3c2t1	3×3	2	2
MRF-n3c2t2	3×3	2	3
MRF-n3c2t3	3×3	2	4
MRF-n3c3t0	3×3	3	1
MRF-n3c3t1	3×3	3	2
MRF-n3c3t2	3×3	3	3
MRF-n3c3t3	3×3	3	4
MRF-n3t0	3×3	-	1
MRF-n3t1	3×3	-	2
MRF-n3t2	3×3	-	3
MRF-n3t3	3×3	-	4
MRF-n5c2t0	5×5	2	1
MRF-n5c2t1	5×5	2	2
MRF-n5c2t2	5×5	2	3
MRF-n5t0	5×5	-	1
MRF-n5t1	5×5	-	2
MRF-n5t2	5×5	-	3

boundaries between the training texture classes, the open-ended texture classifier uses an ideal model to determine a confidence (or a goodness-of-fit) that one type of texture exhibits similar characteristics to another. The difference being, in the later case, no classification boundaries exist by which to discriminate texture classes. Instead each texture model is used to determine the goodness-of-fit between the unknown texture and its training texture. Classification is achieved by assigning, to the unknown texture, the class of the model which achieved the highest goodness-of-fit, unless this goodness-of-fit is too low, then the texture remains "unknown." The disadvantage with such an approach is that discriminative characteristics for a whole set of training texture classes are not as well captured as with supervised classification.

The ability for open-ended texture classification to perform in a supervised clas-

Table 8.4: MeasTex test suite summary scores – not normalised by priors

Model	*Test Suites*					*Rank*
	Grass	*Material*	*OhanDube*	*VisTex*	*All*	
Fractal	.906778	.908636	.904875	.813645	.883483	8
Gabor1	.889978	.967772	.978875	.906591	.935804	3
Gabor2	.880185	.955313	.985975	.898791	.930066	5
GLCM1	.891328	.944863	.883100	.820283	.884893	7
GLCM2	.916157	.964986	.866675	.852266	.900021	6
GMRF-std1s	.917492	.966918	.972000	.885616	.935506	4
GMRF-std2s	.917971	.977545	.991125	.932058	.954674	2
GMRF-std4s	.948892	.969340	.988175	.932437	.959711	1

sification context is governed by the ability of the nonparametric MRF models to produce adequate ideal models of the training texture. That is, models that give a high goodness-of-fit probability for similar texture and low for everything else. This condition is a lot more stringent than what is required of the models in the supervised case. Therefore, unless the nonparametric MRF models are ideal, the use of the open-ended classifier can not be expected to out perform a closed classifier using a standard model where the textures are prior known.

There is also the question of how "similarity" is measured. This is partly determined by the model through neighbourhood size, what clique sizes are used, and the height of the multigrid. However it is also determined by how we measure the classification probability. In our case, we do not use the standard interpretation as given by the combination of various LPDFs, see Eq. (8.4). Instead we use our own unique way of determining a goodness-of-fit probability, by comparing the set of statistics obtainable from the training texture to those obtained from the texture of interest, see Section 8.3. The consequence of this is that when the nonparametric model is under trained, the statistics will not change rapidly as the similarity between a texture and training texture diminishes. Therefore a high goodness-of-fit could be given to a texture that is not all that similar to the training texture. On the other hand, if the nonparametric MRF model was overtrained, then the statistics obtainable from the training texture would have a high entropy, and again dissimilar textures could be given a high goodness-of-fit probability.

Even though the results presented in Table 8.2 show how unamenable the open-ended classifier is to the supervised classification framework, we can still discern

Table 8.5: MeasTex model key

Fractal	Program:	fractalClass
	Command Line:	fractalClass
Gabor1	Program:	gaborClass
	Wavelengths:	2, 4 and 8 pixel
	Angles:	0, 45, 90, 135 degrees
	Mask Size:	17x17 pixels
	Gaussian Window:	texton interpretation (sd = wavelength/2)
	Command Line:	gaborClass -texton -lambda 2,4,8 -theta 0,45,90,135
Gabor2	Program:	gaborClass
	Wavelengths:	2, 4, 8 and 16 pixel
	Angles:	0, 45, 90, 135 degrees
	Mask Size:	17x17 pixels
	Gaussian Window:	texton interpretation (sd = wavelength/2)
	Command Line:	gaborClass -texton -lambda 2,4,8,16 -theta 0,45,90,135
GLCM1	Program:	glcmClass
	Distances:	1 pixel
	Angles:	0, 45, 90, 135 degrees
	Re-quantisation:	32 grey levels
	Rotation Invariance:	average features over angles
	Features:	Energy, Entropy, Inertia, Haralick's Correlation
	Command Line:	glcmClass -q 32 -af -d 1 -impl WDR76 -theta 0,45,90,135
GLCM2	Program:	glcmClass
	Distances:	1 pixel
	Angles:	0, 45, 90, 135 degrees
	Re-quantisation:	32 grey levels
	Rotation Invariance:	average features over angles
	Command Line:	glcmClass -q 32 -af -d 1 -theta 0,45,90,135

some information about the performance of the open-ended classifier. In particular, if we look at the relative rankings of the different models presented in Table 8.2, we can get an overall impression of the effect of varying any one of the nonparametric MRF model's specifications.

Table 8.6 demonstrates the general effect of increasing the neighbourhood size. As the average rank increases with neighbourhood size, we can surmise that a small neighbourhood is better for classification. In Table 8.7 it is the clique size that is varied. From this table we can see that although it is advisable to keep the clique size small, if the the clique size gets too small the model will start to be undertrained. Lastly in Table 8.8 we see that increasing the maximum multigrid height improves the classification accuracy. Just from these three tables we can conclude that the optimal nonparametric MRF model would be MRF-n3c3t3. Note that, although the tables suggest it, the optimal model could not have been MRF-n1c3t3 since

the nearest neighbour neighbourhood can not support third order cliques. Now since the expected optimal nonparametric MRF model is the same one as identified in Table 8.2, we can also conclude that there is not too much interplay between the three different model construction variables. The variables can be used almost independently to optimise the nonparametric MRF model.

Table 8.6: Average rank for various neighbourhoods from Table 8.2

Neighbourhood Size	*Except clique models*	*All models*
nearest 4	6.50	6.50
3×3	8.00	11.17
5×5	14.00	15.50

Table 8.7: Average rank for various clique sizes from Table 8.2

Clique Size	*N3 models*	*All models*
2	20.50	19.00
3	5.00	5.00
-	8.00	9.09

Table 8.8: Average rank for various multigrid heights from Table 8.2

Multigrid Height	*Except clique models*	*All models*
1	12.33	14.17
2	10.00	12.00
3	7.67	10.50
4	5.00	8.25

8.7 Practical application

The final goal of this research was to produce a method by which an operator may be able to take a radar satellite image; segment a small portion from the image where the terrain was known; and use this as the training texture for the ideal texture model. Then with respect to the texture model, find other similar

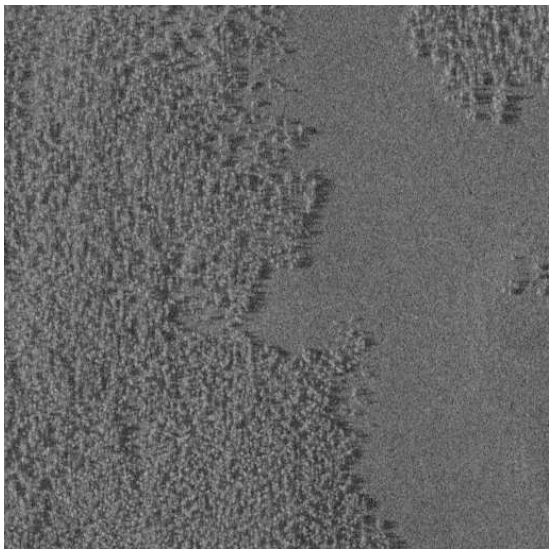
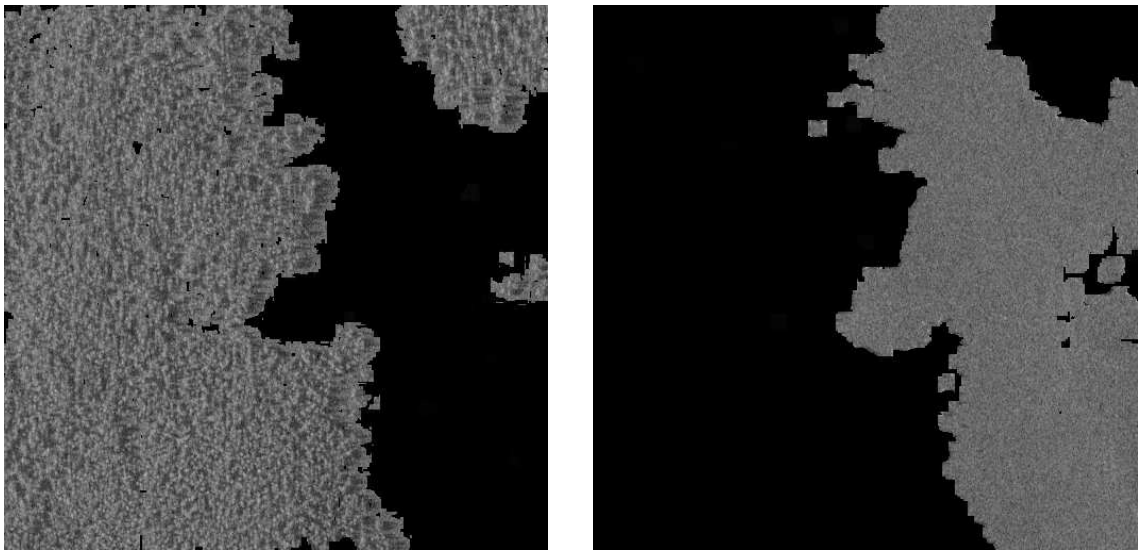


Figure 8.7: Airborne SAR image of Cultana [143].



(a) (b)

Figure 8.8: Probability maps of the trees and grass superimposed on to Cultana image.

terrain types within the image. Such a method of open-ended texture classification would be ideal for terrain mapping of synthetic aperture radar (SAR) images, as it would not require a complete library of textures as for closed N class classifier. With our method, any operator may choose which type of texture they wished to model, without the need for a pre-modelled version existing as part of a library. The nonparametric MRF model is suited to this type of approach as there is no

exhaustive training required to match the model to the texture. However since the probability maps are pre-normalised, best results may be obtained if the user was to iteratively update a labelled image map as probability maps were obtained for each training texture class.

The practical application of segmenting and classifying a SAR image of Cultana, Fig. 8.7, shows the two results if: 1) the operator chose a 64×64 patch of trees from the bottom left corner, Fig. 8.8(a); or 2) the operator chose a 64×64 patch of grass from the bottom right corner, Fig. 8.8(b). Again we have used the same strong MRF texture model as applied in the open-ended classification of the texture mosaics Figs. 8.5 and 8.6. In both cases the resulting probability maps have been superimposed on top of the original SAR image. This gives a clear indication of how the open-ended texture classification has performed. The results show the feasibility of such an approach to segmentation and recognition of texture for terrain mapping of SAR images. Further results of open-ended terrain classification are presented in Appendix C.2. In this case, the results are presented in the form of a combined probability map, where each pixel is given the label of the training texture that achieved the highest goodness-of-fit probability. The open-ended texture classification is not only applicable to terrain recognition, in Appendix C.3 we present further results showing the application of the open-ended texture classifier as a medical diagnostic tool.

Chapter 9

Discussion and Conclusion

The nonparametric Markov random field (MRF) model has been shown to synthesise representative examples of both structured and stochastic texture. This was demonstrated through the use of our multiscale texture synthesis algorithm which used our own novel pixel temperature function to help minimise phase discontinuities in the texture realizations. Although an excellent technique for synthesising texture, its application may be of limited use for now due to the high computational load involved. However we hypothesise that *all* homogeneous textures can be modelled in this way by our nonparametric MRF model.

From our experimental evidence, that being we were able to synthesise texture of high fidelity with respect to a training texture, we conclude that the nonparametric MRF model is close to being an ideal model. That is, a model which can fully characterises a particular texture. With such a model we envisaged it was feasible to use it to recognise similar texture types from an image containing a background of unknown texture types. We achieved this by comparing the texture characteristics captured by the model with the texture characteristics obtained from a region within an image. The nonparametric Kruskal-Wallis test [128] was used to test the null hypothesis that the two sets of texture characteristics were from the same population. The confidence associated with the test was used to produce a probability map of where the training texture class occurred within the image. This type of classification approach was called *open-ended texture classification*. The advantage of such an approach is that it does not require prior knowledge of all the other types of texture present in the image. Such a technique is considered valuable to the practical application of terrain mapping with SAR images [104].

Although the range of textures synthesised, and the high fidelity of the textures synthesised indicate that the nonparametric MRF model is capable of capturing *all* of the visual characteristics of a texture, it was still not considered optimal for open-ended texture classification. To perform open-ended texture classification when unknown texture classes may be present in the image, a model needs to capture those characteristics which uniquely identify the training texture class. However, care should be taken so that the model is not overtrained, otherwise the model will not recognise all textures of the same class. That is, the ideal texture model for open-ended classification should only contain those characteristics specific to the texture class it models.

To acquire the desired ideal model for open-ended classification from our nonparametric MRF model, we followed the minimax entropy philosophy as stated by Zhu Wu and Mumford [211]. Under their philosophy a texture model should maximise its entropy while retaining the unique characteristics of the texture. This amounts to the texture model only modelling known characteristics of a texture, while being completely noncommittal towards any characteristics that are not part of the observed texture, or texture class. Under this philosophy Zhu, Wu, and Mumford [211] designed their minimax model, which was optimised to obtain low entropy for characteristics seen in the texture while maintaining high entropy as a default. This ensured that the model inferred little information about unseen characteristics. In our case, for the nonparametric MRF model, this same philosophy was equivalent to reducing the statistical order of the model while retaining the integrity of the respective synthesised textures. The lower the statistical order of the model, the less likely it was to be overtrained. It was overtraining that hindered its use in the open-ended texture classifier. However, a balance must be sought so that the model remains unique to characteristics of the texture class.

We proposed a second nonparametric MRF model, one that was based on the strong MRF model of Moussouris [148]. This model was shown to be equivalent to the ANOVA construction [67], from which we were able to derive the general ANOVA construction formula Eq. (6.10). The strong MRF model was also shown to be able to synthesise representative versions of a training texture. With this model we were able to limit the statistical order required to uniquely represent a texture, thereby increasing the entropy of the model, which in turn produced a more reliable model for open-ended texture classification.

9.1 Advantages

Although synthesising texture via the multiscale texture synthesis algorithm is computationally intensive, there is a hidden advantage in this method. If the lattice of the synthetic texture was made toroidal (which would be an easy modification) then the generated texture would be ideal for wallpapering or canvassing of a digital surface. The texture synthesis algorithm would only need to be applied once, resulting in a synthetic texture that could be tessellated with no discernible discontinuities. As the synthesis process is able to produce a wide range of synthetic textures with high fidelity, almost any homogeneous texture could be used, resulting in some quite complex wallpapering.

Another important aspect of the nonparametric texture synthesis algorithm is that it can help identify a neighbourhood upper bound required to model a particular texture. With this texture synthesis algorithm we have demonstrated that the nonparametric MRF model is capable of capturing all the relevant characteristics of a particular texture by synthesising subjectively similar texture of a training texture. It may then be said that the neighbourhood used to synthesise this subjectively similar texture would be the upper bound neighbourhood for this particular texture. That is, any larger neighbourhood would be superfluous for this texture model or any other model. Similarly we may also define an upper bound to the relevant statistical order of the texture, a bound that defines a limit to the order of statistics that are significant. This can be achieved by simulating the training texture via our strong nonparametric MRF model.

An advantage of the open-ended texture classification technique is that it requires virtually no training of the texture models, thereby allowing an end user to specify their own type of texture to segment and recognise on an undetermined image. Also the resulting probability maps are pre-normalised in the respect that all values result from a single Kruskal-Wallis test [128]. This allows the end user to iteratively improve their texture classification map by combining probability maps from other open-ended texture classifications.

9.2 Limitations

Although ideally the choice of the strong MRF model should be based on the synthesis results, the texture synthesis algorithm, Fig. 7.4, was too computationally

intensive. However we found that the strong MRF model with a 3×3 square neighbourhood pairwise cliques, and two multigrid levels, was a good general texture recognition model. It must be noted though, any short-comings in the algorithms due to the high computational requirements are not a terminal limitation. Computers are still improving at an exponential rate, and there is little reason why todays research should not anticipate future possibilities.

For the open-ended texture classification algorithm, there is still the problem of choosing the correct structure for the nonparametric MRF model. The size of the neighbourhood, the number of multiscales, and the order of the cliques, are all variables that have to be predefined. This does not make open-ended texture classification independent of human intervention. Therefore a range of results are permissible, and accuracy associated with the results may be hard to define.

When comparing the accuracy of the open-ended texture classification algorithm to a set of conventional N class classifiers, our algorithm did not perform very well. However, the test was performed under a supervised classification framework, which was not conducive to open-ended classification. The problem was that, in order to keep our algorithm efficient, each training texture was used separately. This meant that discriminative characteristics of a whole training texture class were not captured as well as could be achieved by the conventional N class classifiers.

9.3 Future work

As a second order model was found to be the most versatile, in future work it might be worthwhile to change the model used in the open-ended texture classification to another type of second order model, *e.g.,* like those models based on the stochastic modelling of various multi-resolution filter responses [23, 103, 149, 211]. We chose the nonparametric MRF model because we were interested in the search for the order of statistics required for defining (*i.e.,* modelling) textures. In the quest for a fast efficient robust model, one based on multi-resolution filter responses may be more practical.

When using the open-ended texture classification scheme with multiscales and/or the strong MRF model, there is a need for a better confidence test. We took the simple approach of multiplying together all the confidences for each set of statistics from the different multigrid levels and/or cliques. However a better approach would

be to use a single test for multiple population comparisons, something beyond applying multiple Kruskal-Wallis tests [128]. This has been partly addressed by Homer, Meagher, Paget, and Longstaff in [104].

Further research into the open-ended classification of texture via the nonparametric strong MRF model may like to consider such questions as to the effect of shadow over the image, or rotation and rescaling of the texture. Questions that will need to be answered in regards to the practical application of the open-ended texture classification algorithm are: 1) what is the expected accuracy of the classifier; and 2) what is a reliable method for choosing the size of the neighbourhood for the MRF model, or the cliques used in the strong MRF model.

Appendix A

Extracting the Local Clique Set

The Markov random field model, which is used extensively in image processing, is defined with respect to a neighbourhood system. The mathematical interpretation of the model is defined with respect to the corresponding clique set. We present a systematic method for extracting the complete local clique set from any neighbourhood system on which a Markov random field may be defined. The technique described in this Appendix has been published in IEE Proceedings Vision, Image and Signal Processing [154].

A.1 Introduction

Markov Random Field (MRF) models are used in image restoration [82], region segmentation [80] and texture analysis [50]. However the preferred method of analysis in these applications is to use the equivalent Gibbs Random Field model. To obtain this Gibbs model it is first necessary to extract the local clique set from the neighbourhood system defined by the MRF model. This is a complex combinational problem for large neighbourhood systems for which we propose a method to systematically extract the local clique set from any neighbourhood system. Although in practice mostly small neighbourhood systems are used, which may not necessarily benefit from this systematic method of extraction, we found it invaluable in our experiments when we were able to use large neighbourhood systems for nonparametric MRFs.

The property of an MRF is that given a point on a lattice, the probability of that point being set to any particular value is conditional upon the values of its

"neighbouring" points defined by a neighbourhood system. In other words, the MRF is characterised by a local conditional probability function defined with respect to a neighbourhood system. An equivalent Gibbs Random Field defines its probability function over a set of local cliques which are subsets of the neighbouring points [17].

A.2 Neighbourhoods and their cliques

A comprehensive examination of Markov random fields is given by [78]. In this section a brief overview of the MRF theory is presented in order to give the necessary background on neighbourhoods and their respective cliques.

Denote a set of sites on a lattice by S, and the neighbourhood system over S as $\mathcal{N} = \{\mathcal{N}_s, s \in S\}$, where $\mathcal{N}_s$ is the set of "neighbours" for s such that $\mathcal{N}_s \subset S, s \notin \mathcal{N}_s$. Given the random variable X_s at site s with value x_s, the local conditional probability function of a MRF with respect to the neighbourhood system $\mathcal{N}$ is defined by the Hammersley and Clifford theorem [17] as,

$$P(X_s = x_s | X_r = x_r, r \in \mathcal{N}_s) = \frac{1}{Z_s} \exp\left\{ - \sum_{C \in \mathcal{C}_s} V_C(\mathbf{x}) \right\}, \tag{A.1}$$

where Z_s is a constant and V_C is a potential function defined on the clique C. The summation is over all cliques in the local clique set $\mathcal{C}_s$. The variable $\mathbf{x}$ is the set of values $\{x_s, s \in S\}$.

The Hammersley and Clifford theorem implicitly requires the neighbourhood system to adhere to the criterion that $s \in \mathcal{N}_r \Leftrightarrow r \in \mathcal{N}_s$. This implies that neighbourhoods must be symmetrical if the MRF is homogeneous. Three different neighbourhoods are shown in Figs. A.1(a)–(c) which are defined by,

$$\mathcal{N}_s^o = \{r \in S : 0 < |s - r|^2 \leq o\} \ \forall \ r, s \in S, \tag{A.2}$$

as given by [78] where o specifies the order of the neighbourhood.

Given a neighbourhood system $\mathcal{N}$, a clique is a set $C \subseteq S$ such that $s, r \in C, s \neq r$, implies $s \in \mathcal{N}_r$. That is, every pair of distinct sites in a clique are neighbours. The single site subset is also a clique. The local clique set for the site s is defined as $\mathcal{C}_s = \{C \subseteq S : s \in C\}$.

The local clique set for the first-order neighbourhood $\mathcal{N}_s^1$, Fig. A.1(a), is shown

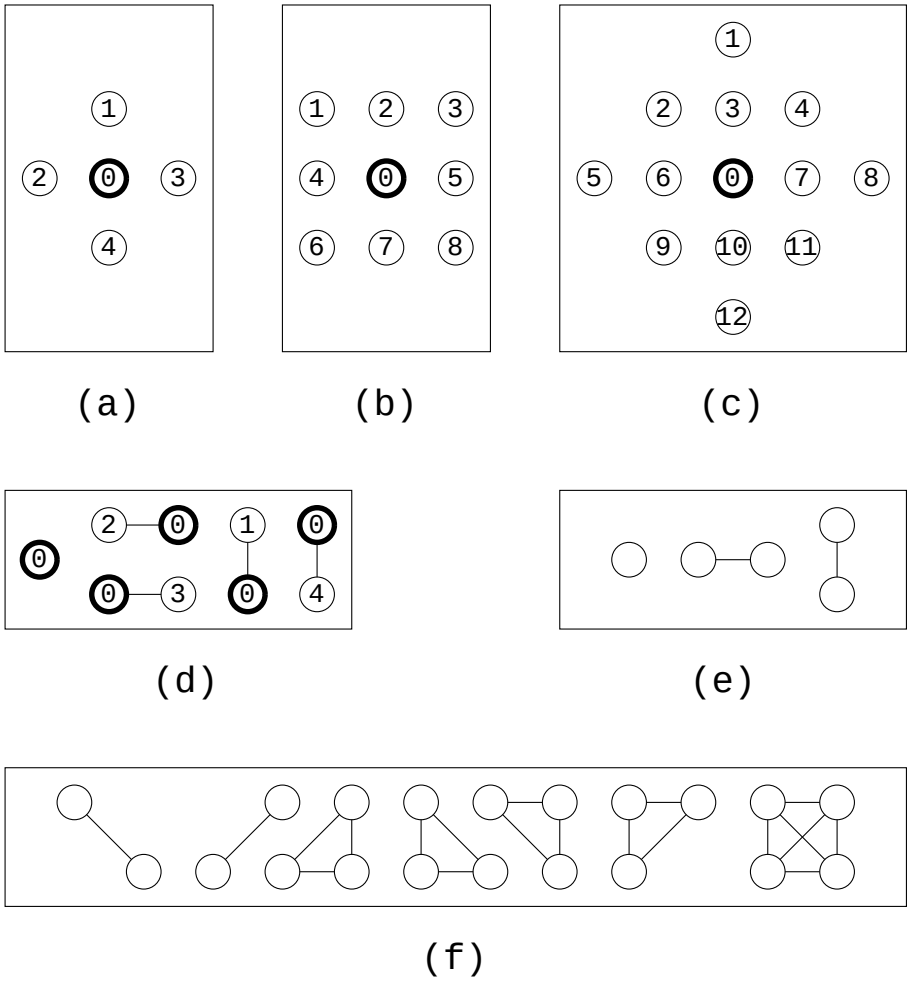


Figure A.1: Neighbourhoods and cliques: (a) The nearest-neighbour neighbourhood; (b) second-order neighbourhood; (c) fourth-order neighbourhood; (d) local clique set for nearest-neighbour neighbourhood; (e) clique types for nearest-neighbour neighbourhood; (f) additional clique types for second-order neighbourhood.

in Fig. A.1(d). This local clique set has three different clique types which are shown in Fig. A.1(e). The local clique set for the second-order neighbourhood, Fig. A.1(b), contains the clique types shown in Figs. A.1(e) and (f).

A.3 Extraction of the local clique set

The proposed method for extracting the local clique set from a MRF neighbourhood system is based on graphing a tree structure. The root of the tree represents a single site. The branches at the first level represent all the pairwise connections to the sites

in the neighbourhood. Further branches at the higher levels represent high order connections that form more complex cliques.

Given a set of sites S, let $n(r)$ denote the node number of the site $r \in S$ with respect to the neighbourhood $\mathcal{N}_s$ where $r \in \mathcal{N}_s$ $s \in S$. Figs. A.1(a)–(c) show the node numbers for neighbourhoods $\mathcal{N}_s^1$, $\mathcal{N}_s^2$ and $\mathcal{N}_s^4$ respectively. The node numbers used in clique trees refer directly to the sites in the respective neighbourhoods.

A.3.1 Method 1: growing the clique tree

Follow the steps outlined in Fig. A.2 as to how to graph the clique tree.

A.3.2 Method 2: reading cliques from the tree

A clique in the tree is represented as any tree transversal following the arrows from one level to the next beginning at the root node (level 1). The single site clique is represented as the single node $n(s) = 0$ at level 1. The pairwise cliques are represented as the node $n(s) = 0$ plus any other node $n(r)$ at level 2. In Fig. A.3 an example of a three site clique is represented by the nodes $\{0, 2, 4\}$.

The complete set of cliques that can possibly be read from the clique tree in Fig. A.3 is the local clique set for the neighbourhood shown in Fig. A.1(b). The clique tree of Fig. A.4 represents the local clique set for the neighbourhood shown in Fig. A.1(c).

A.4 Clique tree theorems

The following theorems prove that the local clique set of a neighbourhood is completely represented by its respective clique tree.

Theorem A.1 *A set of nodes derived from Methods 1 and 2 is a clique*

Proof. By the construction of the clique tree, every node on the tree is contained within the neighbourhood of all other nodes on the tree that can transverse to it by following the arrows.

Theorem A.2 *Each clique represented by the clique tree is unique*

Method 1: *Graphing the Clique Tree*
Input:

$S = \{s, r, t, \ldots\} \leftarrow$ set of sites on a lattice

$\mathcal{N}_s \leftarrow$ neighbourhood for site $s \in S$

$n(r) \leftarrow$ node numbers for sites $r \in \mathcal{N}_s$

Begin

1. Place the node $n(s) = 0$ $s \in S$ at level 1. This is the root node of the tree.

2. Place the nodes $n(r)$ $r \in \mathcal{N}_s$ at level 2.

3. Link the root node $n(s) = 0$ with an arrow to each node $n(r)$ $r \in \mathcal{N}_s$ at level 2.

4. Let $m = 1$.

5. **While** nodes exist at level $m + 1$ **do**

5.1. Increment m

5.2. **For each** $n(r)$ at level m **do**

5.2.1. Let $n(s)$ be the node at level $m - 1$ that directly links to the node $n(r)$ at level m.

5.2.2. Place the nodes $n(t)$ $t \in S$ at level $m+1$ which adhere to the following criteria:

- An arrow directly links the node $n(s)$ at level $m - 1$ with the node $n(t)$ at level m
- $t \in \mathcal{N}_r$
- $n(t) > n(r)$

5.2.3. Link the node $n(r)$ at level m with an arrow to each node $n(t)$ recently placed at level $m + 1$.

5.3. **done**

6. **done**

End

Figure A.2: Method 1: Graphing the Clique Tree

Proof. In growing the clique tree via Method 1, a node $n(s)$ at level m only links to nodes $n(r)$ at level $m + 1$ for which $n(r) > n(s)$. This means that the nodes $\{n(s) = 0, n(r), n(t), \ldots\}$, which can be read from the clique tree via Method 2, are monotonic increasing in node number. Since the nodes are ordered, no permutations of the same set of nodes can be read from the clique tree via Method 2. Therefore each different clique read via Method 2 is unique.

Theorem A.3 *Every local clique is included in the tree*

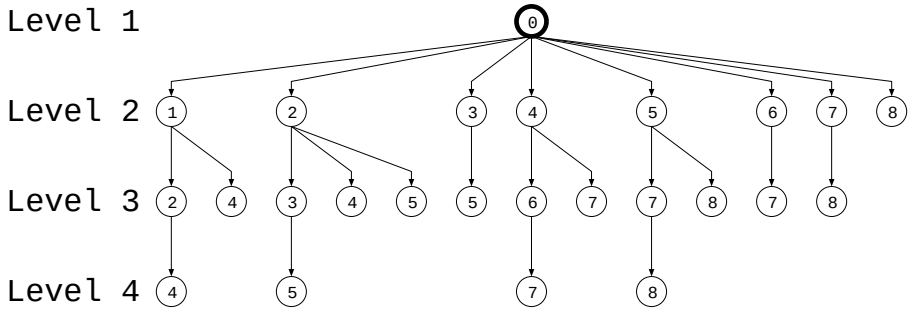


Figure A.3: Clique tree for the neighbourhood shown in Fig. A.1(b).

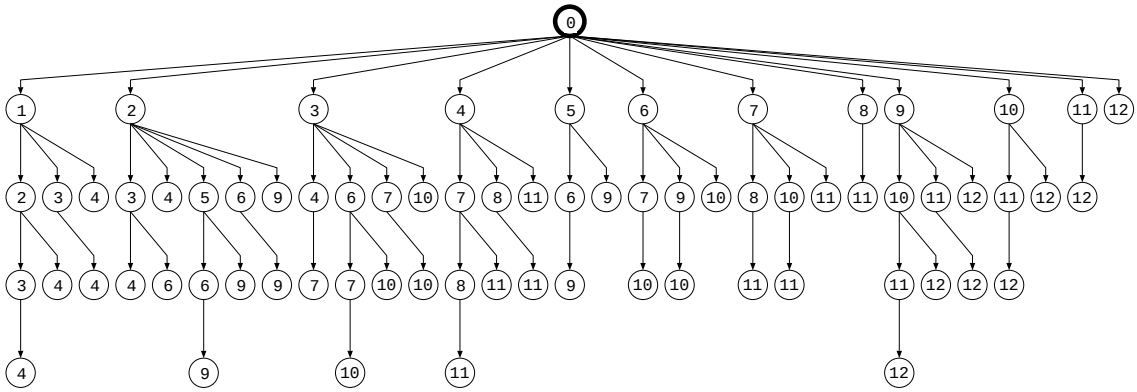


Figure A.4: Clique tree for the neighbourhood shown in Fig. A.1(c).

Proof. Consider any local clique $\{r, \ldots, s\}$ for the site $s \in S$. The clique can be re-arrange into a set of monotonic increasing node numbers given by the neighbourhood $\mathcal{N}_s$ such that,

$$\{t, s \ldots, r\} \Rightarrow \{n(s) = 0, n(r), \ldots, n(t)\}. \tag{A.3}$$

The set of nodes $\{n(s), n(r), \ldots, n(t)\}$ cannot represent a local clique without the first node $n(s) = 0$. The next node $n(r)$ must be contained in the neighbourhood $\mathcal{N}_s$ and is therefore represented at level 2 on the clique tree. Continuing along the list, the next node must be a neighbour to each of the previous nodes. Because of the criteria stated at step 5.2.2 in Method 1, this node exists on the clique tree at level 3 and is linked by an arrow from the node $n(r)$ at level 2. By considering each node from a local clique in a monotonic increasing order, it is clear that by the structure of the clique tree, the local clique must be included in the clique tree.

Theorem A.4 *For each clique type with n nodes,* $\exists$ *n local cliques*

Proof. A local clique of the neighbourhood $\mathcal{N}_s$ is a clique that contains the site s. For a particular clique type with n nodes any one of the nodes may represent the site s. Therefore, there exists n unique local cliques of that type.

A.5 Discussion and conclusion

The clique tree method extracts all the local cliques from any MRF neighbourhood system. The clique tree method only extracts cliques from MRF neighbourhood systems because the clique tree is formed on the premise that $s \in \mathcal{N}_r \Leftrightarrow r \in \mathcal{N}_s$. For a MRF neighbourhood system defined on a homogeneous field, each neighbourhood has to be identical and symmetrical in shape.

The clique tree has been structured so that the local cliques of a particular size reside at the one level. Level 1 holds the single site clique $\{s\}$. The next level, level 2, holds all the pairwise cliques. This ordering of the cliques continues up the levels of the tree until all the local cliques have been accounted for. The tree structure makes it very easy to identify how many local cliques of a certain size exist, it is just the number of sites at the corresponding level of the tree. Therefore, the neighbourhood system of Fig. A.1(b) has 1 single site clique, 8 pairwise cliques, 12 third order cliques, and 4 fourth order cliques. This is shown in Fig. A.3. The total number of cliques in the local clique set is, of course, the total number of sites shown on the tree.

Bibliography

[1] C. O. Acuna, "Texture modeling using Gibbs distributions," *Computer Vision, Graphics, and Image Processing: Graphical Models and Image Processing*, vol. 54, no. 3, pp. 210–222, 1992.

[2] A. Agresti, D. Wackerly, and J. M. Boyett, "Exact conditional tests for cross-classifications: approximation of attained significance levels," *Psychometrika*, vol. 44, pp. 75–83, Mar. 1979.

[3] N. Ahuja and A. Rosenfeld, "Mosaic models for textures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI–3, pp. 1–10, 1981.

[4] H. Akaike, "A new look at statistical model identification," *IEEE Transactions on Automatic Control*, vol. 19, pp. 716–723, 1974.

[5] H. Akaike, "A bayesian analysis of the minimum aic procedure," *The Annals of the Institute of Statistical Mathematics, Part A*, vol. 30, pp. 9–14, 1978.

[6] D. A. Bader, J. JáJá, and R. Chellappa, "Scalable data parallel algorithms for texture synthesis using Gibbs random fields," *IEEE Transactions on Image Processing*, vol. 4, no. 10, pp. 1456–1460, 1995.

[7] M. F. Barnsley, *Fractals everywhere*. Boston: Academic Press Professional, 1993.

[8] M. Basseville, A. Benveniste, K. C. Chou, S. A. Golden, R. Nikoukhah, and A. S. Willsky, "Modeling estimation of multiresolution stochastic processes," *IEEE Transactions on Information Theory*, vol. 38, no. 2, pp. 766–784, 1992.

[9] R. K. Beatson and W. A. Light, "Fast evaluation of radial basis functions: methods for two–dimensional polyharmonic splines," *IMA Journal of Numerical Analysis*, vol. 17, pp. 343–372, July 1997.

[10] Z. Belhadj, A. Saad, S. E. Assad, J. Saillard, and D. Barba, "Comparative study of some algorithms for terrain classification using SAR images," in *Proceedings ICASSP'94 1994 International Conference on Acoustics, Speech and Signal Processing*, vol. 5, (Adelaide), pp. 165–168, 1994.

[11] R. E. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton University press: Princeton, 1961.

[12] C. Bennis and A. Gagalowicz, "2–D macroscopic texture synthesis," *Computer Graphics Forum*, vol. 8, pp. 291–300, 1989.

[13] J. R. Bergen and E. H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, pp. 363–364, May 1988.

[14] J. R. Bergen and B. Julesz, "Rapid discrimination of visual patterns," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13, no. 5, pp. 857–863, 1983.

[15] J. R. Bergen and M. S. Landy, "Computational modeling of visual texture segregation," in *Computational Models of Vision Processing* (M. S. Landy and J. A. Movshon, eds.), pp. 253–271, Cambridge MA: MIT Press, 1991.

[16] J. Berkson, "Maximum likelihood and minimum chi-square estimates of the logistic function," *Journal of the American Statistical Association*, vol. 50, pp. 130–162, 1955.

[17] J. E. Besag, "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society, series B*, vol. 36, pp. 192–326, 1974.

[18] J. E. Besag, "Statistical analysis of non–lattice data," *Statistician*, vol. 24, pp. 179–195, 1975.

[19] J. E. Besag, "On the statistical analysis of dirty pictures," *Journal Of The Royal Statistical Society*, vol. B–48, pp. 259–302, 1986.

[20] J. E. Besag and P. A. P. Moran, "Efficiency of pseudo–likelihood estimation for simple Gaussian fields," *Biometrika*, vol. 64, pp. 616–618, 1977.

[21] Y. M. M. Bishop, "Effects of collapsing multidimensional contingency tables," *Biometrics*, vol. 27, pp. 545–562, Sept. 1971.

[22] Y. M. M. Bishop, S. E. Fienberg, and P. W. Holland, *Discrete Multivariate Analysis: Theory and Practice*. Cambridge: MIT Press, 1975.

[23] J. S. D. Bonet, "Multiresolution sampling procedure for analysis and synthesis of texture images," in *Computer Graphics*, pp. 361–368, ACM SIGGRAPH, 1997. `http://www.ai.mit.edu/~jsd`.

[24] J. S. D. Bonet and P. Viola, "Texture recognition using a non-parametric multi-scale statistical model," in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 1988. `http://www.ai.mit.edu/~jsd`.

[25] J. S. D. Bonet and P. Viola, "A non-parametric multi-scale statistical model for natural images," in *Advances in Neural Information Processing*, vol. 10, 1997. `http://www.ai.mit.edu/~jsd`.

[26] C. Bouman and B. Liu, "Multiple resolution segmentation of textured images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 99–113, 1991.

[27] C. A. Bouman and M. Shapiro, "A multiscale random field model for Bayesian image segmentation," *IEEE Transactions on Image Processing*, vol. 3, no. 2, pp. 162–177, 1994.

[28] P. Brodatz, *Textures – a photographic album for artists and designers*. New York: Dover Publications Inc., 1966.

[29] W. M. Brown and L. J. Porcello, "An introduction to synthetic aperture radar," *IEEE Spectrum*, pp. 52–62, 1969.

[30] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, 1983.

[31] T. Caelli, B. Julesz, and E. Gilbert, "On perceptual analyzers underlying visual texture discrimination: Part II," *Biological Cybernetics*, vol. 29, no. 4, pp. 201–214, 1978.

[32] J. W. Campbell and J. G. Robson, "Application of Fourier analysis to the visibility of gratings," *Journal of Physiology*, vol. 197, pp. 551–566, 1968.

[33] V. Cerny, "Thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm," *Journal Of Optimization Theory And Applications*, vol. 45, pp. 41–51, 1985.

[34] S. Chatterjee, "Classification of natural textures using Gaussian Markov random field models," in *Markov Random Fields Theory And Application* (R. Chellappa and A. Jain, eds.), pp. 159–177, Boston, Sydney: Academic Press, 1993.

[35] B. B. Chaudhuri, N. Sarkar, and P. Kundu, "Improved fractal geometry based texture segmentation technique," *IEE Proceedings Computers And Digital Techniques*, vol. 140, pp. 233–241, Sept. 1993.

[36] R. Chellappa, *Stochastic Models in Image Analysis and Processing*. PhD thesis, Purdue University, 1981.

[37] R. Chellappa and R. L. Kashyap, "Texture synthesis using 2–D noncausal autoregressive models," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP–33, no. 1, pp. 194–203, 1985.

[38] R. Chellappa, R. L. Kashyap, and B. S. Manjunath, "Model–based texture segmentation and classification," in *Handbook of Pattern Recognition and Computer Vision* (C. H. Chen, L. F. Pau, and P. S. P. Wang, eds.), pp. 277–310, Singapore: World Scientific, 1993.

[39] R. Chellappa, "Two–dimensional discrete Gaussian Markov random field models for image processing," in *Progress in Pattern Recognition* (L. N. Kanal and A. Rosenfeld, eds.), vol. 2, pp. 79–122, North–Holland Pub. Co., 1985.

[40] R. Chellappa and S. Chatterjee, "Classification of textures using Gaussian Markov random fields," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP–33, no. 4, pp. 959–963, 1985.

[41] C. C. Chen, J. S. DaPonte, and M. D. Fox, "Fractal feature analysis and classification in medical imaging," *IEEE Transactions on Medical Imaging*, vol. 8, pp. 133–142, June 1989.

[42] C. C. Chen and R. C. Dubes, "Experiments in fitting discrete Markov random fields to textures," in *Proceedings CVPR '89 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, (San Diego, CA), pp. 298–303, IEEE Comput. Soc. Press, June 1989.

[43] C.-C. Chen, *Markov random fields in image processing*. PhD thesis, Michigan State University, 1988.

[44] C. Chubb and M. S. Landy, "Orthogonal distribution analysis: a new approach to the study of texture perception," in *Computational Models of Vision Processing* (M. S. Landy and J. A. Movshon, eds.), pp. 291–301, Cambridge MA: MIT Press, 1991.

[45] M. Clark, A. C. Bovik, and W. S. Geisler, "Texture segmentation using Gabor modulation/demodulation," *Pattern Recognition Letters*, vol. 6, no. 4, pp. 261–267, 1987.

[46] J. M. Coggins, *A Framework for Texture Analysis Based on Spatial Filtering*. PhD thesis, Computer Science Department, Michigan Stat University, East Lansing, MI, 1982.

[47] J. M. Coggins and A. K. Jain, "A spatial filtering approach to texture analysis," *Pattern-Recognition-Letters*, vol. 3, pp. 195–203, May 1985.

[48] F. S. Cohen and D. B. Cooper, "Simple parallel hierarchical and relaxation algorithms for segmenting noncausal Markovian random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 195–219, Mar. 1987.

[49] Computer Vision Group, "Segmentation of textured images," *http://www-dbv.cs.uni-bonn.de/image/segmentation.html*, Apr. 1997.

[50] G. C. Cross and A. K. Jain, "Markov random field texture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 25–39, 1983.

[51] J. Dale, "Asymptotic normality of goodness-of-fit statistics for sparse product multinomials," *Journal of the Royal Statistical Association*, vol. 48, no. 1, pp. 48–59, 1986.

[52] J. G. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, no. 10, pp. 847–856, 1980.

[53] E. J. Delp, R. L. Kashyap, and O. R. Mitchell, "Image data compression using autoregressive time series models," *Pattern Recognition*, vol. 11, no. 5–6, pp. 313–323, 1979.

[54] H. Derin and H. Elliott, "Modelling and segmentation of noisy textured images using Gibbs random fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI–9, no. 1, pp. 39–55, 1987.

[55] H. Derin and C.-S. Won, "A parallel image segmentation algorithm using relaxation with varying neighbourhoods and its mapping to array processors," *Computer Vision, Graphics, and Image Processing*, vol. 40, pp. 54–78, 1987.

[56] R. L. Devalois, D. G. Albrecht, and L. G. Thorell, "Spatial-frequency selectivity of cells in macaque visual cortex," *Vision Research*, vol. 22, pp. 545–559, 1982.

[57] P. Dewaele, L. Van-Gool, A. Wambacq, and A. Oosterlinck, "Texture inspection with self-adaptive convolution filters," in *Proceedings 9th International Conference on Pattern Recognition*, vol. 2, (Rome, Italy), pp. 56–60, IEEE Computer Society Press, Nov. 1988.

[58] L. J. Du, "Texture segmentation of SAR images using localized spatial filtering," in *Proceedings of International Geoscience and Remote Sensing Symposium*, vol. III, (Washington, DC), pp. 1983–1986, 1990.

[59] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *Journal of Applied Statistics*, vol. 16, no. 2, pp. 131–164, 1989.

[60] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Stanford Reseach Institute, Menlo Park, California: John Wiley, 1973.

[61] Earth Resources Observation Systems, "EROS data center: Earthshots," *http://edcwww.cr.usgs.gov/Earthshots*, Feb. 1997.

[62] European Space Agency, *Committee on Earth Observation Satellites: Coordination for the next decade*. United Kingdom: Smith System Engineering Ltd, 1995.

[63] European Space Agency, "Earth observation home page," *http://earth1.esrin.esa.it/index.html*, Jan. 1998.

[64] Y. fai Wong, "How Gaussian radial basis functions work," in *Proceedings IJCNN – International Joint Conference on Neural Networks 91*, vol. 2, (Seattle), pp. 133–138, IEEE, July 1991.

[65] F. Farrokhnia, *Multi-channel filtering techniques for texture segmentation and surface quality inspection.* PhD thesis, Computer Science Department, Michigan State University, 1990.

[66] J. Feder, *Fractals.* Physics of solids and liquids, New York: Plenum Press, 1988.

[67] S. E. Fienberg, *The Analysis of Cross–Classified Categorical Data*, vol. Second Edition. MIT Press, 1981.

[68] R. A. Fisher, *Statistical Methods for Research Workers.* Edinburgh: Oliver and Boyd, 5th ed., 1934.

[69] J. P. Fitch, *Synthetic Aperture Radar.* Springer–Verlag, 1988.

[70] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, vol. 61, pp. 103–113, 1989.

[71] J. M. Francos and A. Z. Meiri, "A unified structural–stochastic model for texture analysis and synthesis," in *Proceedings 5th International Conference on Pattern Recognition*, (Washington), pp. 41–46, 1988.

[72] R. T. Frankot and R. Chellappa, "Lognormal random–field models and their applications to radar image synthesis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 25, pp. 195–207, Mar. 1987.

[73] T. Freeman, "What is imaging radar ?," *http://southport.jpl.nasa.gov/desc/imagingradarv3.html* , Jan. 1996.

[74] K. Fukunaga and L. D. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Trans. Info. Thy.*, vol. IT–21, pp. 32–40, 1975.

[75] A. Gagalowicz, S. D. Ma, and C. Tournier-Lasserve, "Third order model for non homogeneous natural textures," in *Proceedings 8th International Conference on Pattern Recognition (ICPR)*, vol. 1, (Paris, France), pp. 409–411, 1986.

[76] J. Garding, "Properties of fractal intensity surfaces," *Pattern Recognition Letters*, vol. 8, pp. 319–324, Dec. 1988.

[77] J. R. Garside and C. J. Oliver, "A comparison of clutter texture properties in optical and SAR images," in *1988 International Geoscience and Remote Sensing Symposium (IGARSS88)*, vol. 3, pp. 1249–1255, 1988.

[78] D. Geman, "Random fields and inverse problems in imaging," in *Lecture Notes in Mathematics*, vol. 1427, pp. 113–193, Springer–Verlag, 1991.

[79] D. Geman, S. Geman, and C. Graffigne, "Locating texture and object boundaries," in *Pattern Recognition Theory and Applications* (P. A. Devijver and J. Kittler, eds.), New York: Springer–Verlag, 1987.

[80] D. Geman, S. Geman, C. Graffigne, and P. Dong, "Boundary detection by constrained optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 609–628, 1990.

[81] S. Geman and C. Graffigne, "Markov random field image models and their applications to computer vision," *Proceedings of the International Congress of Mathematicians*, pp. 1496–1517, 1986.

[82] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721–741, 1984.

[83] M. A. Georgeson, "Spatial Fourier analysis and human vision," in *Tutorial Essays, A Guide to Recent Advances* (N. S. Sutherland, ed.), vol. 2, ch. 2, Hillsdale, NJ: Lawrence Erlbaum Associates, 1979.

[84] C. J. Geyer and E. A. Thompson, "Constrained Monte Carlo maximum likelihood for dependent data," *Journal of the Royal Statistical Society, Series B*, vol. 54, pp. 657–699, 1992.

[85] J. J. Gibson, *The Perception of the Visual World.* Boston, MA: Houghton Mifflin, 1950.

[86] B. Gidas, "A renormalization group approach to image processing problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 2, pp. 164–180, 1989.

[87] J. W. Goodman, "A random walk through the field of speckle," *Optical Engineering*, vol. 25, no. 5, 1986.

[88] L. A. Goodman, "The multivariate analysis of qualitative data: interactions among multiple classifications," *Journal of the American Statistical Association*, vol. 65, pp. 226–256, Mar. 1970.

[89] C. Graffigne, *Experiments in texture analysis and segmentation*. PhD thesis, Division of Applied Mathematics, Brown University, 1987.

[90] P. J. Green, "Discussion: On the statistical analysis of dirty pictures," *Journal Of The Royal Statistical Society*, vol. B–48, pp. 259–302, 1986.

[91] H. Greenspan, R. Goodman, R. Chellappa, and C. H. Anderson, "Learning texture discrimination rules in multiresolution system," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 9, pp. 894–901, 1994.

[92] G. R. Grimmett, "A theorem about random fields," *Bulletin of the London Mathematical Society*, vol. 5, pp. 81–84, 1973.

[93] T. D. Guyenne and G. Calabresi, *Monitoring the Earth's Environment. A Pilot Project Campaign on LANDSAT Thematic Mapper Applications (1985–87)*. Noordwijk, Netherlands: European Space Agency Publications Division, 1989.

[94] G. J. Hahn and S. S. Shapiro, *Statistical Models in Engineering*. John Wiley and Sons, 1967.

[95] M. Haindl, "Texture synthesis," *CWI Quarterly*, vol. 4, pp. 305–331, 1991.

[96] J. M. Hammersley and P. Clifford, "Markov fields on finite graphs and lattices." unpublished, 1971.

[97] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve," *Radiology*, vol. 143, pp. 29–36, 1982.

[98] R. M. Haralick, "Statistical and structural approaches to texture," *Proceedings of IEEE*, vol. 67, no. 5, pp. 786–804, 1979.

[99] R. M. Haralick, "Texture analysis," in *Handbook of pattern recognition and image processing* (T. Y. Young and K.-S. Fu, eds.), ch. 11, pp. 247–279, San Diego: Academic Press, 1986.

[100] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybertinetics*, vol. 3, no. 6, pp. 610–621, 1973.

[101] M. Hassner and J. Sklansky, "The use of Markov random fields as models of texture," *Computer Graphics and Image Processing*, vol. 12, pp. 357–370, 1980.

[102] J. K. Hawkins, "Textural properties for pattern recognition," in *Picture Processing and Psychopictorics* (B. Lipkin and A. Rosenfeld, eds.), New York: Academic Press, 1969.

[103] D. J. Heeger and J. R. Bergen, "Pyramid–based texture analysis/synthesis," in *Proceedings ICIP–95: 1995 International Conference on Image Processing*, (Washington, D.C.), pp. 648–651, 1995.

[104] J. Homer, J. Meagher, R. Paget, and D. Longstaff, "Terrain classification via texture modelling of SAR and SAR coherency images," in *Proceedings of 1997 IEEE International International Geoscience and Remote Sensing Symposium (IGARSS'97)*, (Singapore), pp. 2063–2065, Aug. 1997.

[105] D. Howard, *Markov random fields and image processing*. PhD thesis, Flinders University, Adelaide, 1995.

[106] R. Hu and M. M. Fahmy, "Texture segmentation based on a hierarchical Markov random field model," *Signal processing*, vol. 26, no. 3, pp. 285–305, 1992.

[107] A. K. Jain and F. F. D. H. Alman, "Texture analysis of automotive finishes," in *Proceedings of SME Machine Vision Applications Conference*, (Detroit, MI), pp. 1–16, Nov. 1990.

[108] A. K. Jain and S. Bhattacharjee, "Text segmentation using Gabor filters for automatic document processing," *Machine Vision and Applications*, vol. 5, no. 3, pp. 169–184, 1992.

[109] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1167–1186, 1991.

[110] B. Julesz, "Visual pattern discrimination," *IRE transactions on Information Theory*, vol. 8, pp. 84–92, 1962.

[111] B. Julesz, "Textons, the elements of texture perception, and their interactions," *Nature*, vol. 290, pp. 91–97, Mar. 1981.

[112] B. Julesz, "A theory of preattentive texture discrimination based on first–order statistics of textons," *Biological Cybernetics*, vol. 41, no. 2, pp. 131–138, 1981.

[113] L. N. Kanal, "Markov mesh models," *Computer Graphics and Image Processing*, vol. 12, pp. 371–375, Apr. 1980.

[114] R. L. Kashyap, "Analysis and synthesis of image pattern by spatial interaction models," in *Progress in Pattern Recognition* (L. N. Kanal and A. Rosenfeld, eds.), pp. 149–186, New York: North–Holland Pub. Co., 1981.

[115] R. L. Kashyap, "Characterization and estimation of two–dimensional ARMA models," *IEEE Transactions on Information Theory*, vol. 30, pp. 736–745, 1984.

[116] R. L. Kashyap and R. Chellappa, "Estimation and choice of neighbors in spatial–interaction models of images," *IEEE Transactions on Information Theory*, vol. IT–29, no. 1, pp. 60–72, 1983.

[117] Z. Kato, J. Zerubia, and M. Berthod, "Unsupervised parallel image classification using a hierarchical Markovian model," in *1995 IEEE 5th International Conference on Computer Vision* (B. Werner, ed.), (Los Alamitos), pp. 169–174, IEEE Computer Society Press, 1995.

[118] J. M. Keller, R. M. Crownover, and R. Y. Chen, "Characteristics of natural scenes related to the fractal dimension," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 621–627, Sept. 1987.

[119] J. M. Keller, S. Chen, and R. M. Crownover, "Texture description and segmentation through fractal geometry," *Computer vision, graphic, and image processing*, vol. 45, p. 150, Feb. 1989.

[120] A. Khotanzad and R. L. Kashyap, "Feature selection for texture recognition based on image synthesis," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 17, pp. 1087–1095, Nov. 1987.

[121] R. Kindermann and J. L. Snell, *Markov Random Fields and their applications*. American Mathematical Society, 1980.

[122] S. Kirkpatrick, C. D. Gellatt, J. Vecchi, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[123] K. J. Koehler, "Goodness-of-fit tests for log-linear models in sparse contingency tables," *Journal of the American Statistical Association*, vol. 81, pp. 483–493, June 1986.

[124] W. L. G. Koontz, P. M. Narendra, and K. Fukunaga, "A graph–theoretic approach to nonparametric cluster analysis," *IEEE Transactions on Computers*, vol. C–25, pp. 936–944, 1976.

[125] S. Krishnamachari and R. Chellappa, "Multiresolution Gauss-Markov random field models for texture segmentation," *IEEE Transactions on Image Processing*, vol. 6, pp. 251–267, Feb. 1997.

[126] S. N. Laboratories, "How SAR works," *http://www.sandia.gov/RADAR/sar_sub/sar_intro1.html*, Jan. 1996.

[127] G. H. Landeweerd and E. S. Gelsema, "The use of nuclear texture parameters in the automatic analysis of leukocytes," *Pattern Recognition*, vol. 10, no. 2, pp. 57–61, 1978.

[128] L. L. Lapin, *Probability and Statistics for Modern Engineering*. Boston: PWS–KENT, 1990.

[129] K. Larntz, "Small-sample comparisons of exact levels for chi-squared goodness-of-fit statistics," *Journal of the American Statistical Association*, vol. 73, pp. 253–263, June 1978.

[130] K. I. Laws, *Textured image segmentation*. PhD thesis, University of Southern California, 1980.

[131] J. H. Lee and W. D. Philpot, "A spectral–textural classifier for digital imagery," in *Proceedings International Geoscience and Remote Sensing Symposium*, (Washington, DC), pp. 2005–2008, 1990.

[132] T. M. Lillesand and R. W. Kiefer, *Remote Sensing and Image Interpretation*. New York: John Wiley and Sons, 1987.

[133] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Transactions on Communications*, vol. COM–28, pp. 84–95, Jan. 1980.

[134] M. R. Luettgen, W. C. Karl, A. S. Willsky, and R. R. Tenny, "Multiscale representation of Markov random fields," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3377–3396, 1993.

[135] A. Lundervold, "Ultrasonic tissue characterisation–a pattern recognition approach," tech. rep., Norwegian Computing Center, Oslo, Norway, 1992.

[136] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *Journal of the Optical Society of America A*, vol. 7, pp. 923–932, May 1990.

[137] B. B. Mandelbrot, *The Fractal Geometry of Nature*. New York: W. H. Freeman, 1983.

[138] B. S. Manjunath and R. Chellappa, "Unsupervised texture segmentation using Markov random field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 478–482, 1991.

[139] D. Marr and E. Hildreth, "Theory of edge detection," *Proceedings of the Royal Society B*, vol. 207, pp. 187–217, 1980.

[140] J. L. Marroquin, S. Mitter, and T. Poggio, "Probabilistic solution of ill–posed problems in compuational vision," *Journal of American Statistics Association*, vol. 82, pp. 76–89, 1987.

[141] C. R. Mehta and N. R. Patel, "Algorithm 643. fexact: A fortran subroutine for fisher's exact test on unordered r x c contingency tables," *ACM Transactions on Mathematical Software*, vol. 12, pp. 154–161, June 1986.

[142] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, pp. 1087–1091, 1953.

[143] Microwave Radar Division, Electronics and Surveillance Research Laboratory at DSTO, Australia, "Project Ingara," 1997.

[144] J. W. Modestino and J. Zhang, "A Markov random field model–based approach to image interpretation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 606–615, June 1992.

[145] Moody and Darken, "Fast learning in networks of locally–tuned processing units," *Neural Computation*, vol. 1, pp. 281–294, 1989.

[146] C. Morris, "Central limit theorems for multinomial sums," *The Annals of Statistics*, vol. 3, no. 1, pp. 165–188, 1975.

[147] A. Mosquera, D. Cabello, and M. J. Carreira, "A fractal–based approach to texture segmentation," *Conference publication*, vol. 354, p. 450, 1992.

[148] J. Moussouris, "Gibbs and Markov random systems with constraints," *Journal of Statistical Physics*, vol. 10, no. 1, pp. 11–33, 1974.

[149] R. Navarro and J. Portilla, "Robust method for texture synthesis–by–analysis based on a multiscale Gabor scheme," in *SPIE Electronic Imaging Symposium, Human Vision and Electronic Imaging '96* (B. Rogowitz and J. Allebach, eds.), vol. 2657, (San Jose, Calfornia), pp. 86–97, 1996.

[150] H. C. Nothdurft, "Texton segregation by associated differences in global and local luminance distribution," *Proceedings of the Royal Society B*, vol. 239, pp. 295–320, 1990.

[151] P. P. Ohanian and R. C. Dubes, "Performance evaluation for four classes of textural features," *Pattern Recognition*, vol. 25, pp. 819–833, Aug. 1992.

[152] R. Paget and D. Longstaff, "Texture synthesis via a nonparametric Markov random field," in *Proceedings of DICTA–95, Digital Image Computing: Techniques and Applications* (A. Maeder and B. Lovell, eds.), vol. 1, (Brisbane, Australia), pp. 547–552, Australian Pattern Recognition Society, Dec. 1995. `http://www.vision.ee.ethz.ch/~rpaget`.

[153] R. Paget and D. Longstaff, "A nonparametric multiscale Markov random field model for synthesising natural textures," in *Fourth International Symposium on Signal Processing and its Applications*, vol. 2, (Gold Coast, Australia), pp. 744–747, ISSPA 96, Aug. 1996. `http://www.vision.ee.ethz.ch/~rpaget`.

[154] R. Paget and D. Longstaff, "Extracting the cliques from a neighbourhood system," *IEE Proceedings Vision, Image and Signal Processing*, vol. 144, pp. 168–170, June 1997. `http://www.vision.ee.ethz.ch/~rpaget`.

[155] R. Paget and D. Longstaff, "Texture synthesis and unsupervised recognition with a nonparametric multiscale Markov random field model," in *Proceedings of 14th International Conference on Pattern Recognition* (A. K. Jain,

S. Venkatesh, and B. C. Lovell, eds.), vol. 2, (Brisbane, Australia), pp. 1068–1070, IAPR, Aug. 1998. `http://www.vision.ee.ethz.ch/~rpaget`.

[156] R. Paget and D. Longstaff, "Texture synthesis via a noncausal nonparametric multiscale Markov random field," *IEEE Transactions on Image Processing*, vol. 7, pp. 925–931, June 1998. `http://www.vision.ee.ethz.ch/~rpaget`.

[157] R. Paget, D. Longstaff, and B. Lovell, "Multiprocessor implementation of a texture segmentation scheme for satellite radar images," in *DICTA–93, Digital Image Computing: Techniques and Applications* (K. K. Fung and A. Ginige, eds.), vol. 1, (Sydney), pp. 203–211, Australian Pattern Recognition Society, Dec. 1993.

[158] R. Paget, D. Longstaff, and B. Lovell, "Texture classification using non-parametric Markov random fields," in *Proceedings of the 1997 13th International Conference on Digital Signal Processing*, vol. 1, (Hellas, Santorini, Greece), pp. 67–70, IEEE Signal Processing Society, July 1997. Invited Paper, `http://www.vision.ee.ethz.ch/~rpaget`.

[159] A. P. Pentland, "Fractal–based description of natural scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 661–674, Nov. 1984.

[160] A. P. Pentland, "Shading into texture," *Artificial Intelligence*, vol. 29, pp. 147–170, Aug. 1986.

[161] D. K. Picard, "Inference for general Ising models," *Journal of Applied Probability*, vol. 19A, pp. 345–357, 1982.

[162] D. K. Pickard, "Inference for discrete Markov feilds: the simplest nontrivial case," *Journal of the American Statistical Association*, vol. 82, pp. 90–96, 1987.

[163] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, pp. 1481–1497, Sept. 1990.

[164] K. Popat and R. W. Picard, "Novel cluster–based probability model for texture synthesis, classification, and compression," in *Proceedings SPIE visual Communications and Image Processing*, (Boston), 1993.

[165] A. Possolo, "Subsampling a random field," in *Spatial Statistics and Imaging* (A. Possolo, ed.), vol. 20, pp. 286–294, IMS Lecture Notes – Monograph Series, 1991.

[166] W. K. Pratt, O. D. Faugeras, and A. Gagalowicz, "Visual discrimination of stochastic texture fields," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 8, pp. 796–804, Nov. 1978.

[167] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*. Cambridge University Press, 2nd ed., 1992.

[168] P. P. Raghu, R. Poongodi, and B. Yegnanarayana, "Unsupervised texture classification using vector quantization and deterministic relaxation neural network," *IEEE Transactions on Image Processing*, vol. 6, no. 10, pp. 1376–1387, 1997.

[169] T. R. C. Read and N. A. C. Cressie, *Goodness–of–fit Statistics for Discrete Multivariate Data*. Springer–Verlag, 1988.

[170] T. R. Reed and H. Wechsler, "Segmentation of textured images and Gestalt organization using spatial/spatial-frequency representations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 1–12, Jan. 1990.

[171] J. A. Richards, *Remote Sensing Digital Image Analysis: An Introduction*. Berlin: Springer–Verlag, 1993.

[172] W. Richards and A. Polit, "Texture matching," *Kybernetic*, vol. 16, no. 3, pp. 155–162, 1974.

[173] E. Rignot and R. Chellappa, "Segmentation of polarimetric synthetic aperture radar data," *IEEE Transactions on Image Processing*, vol. 1, no. 3, pp. 281–299, 1992.

[174] E. Rignot and R. Kwok, "Extraction of textural features in SAR images: Statistical model and sensitivity," in *Pcoceedings International Geoscience and Remote Sensing Symposium*, (Washington, DC), pp. 1979–1982, 1990.

[175] B. D. Ripley, *Statistical inference for spatial processes*. Cambridge: University Press, 1988.

[176] J. Rissanen, "Stochastic complexity," *Journal of the Royal Statistical Society, Series B*, vol. 49, pp. 223–239, 1984.

[177] G. C. Rota, "On the foundations of combinatorial theory," *Zeitschrift Fur Wahrscheinlichkeitstheorie Und Verwandte Gebietee*, vol. 2, pp. 340–368, 1964.

[178] A. H. Schistad and A. K. Jain, "Texture analysis in the presence of speckle noise," in *Proceedings IEEE Geoscience and Remote Sensing Symposium*, (Houston, TX), pp. 147–152, May 1992.

[179] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, pp. 461–464, 1978.

[180] D. W. Scott and J. R. Thompson, "Probability density estimation in higher dimensions," in *Computer Science and Statistics: Proceedings of the Fifteenth Symposium on the Interface*, (Amsterdam: North–Holland), pp. 173–179, 1983.

[181] L. Seymour, *Parameter estimation and model selection in image analysis using Gibbs–Markov random fields*. PhD thesis, The University of North Carolina, Chapel Hill, 1993.

[182] L. H. Siew, R. M. Hodgson, and E. J. Wood, "Texture measures for carpet wear assessment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 92–105, Jan. 1988.

[183] B. W. Silverman, *Density estimation for statistics and data analysis*. London: Chapman and Hall, 1986.

[184] J. Sklansky, "Image segmentation and feature extraction," *IEEE Transactions on Systems, Man, Cybernetics*, vol. 8, pp. 237–247, Apr. 1978.

[185] D. M. Smith, "Speckle reduction and segmentation of synthetic aperture radar images," *International Journal of Remote Sensing*, vol. 17, no. 11, pp. 2043–2057, 1996.

[186] G. Smith, *Image texture analysis using zero crossings information*. PhD thesis, University of Queensland, St Lucia, QLD Australia, 1998, `http://www.cssip.uq.edu.au/~guy/thesis/home.html`.

[187] G. Smith, "Meastex image texture database and test suite," *`http://www.cssip.uq.edu.au/staff/meastex/meastex.html`*, Jan. 1998.

[188] K. R. Smith and M. I. Miller, "A Bayesian approach incorporating Rissanen complexity for learning Markov random field texture models," in *Proceedings ICASSP*, vol. 4, pp. 2317–2320, 1990.

[189] F. Spitzer, "Markov random fields and Gibbs ensembles," *American Mathematical Monthly*, vol. 78, pp. 142–154, 1971.

[190] H. Tamura, S. Mori, and T. Yamawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC–8, pp. 460–473, June 1978.

[191] H. Tamura, S. Mori, and Y. Yammawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, Man, Cybernetics*, vol. 8, pp. 460–473, June 1978.

[192] T. Taxt, P. J. Flynn, and A. K. Jain, "Segmentation of document images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 1322–1329, Dec. 1989.

[193] D. Terzopoulos, "Image analysis using multigrid relaxation methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, pp. 129–139, 1986.

[194] C. W. Therrien, "An estimation-theoretic approach to terrain image segmentation," *Computer Vision, Graphics, and Image Processing*, vol. 22, pp. 313–326, June 1983.

[195] M. Tuceryan, "Moment based texture segmentation," in *Proceedings of the 11th International Conference on Pattern Recognition*, vol. 3, pp. 45–48, Aug. 1992.

[196] M. Tuceryan and A. K. Jain, "Texture analysis," in *Handbook of Pattern Recognition and Computer Vision* (C. H. Chen, L. F. Pau, and P. S. P. Wang, eds.), pp. 235–276, Singapore: World Scientific, 1993.

[197] M. R. Turner, "Texture discrimination by Gabor functions," *Biological Cybernetics*, vol. 55, no. 2–3, pp. 71–82, 1986.

[198] F. T. Ulaby, R. K. Moore, and A. K. Fung, *Active and Passive Microwave Remote Sensing*. Massachusetts: Addison–Wesley, 1982.

[199] F. T. Ulaby and M. C. Dobson, *Handbook of Radar Scattering Statistics for Terrain*. Norwood, Mass.: Artech House, 1989.

[200] M. Unser and M. Eden, "Nonlinear operators for improving texture segmentation based on features extracted by spatial filtering," *IEEE Transactions On Systems, Man, and Cybernet*, vol. 20, p. 804, July 1990.

[201] L. Van-Gool, P. Dewaele, and A. Oosterlinck, "Texture analysis anno 1983," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 336–357, Mar. 1985.

[202] A. Verbeek and P. M. Kroonenberg, "A survey of algorithms for exact distributions of test statistics in r x c contingency tables with fixed margins," *Computational Statistics and Data Analysis*, vol. 3, pp. 159–185, 1985.

[203] Vision and Modeling Group, "Vistex texture." `http://www-white.media.mit.edu/vismod/imagery/VisionTexture/vistex.html%`, 1995.

[204] H. Voorhees and T. Poggio, "Computing texture boundaries in images," *Nature*, vol. 333, pp. 364–367, May 1988.

[205] H. Wechsler, "Texture analysis – a survey," *Signal Processing*, vol. 2, pp. 271–282, 1980.

[206] R. Wilson and M. Spann, "Finite prolate spheroidal sequences and their applications II: Image feature description and segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 2, 1988.

[207] R. Wilson and M. Spann, *Image Segmentation and Uncertainty*. Research Studies Press Ltd, 1988.

[208] M. Woodroofe, "On model selection and arcsine laws," *Annals of Statistics*, vol. 10, pp. 1182–1194, 1982.

[209] J. D. Woodruff, T. L. Angtuaco, and T. H. Parmley, *Atlas of Gynecologic pathology*. New York: Raven press, 2nd ed., 1993.

[210] J. W. Woods, S. Dravida, and R. Mediavilla, "Image estimation using doubly stochastic Gaussian random field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 245–253, Mar. 1987.

[211] S. C. Zhu, Y. Wu, and D. Mumford, "FRAME: filters, random fields, rnd minimax entropy towards a unified theory for texture modeling," *Proceedings 1996 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 686–693, 1996.

[212] S. W. Zucker and K. Kant, "Multiple-level representation for texture discrimination," in *Proceedings IEEE Conference on Pattern Recognition and Image Processing*, (Dallas, TX), pp. 609–614, 1981.