Texture Segmentation by Global Optimization

Sanjay Pandit

Submitted for the Degree of
Doctor of Philosophy
from the
University of Surrey

Vision, Speech and Signal Processing Group
Department of Electronic and Electrical Engineering
University of Surrey
Guildford, Surrey GU2 5XH, U.K.

March 1999
To Shreyas
Abstract

This thesis is concerned with the investigation of a specific approach to the problem of texture segmentation, namely that based on the global optimization of a cost function.

Many tasks in image analysis are expressed as global optimization problems in which the general issue is to find the global minimum of a cost function which describes the interaction between the different variables modelling the image features and the interaction of these variables with the data in a given problem. The minimization of such a global cost function is a difficult problem since the number of hidden variables (labels) is very large and the global cost function may have many local minima. This problem can be overcome to a large extent by using a stochastic relaxation algorithm (for example, Simulated annealing).

Initially, various classical techniques on texture segmentation are reviewed. Ideally, any texture segmentation algorithm should segment an image, so that there is one to one correspondence between the segmented edgels and the ground truth edgels. The effectiveness of an algorithm can be quantified in terms of under and over detection errors for each segmented output image. These measures are used throughout this thesis to quantify the quality of the results.

A particular method which uses global optimization for texture segmentation is initially identified as potentially interesting and is implemented and studied. The implementation proved that this method suffered from many shortcomings and it is not really as good as it was reported in the literature. As the general approach to the problem is a well established methodology for image processing problems, the rest of the thesis is devoted into different attempts to make this method work. The novel ideas introduced in order to improve the method are:

- An improved version of the cost function.
- The use of alternative statistics that characterize each texture.
- The use of a combination of statistics to characterize textures.
- The use of an implicit dictionary of penalizable label configurations, as opposed to an explicit dictionary, leading to penalties applied to anything not acceptable rather than to a selection of unacceptable configurations.
- The introduction of a modified transfer function that maps statistical differences to label differences.
- The use of a database of training patterns instead of assuming that one knows a priori which textures are present in the image to be segmented.
- The use of alternative time schedules with which the model is imposed to the data gradually, in a linear, non-linear and in an adaptive way.

- The introduction of an enhanced set of labels that allows the use of local orientation of the boundary.

- The introduction of a novel way to create new states of the system during the process of simulated annealing in order to achieve faster acceleration, by updating the values of 9 label sites instead of a single label site at a time.

The results obtained by all these modifications vastly improve the performance of the algorithm from its original version. However, the whole approach does not really produce the quality of the results expected for real applications and it does not exhibit the robustness of a system that could be used in practice. The reason appears to be the bluntness of the statistical tests used to identify the boundary. So, my conclusion is that although global optimization methods are good for edge detection where the data are the local values of the first derivative, the approach is not very appropriate for texture segmentation where one has to rely on statistical differences.
Acknowledgements

I would like to thank my supervisor Prof. Maria Petrou, for her guidance and for providing the ideas and motivation that led to this work in the first place.

I would like to express my gratitude to all members in the Centre for Vision, Speech and Signal Processing, for their discussions. In particular, I would like to thank Dr. Bill Christmas for his constructive suggestions. I am also grateful to Rupert Young for his constructive suggestions in software development.

Furthermore, I would like to thank Medha, my wife, for her technical advice and emotional support throughout the studies that led to this work. Also I would like to thank my parents for their constant support.

Finally, I would like to thank the Commonwealth Scholarship Commission for providing me the opportunity for this research.
# Contents

1 Introduction ................................................................. 1

1.1 Defining Texture ...................................................... 2

1.2 Texture Cues ............................................................ 3

1.3 Objective of the Thesis .............................................. 4

1.4 The Scope of This Work ............................................. 5

1.5 Outline of the Thesis ................................................ 7

2 Overview of Texture Segmentation Techniques ............... 9

2.1 Introduction ............................................................. 9

2.2 Feature Based Methods ............................................ 10

2.2.1 Statistical Features .............................................. 11

2.2.2 Operator Based Features ...................................... 17

2.2.3 Transform Domain Features ................................. 20

2.3 Unique Segmentation Based Methods ......................... 21

2.3.1 Region Based Methods ......................................... 21

2.3.2 Boundary Based Methods .................................... 22

2.4 Structural Methods .................................................. 25

2.5 Model Based Methods ............................................. 26

2.5.1 Fractal Model ..................................................... 26

2.5.2 Random Field Model .......................................... 28

2.6 Spatial/Spatial-Frequency Representation .................... 32

2.6.1 The Spectrogram ................................................ 33
2.6.2 The Difference of Gaussians (DOG) Representation . . . .  34
2.6.3 The Wigner Distribution .......................................................... 34
2.6.4 Wavelet Analysis ....................................................................... 37
2.6.5 Multi-Channel Filtering Approach ........................................... 38
2.7 Conclusion ............................................................................................. 43

3 Boundary Detection as a Global Optimization Problem       45

3.1 Introduction ............................................................................................. 45
3.2 Global Optimization: An Overview ................................................... 46
3.2.1 Markov Random Fields and Gibbs Distributions .............. 48
3.2.2 The Bayesian Framework ....................................................... 50
3.3 The Algorithm of Geman et al ............................................................. 53
3.3.1 Stochastic Formulation .............................................................. 60
3.3.2 Results and Discussion .............................................................. 61
3.4 The Modified Cost Function .............................................................. 64
3.4.1 Results and Discussion .............................................................. 65
3.5 The Disparity Measures .......................................................................... 67
3.5.1 Linear Correlation Coefficient ................................................. 68
3.5.2 Chi-Square Test .......................................................................... 68
3.5.3 Contraharmonic Filter .............................................................. 69
3.5.4 Combining the disparity measures .......................................... 70
3.5.5 Selection of normalization constants ................................... 70
3.5.6 Selection of $\phi(\omega)$ .............................................................. 71
3.5.7 Results and Discussion .............................................................. 71
3.6 The Implicit Dictionary .......................................................................... 77
3.6.1 Results and Discussion .............................................................. 78
3.7 The Common Set of Thresholds .......................................................... 84
3.7.1 Results and Discussion .............................................................. 91
3.8 Overview of the Results .............................................................................102
3.9 Summary ...................................................................................................116
# Contents

## 4 The Global Boundary Detection: Incorporating Edge Orientation 117

4.1 Introduction ........................................ 117  
4.2 Theory ............................................. 118  
4.3 Virtual Dictionary ............................... 120  
4.4 Estimation of the Penalty Term ............ 121  
4.5 Experimental Results ......................... 123  
  4.5.1 Finding an initial label assignment .... 124  
4.6 Overview of the Results and Discussion .. 126  
4.7 Summary ......................................... 132  

## 5 Conclusions and Future Work 133

5.1 Conclusions ...................................... 133  
5.2 Future Directions ............................... 136
List of Figures

2.1 Laws' texture masks. ................................................................. 18

3.1 (a) shows pixel sites (o) and boundary sites (+) for a 3 x 3 pixel lattice. Label resolution \( \sigma_l = 1 \). (b) shows pixel sites (o) and boundary sites (+) for label resolution \( \sigma_l = 5 \). ........................................................ 54

3.2 (a) shows pixel pairs (o's) associated with vertical boundary segment \( s, t \) for label resolution \( \sigma_l = 1 \). (b) shows two blocks \( B_s, B_t \) associated with vertical boundary segment for label resolution \( \sigma_l = 3 \). (c) shows two blocks associated with horizontal boundary segment for label resolution \( \sigma_l = 5 \). ............................................................. 55

3.3 Forbidden combinations of labels. ............................................. 56

3.4 Function \( \phi(\omega) \) used in equation 3.28. .............................. 60

3.5 Textural boundary estimation in images. Results with the original cost function and a linear schedule for \( \alpha \) with \( \Delta \alpha = 0.02 \). The Kolmogorov-Smirnov statistic is used as disparity measure. .... 62

3.6 The energy function and its components as a function of the number of iterations of simulated annealing for results of figure 3.5(b). Linear cooling schedule for \( \alpha \) with \( \Delta \alpha = 0.02 \) is used. ............................ 63

3.7 Function \( \phi(\omega) \) used in equation 3.40. \( \omega_{\text{max}} \) here is taken as 3.0. .... 66

3.8 Textural boundary estimation in images. Results with the modified cost function and a linear schedule for \( \alpha \) with \( \Delta \alpha = 0.02 \). The Kolmogorov-Smirnov statistic is used as disparity measure. .... 67

3.9 Textural boundary estimation in some test images. Results with the original cost function and a linear cooling schedule for \( \alpha \) with \( \Delta \alpha = 0.02 \). (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure. ................. 73
3.10 Textural boundary estimation in images. Results with the original cost function and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure. .................................................. 74

3.11 Textural boundary estimation in some test images. Results with the modified cost function and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure. .................................................. 75

3.12 Textural boundary estimation in images. Results with the modified cost function and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure. .................................................. 76

3.13 Example entries of the dictionary. "1" indicates the presence of an edge while blank indicates no edge present. ................................................. 78

3.14 Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. .................................................. 80

3.15 Textural boundary estimation in images. Results with the original cost function, implicit dictionary, and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. .................................................. 81

3.16 Textural boundary estimation in images. Results with the original cost function, implicit dictionary, and a parabolic cooling schedule for $\alpha$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. .................................................. 82
3.17 The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.14(n). The implicit dictionary is used and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$........................................ 83

3.18 The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.15(n). The implicit dictionary is used and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$........................................ 83

3.19 The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.16(n). The implicit dictionary is used with a parabolic cooling schedule for $\alpha$........ 84

3.20 Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure........................................ 85

3.21 Textural boundary estimation in images. Results with the modified cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure........................................ 86

3.22 Textural boundary estimation in images. Results with the modified cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure........................................ 87

3.23 Ten homogeneous textures used for the calculation of the common set of normalization constants for each disparity measure. All images are $128 \times 128$ in size........................................ 89
3.24 Textural boundary estimation in images. Results with the original cost function, implicit dictionary and linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. 93

3.25 Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. 94

3.26 Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. 95

3.27 Textural boundary estimation in images. Results with the modified cost function and the implicit dictionary. The common set of thresholds are used. (a)(b): Results with the Kolmogorov-Smirnov statistic, (c)(d): Results with the linear correlation coefficient measure, (e)(f): Results with the Chi-square measure, (g)(h): Results with the Contraharmonic filter, (i)(j): Results with the joint disparity measure. 96

3.28 Textural boundary estimation in images. Results with the modified cost function and the implicit dictionary. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. 97

3.29 Textural boundary estimation in some test images. Results with the modified cost function and the implicit dictionary. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. 98
3.30 Textural boundary estimation in some test images. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. ................................................................. 99

3.31 Textural boundary estimation in some test images. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure. .......................................................................................... 100

4.1 Dual grid of pixels and labels. ................................................................. 118

4.2 Different blocks with orientations for assigning disparity values to an edgel. ................................................................. 118

4.3 A dictionary entry and the corresponding new dictionary. ................. 120

4.4 Example dictionary entries. —, |, \ and / represent four edgels with $0^\circ, 90^\circ, -45^\circ$ and $45^\circ$ orientations. ................................................................. 121

4.5 An assumed dictionary entry. ................................................................. 122

4.6 Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ in size. ................................. 127

4.7 Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ in size. ................................. 128

4.8 Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ pixels in size. ................................. 129
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The different parameter settings for results of figure 3.8</td>
<td>66</td>
</tr>
<tr>
<td>3.2</td>
<td>The different parameter settings for the results of figures 3.9–3.12</td>
<td>72</td>
</tr>
<tr>
<td>3.3</td>
<td>The normalizing constants for 5 features (Range and 4 Directional residuals)</td>
<td>90</td>
</tr>
<tr>
<td>3.4</td>
<td>The normalizing constants for 5 features (Range and 4 Directional residuals)</td>
<td>91</td>
</tr>
<tr>
<td>3.5</td>
<td>The normalizing constants for 5 features (Range and 4 Directional residuals)</td>
<td>92</td>
</tr>
<tr>
<td>3.6</td>
<td>The normalizing constants for 5 features (Range and 4 Directional residuals)</td>
<td>101</td>
</tr>
<tr>
<td>3.7</td>
<td>Errors for Geman et al’s approach. The Kolmogorov-Smirnov statistic is used as disparity measure with a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>103</td>
</tr>
<tr>
<td>3.8</td>
<td>Errors with the modified cost function. The Kolmogorov-Smirnov statistic is used as a disparity measure with a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>103</td>
</tr>
<tr>
<td>3.9</td>
<td>Errors for various disparity measures. The original cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>103</td>
</tr>
<tr>
<td>3.10</td>
<td>Errors for various disparity measures. The original cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>104</td>
</tr>
<tr>
<td>3.11</td>
<td>Errors for various disparity measures. The modified cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>104</td>
</tr>
<tr>
<td>3.12</td>
<td>Errors for various disparity measures. The modified cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$</td>
<td>105</td>
</tr>
</tbody>
</table>
3.13 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. ................................................. 105

3.14 Errors for various disparity measures. The original cost function is used with the implicit dictionary with a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. .................................................................................................... 106

3.15 Errors for various disparity measures. The original cost function is used with the implicit dictionary and the parabolic cooling schedule for $\alpha$. ................................................................. 106

3.16 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. ................................................................. 107

3.17 Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. ................................................................. 107

3.18 Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. ................................................................. 108

3.19 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 108

3.20 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 109

3.21 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 109

3.22 Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 110

3.23 Edge detection errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 110

3.24 Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. .................... 111
3.25 Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.  

3.26 Edge detection errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.

4.1 The average values of the normalizing constants for 5 features (Range and 4 Directional residuals) for different statistics.

4.2 Errors with the multiple statistics for various images. Errors are estimated over a window of $3 \times 3$. An adaptive cooling schedule for $\alpha$ is used.
Chapter 1

Introduction

Texture is an important element of vision and has been analyzed by researchers in psychophysics as well as computer vision [49] [65] [96]. Any visual phenomenon possesses its own specific texture and hence human beings can use texture as cues to recognize them. Psychophysicalists have studied texture because it provides them with a means of understanding early human visual information processing. Most psychophysical experiments on texture perception have focussed on the preattentive aspect of texture perception. Computer vision researchers have used texture to discriminate between different objects and natural scenes.

There are many tasks where attentive analysis of texture is required. For instance, in the domain of visual inspection, surface texture can be used to check the fidelity of manufacturing processes. Similarly texture is useful feature in the analysis of biomedical images [45], and for the detection of skin cancer [126]. Examples where the texture analysis approaches can be used are [60]:

- Classifying images based on their texture.
- Segmenting an input image into regions of homogeneous texture.
- Extracting surface shape information from the texture gradients.
- Synthesizing textures that resemble natural images for various computer graphics applications.
- Retrieving images with similar textures from a database.

In this work, we consider the problem of texture segmentation which is considered to be most difficult problem by many researchers.

1.1 Defining Texture

Texture is an innate visual property of all surfaces. In image analysis terms, texture refers to a local image context, describing the local properties of the primitive elements and spatial organisation of the primitive elements. The spatial organisation may be random, may have a pairwise dependence of one primitive on a neighbouring primitive, or may have a dependence of $n$ primitives at a time. The dependence may be structural, probabilistic, or functional. Sklansky [105] has suggested the following definition of texture: A region in an image has a constant texture if a set of local statistics or other local properties of the picture are constant, slowly varying, or approximately periodic. Texture, therefore, has both global and local connotations—it is characterized by invariance of certain local measures or properties over an image region. There are three broad classes of textures in computer texture perception for which computational techniques exist: Structural, oriented, and statistical. Structural textures (e.g. a brick wall) can be described in terms of a primitive element that has been repeated in a plane according to a certain placement rule. Oriented textures (e.g. wood grain) are characterized by the orientation field, which assigns a local direction and strength of directionality to each point in the texture. Statistical textures (e.g. sand) are those that show neither repetitiveness nor orientation. They are usually characterized in terms of measures such as fractal dimension or the co-occurrence matrix.

Textures can further be classified as micro-textures and macro-textures. When the texture primitives are small in size and the spatial interactions between primitives are random and constrained to be local, then the resulting texture is a micro-texture.
1.2. Texture Cues

When the texture primitives are large and have their own distinct shape, then the resulting texture is a macro-texture.

1.2 Texture Cues

Most of the psychological research in texture analysis has focussed on preattentive vision. The human visual system exhibits an ability to discriminate between certain kinds of textures preattentively. Psychologists have considered this ability to be based on the preattentive detection and processing of certain primitive features. Julesz [64] was probably the first researcher to perform multidimensional scaling experiments on texture perception. He used a set of sixteen $2 \times 2$ black and white squares to render stochastic textures. He was able to identify certain primitive dimensions of texture perception such as brightness and orientation of clusters. Julesz's work led to the concept of textons, which are features that are responsible for the preattentive discrimination of textures. Some features (textons) are elongated blobs (rectangles, ellipses, or line segments) with specific properties such as colour, orientation, width, length, the ends of lines (terminators), and crossings of line segments. Treisman and Gormican [109] performed an interesting set of experiments in order to understand the extraction of features in early vision. Their research exploited search asymmetries arising from target detection in a background of distractors. They concluded that only a small number of features are extracted early in visual processing, and these include colour, size, contrast, tilt, curvature, and line ends. Rao and Lohse [95] identified three features namely repetition, orientation and complexity from a longer list of features. They concluded that textures can be characterized in terms of these three components. In the light of this research, various computer based techniques were developed for texture analysis.
1.3 Objective of the Thesis

Extensive research has been conducted in texture analysis in the past three decades. A review of existing texture segmentation techniques will be carried out. This review is to be made in order to examine the existing techniques and their merits and demerits. It was aimed at putting in context a certain texture segmentation technique based on a statistical approach which uses a global optimization strategy for texture segmentation post processing. A global optimization algorithm (e.g. Simulated annealing (SA)) has the potential of bringing a smoothly varying energy function with multiple minima to its global minimum regardless of initial conditions [40] [39] [35]. Another advantage of aiming for such a stochastic relaxation algorithm was the design flexibility. SA can be implemented on Markov Random Fields (MRF) which provide a great measure of freedom in energy function definition. Thus complex decision rules (constraints) can be easily incorporated. The main objective of this thesis is to investigate such an approach to the problem of texture segmentation.

Ideally, any texture segmentation algorithm should show one to one correspondence between the segmented output edgels and the ground truth edgels (derived from the input textured image manually) and so the effectiveness of the algorithm can be quantified by estimating the under and over detection errors for each segmented output image.

Once this algorithm was implemented and both errors were estimated for segmented output images, the main objective was how to modify this existing algorithm so as to significantly reduce the errors. Also it was planned to modify the algorithm such that the modified algorithm has the potential for unsupervised texture segmentation.

The particular objective of this thesis are:

- To review and examine existing approaches for texture segmentation.
1.4. The Scope of This Work

The scope of this work is grey level texture segmentation using statistical techniques and global optimization. The main achievements of the work reported in this thesis can be summarized as follows:

A large number of texture segmentation techniques has been reviewed and analyzed. The textural boundary detection using constraint optimization algorithm for grey level texture discrimination proposed by Geman et al [39] was developed. The under and over detection errors were estimated for each output segmented image. Various shortcomings of this algorithm are highlighted and then eliminated in steps. A modified cost function with linear bounded transformation for disparity calculations was proposed. The algorithm proposed by Geman et al [39] uses the Kolmogorov-Smirnov statistic for disparity calculation. It was found that Kolmogorov-Smirnov statistic alone is not sufficient to discriminate a wide variety of composite textures. Three other statistics are proposed, namely the linear correlation coefficient measure, Chi-square measure and the Contraharmonic filter in addition to the Kolmogorov-Smirnov statistic for disparity calculations. A joint disparity measure is then proposed in which disparity calculations are based on multiple statistics. Any statistic giving maximum response at pixel \((i, j)\) out of all four
statistics is used for disparity calculation at that pixel position. The modified algorithm based on individual statistic and multiple statistics is developed. The under and over detection errors are calculated for each of the segmentated output images when individual and multiple statistics are used.

Geman et al used an explicit dictionary of illegal boundary patterns (which are based on prior information about the input test image) which are penalized during optimization. An implicit dictionary of legal boundary patterns is proposed, which are promoted in constrained optimization while the other boundary patterns not part of this implicit dictionary are penalized. A linear schedule is used for penalization.

The normalization selection scheme proposed by Geman et al makes the texture segmentation algorithm supervised. A different normalization constants selection procedure is proposed in which normalization constants are derived from a database of homogeneously textured images and the same set of normalization constants are used for any input textured image to be segmented.

A global texture segmentation algorithm based on multiple statistics and incorporating edge orientation is developed which eliminates the shortcoming of the earlier algorithm. As well as vertical and horizontal edgels, diagonal edgels are also included in the set of allowable labels and a different approach for segmentation is used in this constraint optimization. An extended implicit dictionary which incorporates edgel magnitudes and orientations of permissible configurations, is developed for this algorithm.

The under and over detection errors are calculated for each of the segmentated output images with each of the modifications as above in the existing algorithm and some promising reduction in errors is achieved. The experiments are performed on different types of image and the utility of this modified algorithm is discussed.
1.5 Outline of the Thesis

In chapter 2 various existing texture segmentation techniques are discussed. The techniques are classified into various groups based on unique operators, unique segmentation methods, model based approaches, structural methods and frequency domain methods. The merits/demerits of different techniques are described.

In chapter 3 the problem of texture segmentation by global optimization as given by Geman et al [39] is discussed. The shortcomings of their method are highlighted. A modified cost function using bounded linear transformations for disparity estimation is proposed. I improve their algorithm by incorporating the concept of multiple statistics in place of single statistic for texture boundary localization. I use the linear correlation coefficient, the Chi-square test and the Contraharmonic filter in addition to the Kolmogorov-Smirnov statistic for assigning a disparity value to any boundary segment on the label lattice. A virtual dictionary is proposed in place of an explicit dictionary of Geman et al for penalization of illegal boundary forming labels in constraint optimization.

I propose a different normalization constant selection procedure which aims at deriving a common set of normalizing constants for each statistic from a database of textures. This common set is used in the modified algorithm for segmentation of any of the composite textures. I propose this new scheme of normalization constant selection to make the segmentation unsupervised. The results of such a segmentation are not as good as the results of the supervised segmentation, as expected. I suggest a criterion for evaluating the effectiveness of each of the modifications suggested by estimating the under/over detection errors for each of the segmented output images. I present the results of several experiments I performed.

In chapter 4 I propose another global boundary detection algorithm based on multiple statistics and incorporating edge orientation. I include the diagonal and anti-diagonal edgels alongside the horizontal and vertical edgels in the label lattice and
finally in my cost function. I propose an extension of my earlier virtual dictionary by including the four orientations \(0^\circ, 45^\circ, -45^\circ\) and \(90^\circ\) to any edgel of the label lattice thereby eliminating the shortcoming of the earlier approach that edgels can have only horizontal or vertical orientations.

Finally, I conclude in chapter 5, where the main contributions of this thesis is outlined and future research work in the related directions is suggested.
Chapter 2

Overview of Texture Segmentation Techniques

2.1 Introduction

It is commonly agreed that texture analysis plays a fundamental role in classifying objects and outlining regions. The main problems of concern in texture analysis are [32]:

- Given a textured region, to which of a finite number of classes does the sample belong?
- Given a textured region, how it can be described?
- Given a scene, how can the boundaries between major textured regions can be established.

The third problem (segmentation problem) is mainly difficult because it is usually unknown at which level of complexity a texture is completed up to its physical boundary [32], and also because it is not well understood what kind of grouping mechanism is needed for outlining regions of uniform textures. Texture segmentation has been attempted in various ways in the past. Many authors e.g. [49][100][41]
have mainly grouped segmentation techniques into two categories: statistical techniques and frequency domain techniques. We have followed the nomenclature of Reed et al [99] in grouping the texture segmentation techniques.

Texture segmentation is often achieved in steps of texture feature extraction, feature selection, if the number of features is too large, followed by a segmentation algorithm. Feature extraction methods may be categorized, roughly, as feature based, model based, and structural. In feature based methods, some characteristics of textures are chosen and regions in which these characteristics are constant are determined. Model based methods hypothesize underlying processes of textures and segment using certain parameters of these processes. Structural methods seek to partition images under the assumption that the textures in the image have detectable primitive elements, arranged according to some placement rules. Spatial/Spatial frequency methods are based on image representations that indicate the frequency content in localized regions in the spatial domain.

Once the features are extracted, segmentation methods then analyse the feature space in order to extract homogeneous regions. Segmentation methods in general can be classified into region based, boundary based, or hybrids of the two. The basic difference between boundary based methods and region based methods is that the region based methods seek feature homogeneity while boundary based methods detect feature inhomogeneities for segmentation [41] [99].

### 2.2 Feature Based Methods

We describe in this section, few methods which use unique features for texture segmentation. Such features can, in general, be grouped into statistical features, operator based features and transform domain features.
2.2. Feature Based Methods

2.2.1 Statistical Features

Grey Level Co-occurrence Matrix

Early image texture studies have used autocorrelation functions [66], power spectra, restricted first and second order Markov meshes, and relative frequencies of various grey levels on the unnormalized image [48]. Although they had some degree of success, their experimental results did not aim at defining, modelling or characterizing texture. They only used some form of general mathematical transformation which assigns numbers to the transformed image in a non-specific way. Some attempts were made in developing algorithms for extracting specific image properties such as coarseness and presence of edges and were tried on special classes of image only [110]. Haralick et al [48] were the first who presented a general procedure for extracting textural properties of blocks of image data. The features were calculated in the spatial domain and the statistical nature of textures was taken into account which is based on the assumption that the texture information in an image \(I\) is contained in the overall or average spatial relationship which the grey tones in the image have to one another. A set of grey tone spatial dependence probability distribution matrices were computed and a set of 14 textural features were extracted from each of these matrices. These features contained information about image texture characteristics such as homogeneity, grey tone linear dependencies, contrast number and the nature of boundaries present, and the complexity of the image. We briefly describe here how Haralick et al [48] computed various features from the grey level co-occurrence matrices.

The spatial grey level dependence method (SGLDM) is based on the estimation of the second order joint conditional density functions, \(f(i, j \mid d, \theta)\). Each \(f(i, j \mid d, \theta)\) is the probability of going from grey level \(i\) to grey level \(j\), given the intersample spacing is \(d\) and the direction is given by angle \(\theta\). The estimated values can be written in a matrix form, the so called co-occurrence matrix. If the texture is coarse and \(d\) is small compared to the size of texture primitives, the pairs of pixels at
separation $d$ will have similar grey values which means a high concentration on or near the diagonal axis of the co-occurrence matrix. For a fine texture ($d$ is large), the grey level values of the pixels should differ significantly and the elements on the co-occurrence matrix should be sparsely distributed [41]. Haralick et al, who first used these co-occurrence matrices to classify terrain in aerial photographs, computed 14 features for discriminating between textures, but usually only a set of 5 features are used. These features are:

Energy:

$$E = \frac{1}{N_x-1} \frac{1}{N_y-1} \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} [f(i, j | d, \theta)]^2$$

(2.1)

Entropy:

$$H = -\frac{1}{N_x-1} \frac{1}{N_y-1} \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} f(i, j | d, \theta) \log(f(i, j | d, \theta))$$

(2.2)

Local Homogeneity:

$$L = \frac{1}{N_x-1} \frac{1}{N_y-1} \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} \frac{1}{(i-j)^2} f(i, j | d, \theta)$$

(2.3)

Inertia:

$$I = \frac{1}{N_x-1} \frac{1}{N_y-1} \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} (i-j)^2 f(i, j | d, \theta)$$

(2.4)

Correlation:

$$C = \frac{\sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} (i - \mu_x)(j - \mu_y)f(i, j | d, \theta)}{\sigma_x \sigma_y}$$

(2.5)

where

$$\mu_x = \frac{1}{N_x-1} \frac{1}{N_y-1} \sum_{i=0}^{N_x-1} \sum_{j=0}^{N_y-1} i f(i, j | d, \theta)$$

(2.6)
2.2. Feature Based Methods

\[ \mu_y = \sum_{j=0}^{N_y-1} \sum_{i=0}^{N_x-1} f(i, j \mid d, \theta) \]  \hspace{1cm} (2.7)

\[ \sigma_x = \sum_{i=0}^{N_g-1} (i - \mu_x)^2 \sum_{j=0}^{N_g-1} f(i, j \mid d, \theta) \]  \hspace{1cm} (2.8)

\[ \sigma_y = \sum_{j=0}^{N_g-1} (j - \mu_y)^2 \sum_{i=0}^{N_g-1} f(i, j \mid d, \theta) \]  \hspace{1cm} (2.9)

and \( N_g \) is the number of distinct grey levels in the quantized image.

This method has been quite successful. It has the advantage that values in the co-occurrence matrix depend not only on the coarseness or fineness of image texture, but also on the brightness and the contrast. However, the spatial grey level dependence matrix is then sensitive to histogram stretching. At a large separation \( d \), individual pixels tend to be weakly correlated and the resulting statistics are noisy [121]. If grey levels are compared between local neighbourhoods rather than individual pixels, this effect can be eliminated. This method suffers from the following drawbacks.

1. Many of the texture features derived using this method have little correlation with features visible to the human eye [99]. Tamura et al [52] made an attempt at defining a set of visually relevant texture features. Most of these measures showed a reasonable correspondence to the results of psychophysical tests, in which human subjects ranked the Brodatz textures with respect to these subjective attributes. The six features were selected as: Coarse versus fine, High contrast versus low contrast, Directional versus non directional, Linelike versus bloblike, Regular versus irregular, Rough versus smooth.

2. A large volume of data must be processed. The computational requirements can be reduced by requantizing the image into a smaller number of grey levels thereby reducing the dimensionality of the matrix. Alternatively, Weszka et al [121] suggested the use of absolute grey level differences to create a histogram, and proposed
Chapter 2. Overview of Texture Segmentation Techniques

that features should be computed from this histogram analogous to those used with the full co-occurrence matrix. They found the performance of the two approaches quite comparable. The advantage of Weszka's approach is that it is computationally less expensive. Unser [111] suggested a modification to the co-occurrence matrix approach by defining principal axes of the joint probability densities of two pixels and thereby deriving Sum and Difference (S&D) histograms [112] for texture representation. This method is able to extract features that are similar to Haralick's co-occurrence feature set and it is computationally less expensive.

Grey Level Run Length

The grey level run length method (GLRLM), is based on computing the number of grey level runs of various lengths. A grey level run is a set of linearly adjacent picture points having the same grey level value. Each element of the grey level run-length matrix $p(i, j)$ specifies the number of times the image texture contains a run of length $j$, for pixels having grey level $i$ in the angle $\theta$ direction. These matrices are usually calculated for several values of $\theta$ and various features are computed. The most common features are:

Short run emphasis:

$$\sum_i \sum_j p(i, j) \cdot \frac{1}{j^2} / \sum_i \sum_j p(i, j)$$  \hspace{1cm} (2.10)

Long run emphasis:

$$\sum_i \sum_j p(i, j) \cdot j^2 / \sum_i \sum_j p(i, j)$$  \hspace{1cm} (2.11)

Grey nonuniformity

$$\sum_j (\sum_i p(i, j))^2 / \sum_i \sum_j p(i, j)$$  \hspace{1cm} (2.12)
2.2. Feature Based Methods

Run length nonuniformity

\[
\sum_i \left( \sum_j p(i,j) \right)^2 / \sum_i \sum_j p(i,j)
\]  

(2.13)

Run percentage

\[
\sum_i \left( \sum_j p(i,j) \right) / N_T
\]  

(2.14)

where, \( i,j \) scan the total size of the image in the two dimensions and \( N_T \) is the total number of pixels. A variation of this method was introduced by Galloway [37] to improve the computational efficiency by grouping the grey level values in the image texture into various groups. A generalization of the run length concept was described by Shu et al in [104]. The length of the run through the pixels, \((i,j)\), along direction \( \theta \) is defined as the maximum number of collinearly connected pixels with maximum and minimum grey levels differing by less than a specified threshold. Run lengths in horizontal, vertical, diagonal and anti-diagonal directions are used as features for segmentation.

Abele et al [1] proposed a method which uses both statistical and structural features for segmentation. Features are selected from a large set of features associated with texture primitives. These texture primitives are connected subregions with approximately constant intensity. The texture features are: x-coordinate, y-coordinate, brightness, directionality, orientation, x-regularity, y-regularity, area, roundness, brightness background, window size x, window size y, average intensity, standard deviation, contrast and angular second moment. This 16 dimensional feature vector can be divided into so-called primitive related features (PRF) and window related features (WRF) (measures of first and second order statistics). The features are selected by a space invariant, non-linear operation, which reduces the distance between primitives in the same class, while increasing the distance between different classes.
The distance function between primitives \( Q_i \) and \( Q_j \) is given by [1]

\[
D_{i,j} = \begin{cases} 
\frac{1}{n_{ij}} |(f_i - f_j)t_{ij}| + \delta_{ij} : & \text{if } n_{ij} \neq 0 \\
\delta_{ij} : & \text{if } n_{ij} = 0
\end{cases}
\]

where,

\( f_i, f_j \) are the portions of the feature vectors for \( Q_i \) and \( Q_j \) that are primitive related and

\[
t_{ij} = (s'_i \text{AND} s'_j)
\]

\( n_{ij} = t_{ij}^T t_{ij} \) = number of features set to one in \( s_i \) and \( s_j \). \( d_{ij} \) is the spatial distance between the gravity centers of the primitives and

\[
s_i(k) = \begin{cases} 
1 : & \text{if } \sum_j C_{ij}(k) > 0 \text{ for } j = 1 \text{ to } \# \text{ of primitives.} \\
0 : & \text{if } \sum_j C_{ij}(k) \leq 0 \text{ and } j \neq i
\end{cases}
\]

where,

\( C_{ij} = 1/d_{ij} \cdot \text{SIGN}(t - |f_i(k) - f_j(k)|), \quad k = 3, 10, \)

\( s'_i \) is the complete features switch vector, derived from \( s_i \) under additional texture regularity constraints. \( t \) is a threshold and \( z \) is an arbitrary large number, which increases the distance function, when the feature spaces of \( Q_i \) and \( Q_j \) have no common features. Clustering is then performed with the selected features. Wermser [119] in their comparative study of texture segmentation algorithms, showed that Abele's approach gives very good performance but found it to be computationally expensive.

Lowitz [74] proposed information extracted from local histograms as features for texture segmentation. Mainly, the module and state of the histogram are suggested as possible features which constitute useful local spatial information closely related to texture. The module for the pixel \((m,n)\), centered in a window containing \( N \) pixels, and with \( r \) possible grey levels in the image, was defined by

\[
I_{MH}(m,n) = \sum_{i=1}^r \frac{n_i - N/r}{\sqrt{n_i(1-n_i/N) + N(1-1/r)/r}}
\]
where \( n_i \) is the number of counts at the \( i^{th} \) grey level. The state of the histogram was defined to be the largest count of the local histogram.

Zhu and Yuille [129] proposed a novel statistical and variational approach to image segmentation based on a new algorithm named ‘Region Competition’ which is derived by minimizing a generalized Bayes/MDL (Minimum description length) criteria using the variational principle. Their algorithm can be generalized for multiband segmentation. They demonstrated it on grey level images, colour images, and texture images.

2.2.2 Operator Based Features

The Use of Filter Masks

Laws [72] suggested an operator based feature extraction approach for texture segmentation. The four most important micro masks according to him are shown in figure 2.1. These masks are designed to act as matched filters for certain types of quasi-periodic variations commonly found in textured images. The center weighted filter masks are convolved with the image to be segmentated and statistics, such as the variance, are computed within a window about each pixel in the resulting filtered image. The values of these statistics are assigned as features to the corresponding pixel in the original image. The final features are “texture energy measures” computed using a macro mask. Twelve feature images are formed. Laws’ set of features worked well for the texture classification problem but limited success was achieved in the case of texture segmentation [55]. John et al [55] argued that the major reason for the segmentation error along the region border is due to the “texture energy” method of forming a moving average of the absolute values of the micro-texture features and an improved method for the extraction of texture energy measure was proposed. The other shortcoming of Laws’ approach lies in the definition of the convolution masks, each of which measures fairly specific features of the texture such as ‘edge like’, ‘v-shape like’, ‘high frequency spot like’ and so on. Using this
Chapter 2. Overview of Texture Segmentation Techniques

approach it is necessary to use a combination of a fairly large number of masks in order to characterize a broad range of textures. Also, since the filter masks are relatively small, only high frequency components of texture can be analysed [99].

\[
\begin{bmatrix}
-1 & -4 & -6 & -4 & -1 \\
-2 & -8 & -12 & -8 & -2 \\
0 & 0 & 0 & 0 & 0 \\
2 & 8 & 12 & 8 & 2 \\
1 & 4 & 6 & 4 & 1
\end{bmatrix}
\]

Horizontal edge detector

\[
\begin{bmatrix}
1 & -4 & 6 & -4 & 1 \\
-4 & 16 & -24 & 16 & -4 \\
6 & -24 & 36 & -24 & 6 \\
-4 & 16 & -24 & 16 & -4 \\
1 & -4 & 6 & -4 & 1
\end{bmatrix}
\]

High frequency spot detector

\[
\begin{bmatrix}
-1 & 0 & 2 & 0 & -1 \\
-4 & 0 & 8 & 0 & -4 \\
-6 & 0 & 12 & 0 & -6 \\
-4 & 0 & 8 & 0 & -4 \\
-1 & 0 & 2 & 0 & -1
\end{bmatrix}
\]

Vertical line detector

\[
\begin{bmatrix}
-1 & 0 & 2 & 0 & -1 \\
-2 & 0 & 4 & 0 & -2 \\
0 & 0 & 0 & 0 & 0 \\
2 & 0 & 4 & 0 & 2 \\
1 & 0 & 2 & 0 & 1
\end{bmatrix}
\]

V shape detector

Figure 2.1: Laws' texture masks.

Wang et al [116] described another method which is similar to that of Laws. Instead of using the filter masks suggested by Laws, they used simple (vertical, horizontal, diagonal, and anti-diagonal) masks and a texture measure based on a local-area statistical measure of the variance (or standard deviation) of the intensity image. They defined the local texture feature as Local Directed Standard Deviation (LDSD) over a micro-window while global texture information was captured by means of LDSDs over some macro window. Chen et al [19] proposed an unsupervised texture segmentation scheme using hidden Markov models (HMM's) in which they used Laws' texture mask in their original form for feature map formation.
2.2. Feature Based Methods

MAXDIF Operator

Some visual inspection applications require discrimination between textured and homogeneous regions of images. The MAXDIF operator was proposed by Dinstein et al [30] for fast discrimination between textured and uniform regions. The pixel in the output image corresponding to the center pixel of each $k \times k$ window in the original image is set equal to the difference between the maximum and the minimum grey level occurring in the window. The operator returns a high value for textured regions and low value for homogeneous regions.

General Operator Approach

This General operator approach was proposed by Granlund [43] in which an operator is defined which describes and detects structure as opposed to uniformity, whatever structure implies at a certain level in an image. The effect of this operator is to generate a transformed image of a given input image. An operator field of a certain size, say $9 \times 9$ elements, scans the input image and for each position of the operator field, a complex value is computed for the corresponding position in the transformed or output image. The magnitude which reflects the amount of variation within the window, e.g. step size of an edge, is computed by applying a set of orientation sensitive masks e.g. eight, to the image and then selecting the magnitude and also orientation from the mask giving the largest response. The operator can be used repeatedly upon earlier transformed images to detect structure and simplify the image. Granlund described texture discrimination using a hypothetical example in which the first transformation of image yields a slowly varying field while the second transformation gives the boundary between two fields. The advantage of using such a operator is the following:

There is no need to tune the frequency characteristic of the operator to that of patterns, as a set of operators with different frequency characteristics is used and so the information will be picked up by one or the other [43].
Chapter 2. Overview of Texture Segmentation Techniques

After the first transformation, we obtain a slowly varying field which does not contain the high frequency components existing in the texture but only a description of the structural properties of the texture.

Circular-Mellin Operator

Gopalan et al [97] proposed some Circular-Mellin features which correspond to Fourier features in the polar-log domain. The polar-log coordinate transformation is used to achieve rotation and scale invariance properties. They showed that this operator has similar functional form to that of Gabor operators, described in section 2.6, but its distortion-invariant characteristics make it more suitable as than Gabor operators for texture segmentation.

2.2.3 Transform Domain Features

Xu and Fu [125] proposed a method of natural scene segmentation which is based on:

- Pre-segmentation by multiple thresholds to reduce the number of grey levels in the image and thereby reduce the dimensionality of the co-occurrence matrix.

- Segmentation of texture by using a co-occurrence matrix and a split and merge algorithm.

- Refinement of the segmentation according to texture coarseness based on Walsh transform to split regions that were over-merged previously.

The Fourier Power Spectrum can also used as a measure of texture coarseness but its computation is time-consuming while the Walsh transform requires simple computations.
2.3 Unique Segmentation Based Methods

2.3.1 Region Based Methods

Davis and Mitiche [28] proposed a model-driven, iterative texture segmentation algorithm which combines selective feature smoothing with clustering. In supervised mode, they used statistical texture models supplied by users for determining pixel classification. Each pixel is replaced by an average of the pixels (within an analysis window centered at the pixel of interest) that are in the same class as the pixel replaced and are contiguous neighbours. The classification and smoothing steps are repeated until change in pixel labels becomes small. In unsupervised mode, they replaced the classification step by clustering.

Lumia et al [75] proposed a method which uses the facet model for image segmentation. The image of interest is first segmentated using this model yielding an image in which each pixel is assigned a region number. These region numbers were used in making a region adjacency graph (RAG). For each region, features such as region size, region shape, grey level statistics within the region and so on are derived. The regions are assigned texture categories based on these features, features of adjacent regions, and the features of textures from known images (obtained during a training phase).

Raafat and Wong [93] described a method for image segmentation in which the segmentation process is directed by texture information inherent in the various regions of an image. The resolution dependent texture information measure called I-measure was used to indicate the typicality of image blocks relative to other blocks in the image. A texture dissimilarity (distance) between a pair of texture blocks is defined and used with the I-measure for directing the growth of various homogeneous regions. This is achieved by grouping blocks with low I-measure values. In fine tuning of boundaries, the texture distance measure is used. The boundary blocks are broken into four sub-blocks and each sub-block is merged with the re-
Chapter 2. Overview of Texture Segmentation Techniques

region at the minimum texture distance. They achieved a good segmentation result on a difficult textured image of a fish.

Pietikainen and Rosenfeld [90] described a technique based on pyramid node linking in which pyramid structure is formed by producing images of decreasing resolution, with high resolution image at the top of the pyramid. The images are segmented by linking regions (nodes) of one level with the most similar regions in the neighbouring levels. This can be achieved either by a “top-down” approach, in which case blocks found to be homogeneous are linked to all of their sub-blocks, or a “bottom-up” approach, where sub-blocks similar to the parent block are linked to that block. They also suggested a more effective method in which the above “top-down” and “bottom-up” approaches are combined.

Reed et al [98] suggested a diffusion region growing technique. The region growing process was described as a modified random walk, where the probability of moving in a particular direction and adding the pixel which is encountered to the region being grown, is determined by the similarity of the current and adjacent pixel characteristics. They showed that the method is able to grow several separate (but similar) regions simultaneously and hence it is well suited for parallel implementation.

2.3.2 Boundary Based Methods

Grinaker [44] proposed a generalization of a common edge detector for texture segmentation. A feature gradient function (FGF) is formed by applying the gradient operator to a set of features derived from the image and then summing all the weighted resolution consistent feature gradients. The weights are taken as variable vector functions in order to make optimal use of information about edge orientations and expected feature values of the segments. Frequency consistency is imposed on FGF by transforming all feature functions to the same resolution. First of all, a very coarse edge detection is determined by using frequency consistent FGF, where low resolution information is used. Then, step by step, increase in the
accuracy of edge localization is obtained by using increasing frequency FGF's. Although experiments were run on only two artificially constructed textured images, they highlighted how to incorporate this edge based segmentation in a general high-performance system.

Diederich Wermser [118] proposed a boundary based unsupervised texture segmentation method which is based on the computation of texture gradient which is a measure of amount and direction of local change of texture. The texture gradient gives a probability for the existence of a texture boundary and its probable direction for each pixel of an image. The features used for texture characterization were determined by a model of the human texture perception and weighted according to the results of subjective tests carried out with a large set of natural textures from the Brodatz texture album. The magnitude $a$ and direction $d$ of the local texture gradient $TG(a,d)$ was found by using a set of four masks (horizontal, anti-diagonal, vertical and diagonal), each with two subregions. Each mask was moved over all 14 feature images, thus extracting 8 set of feature vectors representing the textures in the subregions within each of the four masks. For each mask, a dissimilarity measure $s_n$ was determined by comparing the feature vectors extracted with the sub-regions. The magnitude of texture gradient $TG(a,d)$ was given by:

$$ a = \text{Max}(s_n) \quad (2.16) $$

The gradient direction $d$ is encoded as the number, $n$, of the mask yielding the maximum $s_n$. The best method for determining the dissimilarity measure was found to be the ratio of inter- and intra-class distance. The disadvantage of the method pointed out that it is very time consuming. Experiments were shown in which segmentation is achieved by thresholding the magnitude of texture gradient and applying line thinning to the detected boundary.

Hofman and Buhmann [53] proposed a deterministic annealing approach to pair-
wise data clustering which is used to segment textured images. Pairwise data clustering is combinatorial optimization method for data grouping which extracts hidden structure from proximity data. Later Hofmann et al [54] presented a framework for unsupervised texture segmentation that relies on statistical tests for texture comparison. A scale space approach for data representation based on Gabor filters (see section 2.6) has been suggested and the segmentation is formulated as a pairwise data clustering problem based on dissimilarities between texture blocks with a sparse neighbourhood structure. They developed a general mathematical framework for applying the optimization principle of deterministic annealing to arbitrary partitioning problem. The experimental results of deterministic algorithm were compared with the ICM algorithm [8] and the Gibbs sampler [40] and superior results with the former were shown.

Haris et al [50] proposed a hybrid multidimensional image segmentation algorithm which combines edge and region based technique through the morphological algorithm of Watersheds. The region adjacency graph (RAG) is used to represent the image partitions and is combined with nearest neighbour graph (NNG), in order to accelerate the region merging process. Experimental results obtained with 2-D/3-D magnetic resonance images were shown. They pointed out that the drawback of the method is the high memory requirement imposed by the Watershed algorithm.

Chakraborty and Duncan [17] argued that the conventional methods of region-based segmentation and gradient-based boundary finding are often frustrated by poor image quality. They proposed a technique for integrating region-based segmentation and gradient-based boundary finding using the game-theoretic framework. They posed this as a nonzero sum two-person noncooperative game where the cost functions of both the modules are based on the Bayesian theory of maximization of the a posteriori probability. The main advantage of using the game-theoretic integration is that it can bring together the incommensurate region and boundary methods that operate in different probability spaces into a common information-sharing frame-
work of rational decision making which leads to improved output for both the modules without excessive computational overload. From the experimental results on synthetic and natural images, they concluded that this integrated approach is more robust to both noise and poor initialization.

2.4 Structural Methods

Structural texture segmentation methods assume that textures are composed of well-defined texture elements. Jayarammurthy et al [62] proposed a deconvolution filter for texture discrimination in the spatial frequency domain. According to their approach, any two dimensional texture scene \( t(x,y) \) may be considered as a convolution of a sub-pattern \( h(x,y) \) and an array of impulse functions \( c(x,y) \), which ultimately, define the placement rules. Thus

\[
t(x,y) = h(x,y) * c(x,y)
\]

where

\[
c(x,y) = \sum \delta(x-x_m,y-y_n)
\]

and \( x_m, y_n \) are the coordinates of the impulse functions. In the spatial frequency domain the above expression can be rewritten as,

\[
T(u,v) = H(u,v)C(u,v)
\]

We can rewrite the above expression as:

\[
C(u,v) = T(u,v)H(u,v)^{-1}
\]

Thus if \( t(x,y) \) and \( h(x,y) \) are given then it is possible to obtain \( c(x,y) \). In other words, a filter whose frequency transfer function is \( H(u,v) \) acts as deconvolution filter and recovers \( c(x,y) \) from \( t(x,y) \). This filter when applied to an image containing the texture of interest, results in an array of impulses in the region of image...
containing that texture. It was shown that this approach is insensitive to adverse effects such as overlapping of textures and illumination but suffers from sensitivity to any changes in the sub-pattern such as magnification and orientation, which alter the Fourier transform significantly.

Matsuyama et al [80] proposed a structural approach in which they extracted texture features from a regular texture by region growing. They assumed that a texture element is composed of connected pixels with similar grey levels. They calculated the vectors between the elements. They estimated the location of missing elements and then produced a relatively compact description of placement of the elements. Later segmentation is achieved by moving a template of each texture element around the boundaries of their respective regions according to the placement rule.

You et al [128] proposed an approach for image segmentation by texture classification based on fractional discrimination functions. The method is based on enhancing the texture edge points by means of image decomposition and contextual filtering in terms of the proposed fractional function. They concluded that using this method it is possible to extract features which cannot be normally extracted using statistical and structural methods.

2.5 Model Based Methods

2.5.1 Fractal Model

Fractal geometry has received increased attention as a model for natural phenomena [77]. Pentland [88] presented evidence that most natural surfaces are spatially isotropic fractals and that intensity images of these surfaces are also fractals. This provided the foundation for the use of features derived from fractal models in image analysis. We describe in brief how fractal dimensions are calculated [68].

Consider a bounded set \( A \) in a Euclidean \( n \)-space. The set \( A \) is said to be self-similar when \( A \) is the union of \( N \) distinct (non-overlapping) copies of itself, each of
which has been scaled down by a ratio $r$ in all coordinates. The fractal or similarity dimension of $A$ is given by the relation,

$$D = \frac{\log N}{\log(1/r)}$$  \hspace{1cm} (2.21)

It is often difficult to estimate the fractal dimension using the above rule. A more general approach for finding a relative measure of fractal dimension, the box dimension was suggested. Suppose one can cover the set $A$ with an $n$-dimensional box of size $L_{\text{max}}$. If the set $A$ is scaled down by ratio $r$, then the number of boxes of size $L = rL_{\text{max}}$ needed to cover the whole set is given by:

$$N(L) = \frac{1}{r^D} = \left[\frac{L_{\text{max}}}{L}\right]^D$$  \hspace{1cm} (2.22)

Thus, $D$ can be estimated by dividing the $n$-dimensional space into a grid of boxes with side $L$ and count the number of non-empty boxes.

Various approaches were used for the estimation of the fractal dimension. Pentland [88] estimated the fractal dimension by examining the one-dimensional power spectrum along several directions and by averaging the results. They showed that using the fractal dimension has the added advantage of being invariant to scaling of the image. Pelag et al [87] derived 48 features based on the $\varepsilon$-blanket method of estimation of fractal dimension. Their observation was that textures in general are not fractals and at best can be considered so over a limited range of scales. James et al [68] also claimed that fractal dimension alone does not provide sufficient information to describe and segment natural textures. They proposed a new box dimension estimate for describing and segmenting fractal surfaces. They introduced new features based on the concept of lacunarity which capture the second-order statistics of fractal surfaces. They used a K-means clustering algorithm for final segmentation.
Chaudhary et al [18] proposed a modified box counting method (Differential Box Counting) for estimating the fractal dimension. This new Differential Box Counting approach was shown to be faster and more accurate than other box-counting approaches. They proposed a set of six features in order to discriminate between different textures that may have the same fractal dimensions [78]. The six features were based on the fractal dimension of the original image, the high grey-valued image, the low grey-valued image, the horizontally smoothed image, the vertically smoothed image, and the multi-fractal dimension of the original image. Again an unsupervised K-means like clustering algorithm was used for scene segmentation.

2.5.2 Random Field Model

Model based approaches were initially introduced in the field of texture synthesis but were also found to be quite useful in the area of texture analysis by many researchers. We describe here a few methods using different random field models.

The image texture in these model based techniques is normally described as a multivariate probability distribution. Each texture class is described by an appropriate set of parameters. The commonly used models are: autoregressive models, Gaussian Markov random field models, and Gibbs-Markov random field models. Autoregressive models are derived from time series models used in the field of signal processing. These models describe the linear dependence of a pixel on its neighbourhood [29]. Although these models involve simple computations, they are not effective in capturing the structure of natural textures [106].

Kashyap and Khotanzad [67] proposed a rotationally invariant model in the form of the circular autoregressive model (CAR) and they addressed the problem of texture segmentation based on two SAR (simultaneous autoregressive model) and CAR models and presented segmentation results on natural textures consisting of micro and macro textures.
Cross and Jain [26] used the Markov Random field (MRF) as the model and produced blurry, sharp, line-like and blob-like textures. They computed the MRF parameters of some natural texture and then used the same parameters to reconstruct the texture.

Manjunath and Chellappa [79] proposed an unsupervised texture segmentation algorithm using Gauss Markov random field models (GMRF). They assumed that the texture intensity distribution can be modelled by second order GMRF and so the problem was reduced to estimating the GMRF parameters for segmentation. A modified algorithm which minimizes the expected classification error per pixel (or maximizes the posterior marginal (MPM) distribution) was used for obtaining segmentation. They also proposed an algorithm which combines the deterministic algorithm of Besag [8] with stochastic learning which has the advantage that it requires a smaller number of iterations compared to simulated annealing [40] and the results are better than using the deterministic relaxation alone. They showed results of texture segmentation on images having 2 and 3 homogeneous textures using stochastic learning and the MPM algorithm.

A supervised texture segmentation approach using the maximum likelihood (ML) estimation principle was proposed by Cohen and Cooper [22]. Later Cohen and Fan [23] proposed an unsupervised texture segmentation algorithm again based on the maximum likelihood (ML) estimation principle. The different textures in the image were modelled by Gaussian Markov random models. A two stage procedure was adopted to achieve ML segmentation. First, the image is partitioned into disjoint square windows and a coarse segmentation is obtained by combining windows into homogeneous regions using unsupervised ML window segmentation which consists of a ML grouping of the windows into a fixed number of regions and finding the best number of regions. From this coarse segmentation, the parameters of different texture regions are estimated and later used by the supervised ML high resolution segmentor for segmenting the potentially mixed window. Experimental results were
shown on synthesized textured images and on a real outdoor scene.

Won and Derin [124] proposed a general unsupervised texture segmentation algorithm based on a Markov Random field model which estimates all model parameters, including the number of regions, as part of the segmentation. An alternative optimality criterion to maximum a posteriori (MAP) was proposed which yields a partial optimal solution (POS) for the model parameters and the region labels. The POS is such that the POS of the region labels maximizes the objective function with respect to the region labels, given the POS of the parameters, and vice versa. Experimental results were shown on images synthesized from the model, hand drawn images, and natural images.

Nguyen and Cohen [85] suggested an unsupervised texture segmentation strategy where textures are modelled as Gaussian Gibbs fields. The segmentation is achieved in two stages. First, the model parameters were evaluated from disjoint blocks which are classified as homogeneous. This phase of the algorithm used a fuzzy clustering procedure to determine the number of textures in the image and to roughly locate the corresponding regions. The second phase involves fine segmentation of images, using the Bayesian local decisions based on the previously modelled parameters. Experimental results were shown on images composed of natural textures taken from Brodatz [13] album, aerial images, and images of micro-structures which are commonly encountered in metallurgical engineering.

Yin and Allinson [127] proposed an unsupervised texture segmentation algorithm based on a hierarchical neural structure. The textures are modelled by MRF. The local MRF model parameters are obtained by randomly placing a window on the image at each iteration. These crude or noisy parameters are used as input to a one dimensional Kohonen self-organising map (SOM) network, whose size is determined by the number of regions to be segregated. In the second phase, a local voting network, which represents the region label of the pixels, updates the label votes according to the winner of the SOM for estimating texture labels. Experimental
results were shown on synthetic and natural images.

Raghu and Yegnanarayana [94] presented a Bayesian approach for supervised segmentation of texture images that is based on the Gaussian random process and MRF models. The texture features are extracted using a set of Gabor filters with different frequencies, orientations, and bandwidths, and are modelled as Gaussian distribution by means of feature formation process. The neural network model, which is an extension of a Hopfield network to three dimension, is used to find the segmentation state using the MAP criteria.

Goussard et al [42] presented an unsupervised segmentation method based upon a discrete level unilateral Markov random field model of the image. They introduced parsimonious telegraphic parameterization of the unilateral Markov field which reduces the computational complexity of the algorithm used in the segmentation and the parameter estimation stages. The proposed method was tested on simulated and real images, under the assumption that the noise distribution is Gaussian.

Krishnamachari and Chellappa [69] proposed multiresolution models for Gauss-Markov random fields (GMRF) with application to texture segmentation. They discussed two techniques to estimate the GMRF parameters at coarser resolution from the fine resolution parameters, one by minimizing the Kullback-Leibler distance and another based on local conditional distribution invariance. Different texture regions in an image were modelled by GMRF’s and the associated parameters were assumed to be known. They used Iterated conditional mode (ICM) [8] minimization at all resolutions. They concluded that the multiresolution technique performs better than the single resolution approach and the technique can be extended to perform unsupervised texture segmentation.

Andrey and Terroux [5] proposed a selectionist relaxation algorithm as a new method for segmentating images that contain textures modelled using Markov random fields. The generalized Isling model was used to represent textured data. Experimental results on images containing various synthetic and natural textures were presented.
They concluded that the selectionist relaxation does not rely on parameters estimated on the image blocks, the reliability of which degrades as the number of texture increases and thus it is suitable for unsupervised segmentation.

Langan et al [71] investigated the cluster validation problem for unsupervised stochastic model based image segmentation. They modelled an image as a doubly stochastic field in which the state or region map and the intensity data are modelled as random fields. They proposed a model fitting technique in which the complete data log-likelihood functional is modelled as an exponential function. The estimated number of classes are then determined in a manner similar to finding the rise time of the exponential function.

Eom [33] presented a feature extraction method for texture segmentation which is based on 2-D moving average (MA) modelling approach. The 2-D MA modelling approach characterizes a texture over a large neighbourhood and therefore textures with long-correlation characteristics are well represented by a 2-D MA model. The maximum likelihood (ML) estimators of the 2-D MA model are used as texture features. A neural network classifier is used for supervised segmentation while a fuzzy clustering algorithm is used for unsupervised segmentation. He presented the experimental results on both monochrome and colour images. He concluded that in comparison with grey level co-occurrence matrix (GLCM) features [48], the MA model features perform better when the textures have long correlations.

### 2.6 Spatial/Spatial-Frequency Representation

Strong evidence was given by Gagalowicz [36] and by Julesz and Bergen [65] that visual discrimination is a local process. The poor performance of earlier frequency analysis method which are global in nature [24] [120] was thus justified. Joint spatial/spatial-frequency methods are based on image representations that indicate the frequency contents in localized regions in the spatial domain, thereby eliminat-
2.6. Spatial/Spatial-Frequency Representation

The shortcomings of the Fourier based techniques. These methods can achieve high resolution in both the spatial and spatial frequency domains and are consistent with recent theories on human vision [100]. We describe here few spatial/spatial-frequency methods.

2.6.1 The Spectrogram

The 1-d Fourier transform, which is used in the analysis of time-varying signals [4], when extended to two dimensions to yield the finite support Fourier transform, takes the form:

\[
F_{xy}(u, v) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f_{xy}(x', y') e^{-jux'} e^{-jvy'} dx' dy' \quad (2.23)
\]

where

\[
f_{xy}(x', y') = f(x', y') h(x - x', y - y') \quad (2.24)
\]

with \(f(x', y')\) being the original image, and \(h(x - x', y - y')\) a window centered at \((x, y)\).

The spectrogram of the image is simply the square magnitude of \(F_{xy}(u, v)\), given by [100]

\[
S_f(x, y, u, v) = |F_{xy}(u, v)|^2. \quad (2.25)
\]

The Fourier power spectrum of an image is sensitive to the texture coarseness. A coarse texture will have high values concentrated near the low frequency part of the spectrum while a fine texture will have values that are more spread out across the
Chapter 2. Overview of Texture Segmentation Techniques

The use of DOG filters as generating filters has been popular due to the correlation between this response and the measured receptive fields of both retinal ganglion and lateral geniculate cells [100].

2.6.3 The Wigner Distribution

The Wigner distribution (WD) is a spatial/spatial frequency representation [101] which was first introduced in its 1D form in quantum mechanics, to characterize
the positions and momenta of particles. A property of the 2-D Wigner distribution which is of particular interest in image processing is that it is a strictly real valued function. The 2D pseudo-Wigner distribution (PWD), a discrete approximation of the continuous WD, was first used for texture segmentation by Reed and Wechsler [100] [99].

The 1-D pseudo-Wigner distribution (PWD) can be defined as:

\[
P_{W}(t, \lambda) = 2 \sum_{k=-N+1}^{N-1} e^{-j2\lambda k} |h_N(k)|^2 \sum_{l=-M+1}^{M-1} g_M(l)x(t+i+k)x^*(t+l-k) (2.28)
\]

where \( h_N(k) \), \( g_M(l) \) are window functions and \( * \) denotes complex conjugation.

The 2-D PWD is an extension of above 1-D definition and is given by

\[
P_{W}(m,n,p,q) = \frac{4}{P} \sum_{k=-N+1}^{N-1} \sum_{l=-N+1}^{N-1} h_{N_1,N_2}(k,l) \sum_{r=-M+1}^{M-1} \sum_{s=-M_1+1}^{M_1-1} g_{M_1,M_2}(r,s)
\]

\[
f(m+r+k,n+s+l)f^*(m+r-k,n+s-l)e^{-j(2\pi kp/P+2\pi lq/Q)}
\]

where,

\( p = 0, \pm 1, \ldots, \pm (N_2 - 1) \),
\( q = 0, \pm 1, \ldots, \pm (N_1 - 1) \),
\( P = 2N_2 - 1 \)
\( Q = 2N_1 - 1 \)

and \( m,n \) are integers. \( N_1 \times N_2 \) and \( M_1 \times M_2 \) represent the size of two windows used in the functions \( h_{N_1,N_2}(k,l) \) and \( g_{M_1,M_2}(r,s) \). The size of window function \( h_{N_1,N_2}(k,l) \) is dictated by the resolution required in the spatial-frequency domain. Reed and Wechsler used a 2-D extension of 1-D Kaiser Window for \( h_{N_1,N_2}(k,l) \). The function of the window \( g_{M_1,M_2}(r,s) \) is to allow local averaging. A normed rectangular window was chosen \( (M_1 \neq M_2) \). Reed and Wechsler [100] used the
relaxation process for boundary identification between homogeneous regions which consist of two steps.

a. Averaging:

\[ A^{(i)}(x,y) = \sum_{m=-L}^{L} \sum_{n=-L}^{L} \frac{1}{(2L+1)^2} I^{(i-1)}(x+m, y+n) \]  

(2.30)

b. Transformation:

\[ I^{(i)} = L(A^{(i)}(x,y)) = \frac{1 - e^{-\left\{ A^{(i)}(x,y) - \alpha \right\}/\beta}}{1 + e^{-\left\{ A^{(i)}(x,y) - \alpha \right\}/\beta}} \]  

(2.31)

where,

\( I^{(i)}(x,y) \) is the 2-D array at \( i^{th} \) iteration. Relaxation is stopped when the 2-D array changes by only a small amount [14].

A key issue in comparing joint spatial/spatial-frequency representations is the resolution that can be attained (simultaneously) in the two domains [122]. Daugman examined the class of Gabor filters, described in the next section, and found that they achieve the lower limit of resolution/uncertainty as measured by the product of effective widths corresponding to the spatial and spectral domains, respectively. Specifically, the resolution/uncertainty is measured separately along each dimension (i.e. the \( s \) and \( sf \)) of the joint \( s/sf \) representation. Jacobson and Wechsler [57] further examined the resolution/uncertainty issue and concluded that

1. For joint \( s/sf \) representations, resolution should be derived from the joint cartesian domain \( (s \times sf) \) rather than computing over two independent dimensions (i.e. \( s \) and \( sf \)).

2. The spectrogram, the DOG, and the Gabor power representations are smoothed versions of the WD, and they are all members of the more general Cohen class of distributions.
2.6. Spatial/Spatial-Frequency Representation

Reed and Wechsler [100] concluded that the above-mentioned representations cannot improve on resolution what can be achieved by the WD. Another advantage of using the WD over the above-mentioned representations is that it encodes phase information which is critical for cases in which textures differ only in phase.

Cristobal and Hormigo [25] proposed a method for texture segmentation based on the use of texture feature detectors derived from a decorrelation procedure of a modified version of a Pseudo-Wigner distribution (PWD). The decorrelation procedure was accomplished by a cascade recursive least squared (CRLS) principal component (PC) neural network. They were aimed at obtaining a more efficient analysis of images by combining the advantages of using a high resolution joint representation given by PWD with an effective adaptive principal component analysis (PCA) through the use of feedforward neural nets. They concluded that the PWD-PCA method provides excellent segmentation results for textures with a high degree of homogeneity. For textures with low degree of homogeneity, the performance of the proposed method can be improved by increasing the spatial window analysis through increasing the computational cost.

2.6.4 Wavelet Analysis

Porter and Canagarajah [92] proposed a scheme that automatically selects the optimal features for each pixel using wavelet analysis. They argued that the widely used K-means clustering routine usually requires a threshold in its determination of the optimal number of regions of segmentation. They instead suggested a completely automatic method for true cluster number estimation using the second derivative of the within-cluster distances which does not require any threshold settings.

Laine and Fan [70] introduced a method of feature extraction for texture segmentation that relies on multichannel wavelet frames and 2-D envelope detection. A comparative study of two algorithms based on the Hilbert transform and Zero crossings for feature extraction was given. They presented experimental results for both
natural and synthetic textures.

Barni and Mecocci [10] proposed a wavelet-based fuzzy clustering algorithm which is suitable for remote sensing textured images. The algorithm receives as input both the remotely sensed image and a texture image based on a fractal model, derived from the wavelet representation itself. They concluded that the wavelet representation allows large computation time saving and fractal measure improves the performance of the texture segmentation algorithm.

### 2.6.5 Multi-Channel Filtering Approach

The multi channel filtering approach for texture segmentation is inspired by a multi-channel filtering theory for processing visual information in the early stages of the human visual system. This particular theory, first proposed by Campbell and Robson [15], holds that the visual system decomposes the retinal image into a number of filtered images, each of which contains intensity variations over a narrow range of frequency and orientation. Psychophysical experiments [15][113] suggested that there are mechanisms in the visual cortex of mammals that are tuned to combinations of frequency and orientation in a narrow range. These mechanisms are referred to as channels and are interpreted as band-pass filters. This multi-channel filtering approach is intuitively appealing because it allows us to exploit differences in dominant sizes and orientation of different textures. Another important advantage of the multi-channel filtering approach to texture analysis is that one can use simple statistics of grey values in the filtered images as texture features [58]. This is because of decomposing the original image into several filtered images with limited spectral information.

Thus the main issues involved in multi-channel filtering approach to texture segmentation are [58]:

- Functional characterization of channels and the number of channels.
2.6. Spatial/Spatial-Frequency Representation

- Extraction of appropriate texture features from the filtered images.
- The relationship between channels.
- Integration of texture features from different channels to produce a segmentation.

In the earlier techniques, the channels were represented by a set of spatial filters with frequency-and/or orientation properties. One of the earlier techniques was by Faugeras [34] who used a set of bandpass filters which had both frequency and orientation selective properties. Recent techniques have used filters that are obtained by fitting band limited functions to the receptive field profiles of simple cells in the visual cortex of some mammals. Malik and Perona [76] used Gaussian derivative models (radially symmetric difference of Gaussians (DOG) and directionally tuned difference of offset Gaussians (DOOG)) filters. Jain and Farrokhina [58] presented a multi-channel filtering technique which uses a bank of even-symmetric Gabor filters to characterize the channels. They represented the channels with a bank of real-valued, even symmetric Gabor filters. Gabor functions have been shown to be good fits to the receptive field profile of simple cells in the striate cortex. Back et al [56] have shown correlation between the ability of humans to segment tripartite textured images and the outputs of a bank of 2-D Gabor filters applied to the images. The impulse response of an even symmetric Gabor filter is given by

\[ h(x,y) = \exp\left\{-\frac{1}{2}\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right\} \cos(2\pi u_0 x) \]

(2.32)

where \( u_0 \) is the frequency of a sinusoidal plane wave along the x-axis, and \( \sigma_x \) and \( \sigma_y \) are the space constants of the Gaussian envelope along the x and y axes, respectively.

The Fourier domain representation of the above equation is given by
Chapter 2. Overview of Texture Segmentation Techniques

\[ H(u, v) = A \left( \exp \left\{ -\frac{1}{2} \left[ \frac{(u-u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2} \right] \right\} + \exp \left\{ -\frac{1}{2} \left[ \frac{(u+u_0)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2} \right] \right\} \right) \]

where,

\[ \sigma_u = 1/(2\pi\sigma_x), \quad \sigma_v = 1/(2\pi\sigma_y), \quad \text{and} \quad A = 2\pi\sigma_x\sigma_y \]

The Fourier domain representation specifies the amount by which the filter modifies or modulates each frequency component of the input image. Such representations are referred to as modular transfer functions (MTF) [58].

Texture segmentation requires simultaneous measurements in both the spatial and the spatial-frequency domain. Filters with smaller bandwidths in the spatial-frequency domain are more desirable because they allow us to make finer distinctions among different textures. On the other hand, accurate localization of texture boundaries requires filters that are localized in the spatial domain. The most important property of Gabor filters is that they have optimal joint localization in both the spatial and the spatial-frequency domain [27].

Jain et al [58] suggested a filter selection scheme based on reconstruction of the input image from the filtered images. Each selected filtered image was subjected to a non-linear transformation that behaves as a blob detector. The texture discrimination is associated with differences in the attributes of these blobs in different regions. A statistical approach was then used where attributes of blobs are captured by texture features defined by a measure of energy in a small window around each pixel in each response image. The total number of filters selected were based on their energy such that \( R^2 > 0.95 \) where \( R^2 \) is the fraction of intensity variations in the reconstructed input image obtained by adding all the filtered images. A square error clustering algorithm was used to identify the texture categories and modified Hubert statistics was used as a relative index to estimate the true number of texture categories.
2.6. Spatial/Spatial-Frequency Representation

Teuner et al [108] highlighted the drawback of this approach of multichannel Gabor decomposition, namely that when used for unsupervised image analysis tasks, imposes excessive storage requirements due to the nonorthogonality of the basis functions. They proposed a texture segmentation approach based on the use of selected tuned Gabor filters in which a pyramidal Gabor transform forms the basis for an unsupervised filter selection. A tuning algorithm was used to choose and control the dyadic Gabor filters appropriate for feature extraction. The pyramid yields different resolution at different levels due to different sizes of Gabor cells at these levels. This yielded an optimum segmentation of textured images without a priori knowledge of the image.

Dennis et al [31] proposed an approach in which Gabor filter outputs are modelled as Rician random variables and a decision-theoretic model was developed for selecting optimal filter parameters. The basic assumption in their proposed algorithm was that the image contains two textures of interest and the prototype samples of the desired textures are given a priori. For robust segmentation of difficult texture pairs, a multiple-filter segmentation scheme was proposed.

Weldon and Higgins [117] addressed the design of multiple Gabor filters for segmentating multi-textured images. The segmentation scheme is based on the development of a mathematical model that relates the texture power spectra, filter parameters, and segmentation errors.

Jain and Karu [59] proposed a neural network texture classification method as a generalization of the multi channel filtering method. Instead of using a general filter bank, a neural network is trained to a minimal set of specific filters, so that both the feature extraction and classification task are performed by the same network. They applied this technique for locating barcodes in the images and segmentating a printed page into text, graphics and background. They concluded that compared to the traditional multi channel filtering method, the neural network approach performs the same texture segmentation or classification task more efficiently.
Pichler et al [89] presented an unsupervised texture segmentation algorithm based on feature extraction using multichannel Gabor filtering. They showed that feature contrast, a criteria derived for Gabor filter parametric selection, is well suited for feature coordinate weighing in order to reduce the feature space dimension. The central idea of the proposed algorithm was to cut the actual segmented image into disjoint “Scrap” images and use them after low pass filtering as additional features for repeated K-means clustering and minimum distance classification.

Recent research has concentrated in finding higher order Gabor features but the fundamental problem of these higher order Gabor features is the fact that elongated regions are often detected on texture boundaries if a region based algorithm is applied. This effect is due to sensitivity of some parameters to the mixture of two power spectra at a boundary. This effect can be eliminated to some extent by a boundary based method in Gabor space.

There are few methods which cannot be neatly classified in any of the above classes. We describe in brief such methods here.

Hansen and Higgins [47] proposed two methods of supervised image segmentation: Supervised relaxation labelling and Watershed-driven relaxation labelling. These methods were shown to be particularly well suited to problems in 3D medical image analysis, where the images are large, the regions are topologically complex, and the tolerance of errors is low. They showed that the supervised relaxation labelling which operates on a pointwise basis, tends to be noise resistant, but computationally very expensive. The watershed driven relaxation labelling, exploits the computational efficiency of watershed analysis and the noise resistance of relaxation labelling.

Shafarenko et al [103] proposed an algorithm for segmentation of colour images, which takes into account the noise that is inevitably present during the image acquisition. They used clustering method based on the morphological Watershed transform performed on the 3-D colour histogram. They concluded that the algorithm is
highly suitable for automatic colour segmentation.

Jansing et al [61] proposed a method of 2-D entropic segmentation using a linear discriminant function. This segmentation method automatically highlights desired objects against background (with no user input or parameter specification) using the information from a 2-D frequency distribution of the chosen image features. They demonstrated the robustness of the method by deriving the data from an aerial image in which the object data set is the cultural objects present, and an MRI image of a cross-section of brain where the ventricles represent the objects of interest. They concluded that this method is comparable to a Bayesian classifier for two-class segmentation, while no a priori information about the desired objects was needed.

2.7 Conclusion

In this chapter, I described a few texture segmentation techniques. Texture segmentation was considered as a two stage problem of feature extraction and segmentation. Feature extraction techniques were mainly based on some unique features (which included operator based, statistical, and transform domain features), model based or structural methods. A few algorithms which use unique segmentation approach which includes region based and edge based approaches or hybrid of the two, were discussed. I discussed mainly two types of model based approaches: Fractal models and Random field models. I also discussed in brief various random field models used for supervised and unsupervised texture segmentation. I discussed structural approaches which are effective for specific class of textures. Finally, I described various approaches based on spatial/spatial-frequency representations with emphasis on Wigner distribution and Gabor filtering. Few techniques which do not clearly come under any of the above categories are also discussed.

From all the presented methods one can see that the problem of texture segmentation
is still an open problem and there is plenty of scope for further research in this field.
Chapter 3

Boundary Detection as a Global Optimization Problem

3.1 Introduction

Image segmentation is a very important task in many image analysis or computer vision applications. Any image analysis system for image segmentation starts by representing observed data (grey values) in terms of unobserved (label) variables. These label variables are the intermediate data structures for further analysis. The global optimization approaches in general are quite popular in the field of image segmentation.

In this chapter first I shall introduce global optimization, its merits, and its implementation using the technique of simulated annealing. In section 3.3, I explore the algorithm proposed by Geman et al [39] for the detection of texture boundaries by global optimization. According to this approach, the interaction between the data and the label model is represented by an energy function which involves two components: One expresses the interaction between the data and the labels while the other encodes constraints derived from the general information or expectations about label patterns [39]. The energy function also incorporates a disparity measure between certain special features (Range and Directional residuals) of pairs of blocks of pixel grey levels. The optimization of the cost (energy) function is done using
simulated annealing [102] [40] [84].

In section 3.4, I introduce a modified cost function designed to correct some of the shortcomings of the cost function used by Geman et al. Geman et al [39] measured the disparity between two blocks of data with the help of the Kolmogorov-Smirnov statistic. In section 3.5, I explore other options. In particular, I explore the possibility of using the linear correlation coefficient, the $\chi^2$ test and the Contraharmonic filter in order to quantify the disparity between two blocks of pixels. Based on the observation that different statistical measures work well for different textures, I propose a combination of all these measures to achieve best results.

Geman et al [39] introduced the explicit dictionary of various combinations of labels which are illegal patterns for real world images and which are penalized during optimization. On the other hand, the behaviour of the physical edges which is governed by the connectivity on a pixel/label lattice can be encapsulated in a dictionary which represents all possible combinations of local arrangements arising in real images. So, in section 3.6, I introduce the concept of a virtual dictionary: Any label pattern not part of this explicit dictionary is treated as an illegal pattern and it is therefore penalized during optimization. This way, instead of using a cost function which simply discourages certain label configurations, I use one which actively encourages the legal patterns.

In section 3.7, I introduce the concept of a database for finding a common set of normalizing constants.

In section 3.8, I quantify various results by estimating under/over segmentation errors. I summarize in section 3.9.

### 3.2 Global Optimization: An Overview

Many tasks in computer vision and image analysis have been expressed as global optimization problems [12] [39] in which the general issue is to find the global
minimum of an objective (energy) function which describes the interaction between the different variables modelling the image features in a given problem. The generally chosen variables are the observation variables which represent the observed data and the hidden variables (labels) which form the desired representation to be extracted from the original image. The energy function expresses the interaction between the observed data and the hidden labels and also includes some form of constraints for achieving the desired solution. The energy function is decomposed as a sum of local interaction functions defined on a neighbourhood and standard regularization approaches as well as MRF-based image analysis lead to minimization of such global energy functions. The global energy function is a powerful tool for specifying nonlinear interactions between different image features thereby combining and organizing spatial and temporal information by introducing strong generic knowledge about the features to be estimated. Few examples of such global energy functions are in [40] for image restoration, [39] for boundary estimation, [7] for stereovision, [84] for visual motion analysis and in [83] for scene interpretation.

The minimization of such a global energy is often a difficult problem since the number of possible label configurations is very large and the global energy function may have many local minima. The deterministic relaxation algorithms such as Iterative conditional mode (ICM) [8], highest confidence first (HCF) [21] [81] and graduated nonconvexity (GNC) [12] can only be used for minimization of these global energy functions when there is a good initial guess available. The deterministic approaches often converge to configurations corresponding to a local minimum of the global energy function. This problem can be overcome to a large extent by using a stochastic relaxation algorithm (Simulated annealing) or by using multigrid methods [51]. Here I concentrate on the simulated annealing approach for achieving optimal solutions.

Simulated annealing is a new approach to the approximate solution of difficult combinatorial optimization problems. It was originally proposed by Kirkpatric et al
[102], who reported promising results based on sketchy experiments. For practical purposes, it has effectively solved the famous travelling salesman problem of finding the shortest cyclical itinerary for a travelling salesman who must visit each of \( N \) cities in turn. The method has also been used successfully for designing complex integrated circuits: The arrangement of several hundred thousand circuit elements on a tiny silicon substrate is optimized so as to minimize interference among their connecting wires [114]. The main difference between a local optimal approach and the simulated annealing approach is that:

A local optimal solution starts with an initial solution generated by some means, and repeatedly attempts to find a better solution by moving to a neighbour with lower cost, until it reaches a solution none of whose neighbours have a lower cost. Such a solution is a locally optimal. Simulated annealing is motivated by the desire to avoid getting trapped in poor local optima, and hence, it occasionally allows “uphill moves” to solutions of higher cost under the guidance of a control parameter called the temperature.

Geman and Geman [40] were the first who proposed the use of this technique for the Bayesian restoration of images. Later this optimization technique was applied to many applications. A few of them are: binary image restoration [123], source coding [38], scene segmentation from visual motion [84], edge detection [73], image reconstruction [2], graph colouring and number partitioning [63], curve detection [107] and image segmentation [45] [115].

Now I describe the mathematical framework required for the implementation of such a stochastic algorithm.

### 3.2.1 Markov Random Fields and Gibbs Distributions

Let \((S, \mathcal{R})\) denote an arbitrary graph. Let \( X = \{X_s, s \in S\} \) denote any family of random variables indexed by \( s \) and \( \mathcal{R} \) denote a neighbourhood. Assume a common state space, say \( \Lambda = \{0, 1, 2, \ldots, L' - 1\} \) so that \( X_s \in \Lambda \) for all \( s \) and \( L' \) is finite.
Let $\Omega$ be the set of all possible configurations, which can be expressed as

$$\Omega = \{x = (x_1, \ldots, x_N) : x_i \in \Lambda, 1 \leq i \leq N\}$$

where $N = m^2$ is any ordering of lattice points.

We assume that all configurations in $\Omega$ are possible, i.e.

$$P(X = x) > 0, \forall x \in \Omega$$

(3.1)

where $P(X = x)$ is the probability of configuration $x$ to arise.

Now, $X$ is a Markov Random Field with respect to $\mathcal{R}$ if

$$P(X_s = x_s \mid X_r = x_r, \forall r \neq s) = P(X_s = x_s \mid X_r = x_r, r \in \mathcal{R}_s)$$

(3.2)

for every $s \in S$ and $(x_1, \ldots, x_n) \in \Omega$. $\mathcal{R}_s$ is the neighbourhood system with respect to $s$. The function on the right hand side is called the local characteristic of the Markov Random Field (MRF).

A Gibbs distribution relative to $\{S, \mathcal{R}\}$ is a probability measure $\pi$ on $\Omega$ with the following representation.

$$\pi(x) = \frac{1}{Z} e^{-\frac{U(x)}{T}}$$

(3.3)

where $Z$ is a normalizing constant called the partition function and $T$ is a constant parameter called temperature.

The equivalence between the Markov random fields and the Gibbs distributions can be described by the following statement[7]:

3.2. Global Optimization: An Overview

49
Chapter 3. Boundary Detection as a Global Optimization Problem

$X$ is an MRF with respect to $\mathcal{R}$ if and only if $P(X = x)$ is a Gibbs distribution with respect to $\mathcal{R}$ i.e.

$$P(X = x) = \pi(x)$$  \hspace{1cm} (3.4)

It says that it is the same thing either we model $X$ as an MRF or we assume that the probability law of $X$ is a Gibbs distribution.

### 3.2.2 The Bayesian Framework

In most problems of image analysis, incorporation of prior knowledge is important for making inference based on the images. In the recent years there has been an increasing interest in use of statistical techniques for modelling and processing image data. In the Bayesian approach of statistical inference, the goal is to make proper use of the prior information that is available. Use of this approach in image analysis is quite popular. Besag [9] defined the Bayesian paradigm to consist of four successive stages:

- **Construction of a prior probability distribution** $\pi(x)$ where $x$ is to be reconstructed. In my case this is the assignment of boundary labels to all label sites.

- **Combine the observed image $g$ with the underlying labels $x$ through a conditional probability density** $\pi(g \mid x)$.

- **Construct the posterior density** $\pi(x \mid g)$ from $\pi(x)$ and $\pi(g \mid x)$ by Bayes theorem giving

$$\pi(x \mid g) \propto \pi(x) \pi(g \mid x).$$  \hspace{1cm} (3.5)

- **Base any inference about the boundary labels $x$ on the posterior distribution** $\pi(x \mid g)$.
3.2. Global Optimization: An Overview

In the Bayesian analysis, all kinds of inferences are made from \( \pi(x \mid g) \). Maximum a posteriori (MAP) is one such choice of inference which looks for that \( x \) which maximizes the probability \( \pi(x \mid g) \).

Thus the general form of the annealing algorithm can be described as follows:

1) Set \( k = 0 \) where \( k \) is the iteration number. We shall index the family of random variables \( X \) by \( k \), to give it a dynamic meaning that will allow us to describe its evolution from one iteration step to the next. We shall also index the temperature parameter \( T \) by \( k \), to allow the flexibility of changing it from one iteration step to the next.

2) Generate an initial state \( X_0 \) with probability \( \pi(X_0 = x) = Q(x) \).

3) Generate a candidate state \( Y_k \) with the conditional probability \( \pi(Y_k = x' \mid X_k = x) = Q(x, x') \) where \( Q(x, x') \) is the transition probability from state \( x \) to \( x' \). This is called the stochastic dynamics of annealing.

4) Let \( X_{k+1} = Y_k \) with probability \( \exp\left(-\frac{[U(x')-U(x)]_+}{T_k}\right) \) and \( X_{k+1} = X_k \) otherwise. Where \( [U(x') - U(x)]_+ = \max(U(x') - U(x), 0) \).

5) Set \( k = k + 1 \), and go to step 3).

The annealing algorithm generates a discrete time Markov chain \( X_k \). Thus we require three things in order to implement the algorithm.

- Specify the probability distribution \( Q(\cdot) \) for generating the initial state.
- Specify the transition distribution \( Q(\cdot, \cdot) \) for generating candidate states.
- A temperature schedule \( T_k \).

According to Geman et al [40], the initial state \( X_0 \) can be selected arbitrarily. In practice it is better to start annealing from good initial guess \( X_0 \) estimated from the data.
There are mainly two stochastic dynamics in the domain of image analysis: Metropolis dynamics [11] and dynamics of Gibbs sampler [40]. We describe here the Gibbs sampler which is often preferred because of its easy programmability and simplicity in implementation.

The Gibbs Sampler dynamics is a Markovian updating scheme for obtaining samples from a joint distribution, via iterated sampling from the full conditional distributions. Let the total configuration of label lattice be $X_k = (X_{s1}(k), X_{s2}(k), \ldots X_{sN}(k))$ and the starting configuration be arbitrary, say, $X_0$. At each instant of time, only one site is updated and therefore $X_{k-1}$ and $X_k$ differ in at most in one coordinate. At time $t$, a sample is drawn from the local characteristic of $\pi$ (see equation 3.2 for $s = n_t$ and $x = X(k-1)$ where $n_t, t=1,2,\ldots$, are the sequence in which sites are visited for updating. Thus we choose a state $x$ from the conditional distribution of $X_{nt}$ given the observed states of the neighbouring sites $X_{r}(k-1)$, $t \in \mathcal{R}_{nt}$. $\mathcal{R}_{nt}$ is again the neighbourhood with respect to $n_t$. In mathematical terms, it can be defined as

$$P(X_{s}(k) = x_s, s \in S) = \pi(X_{nt} = x_{nt} \mid X_s = x_s, s \neq n_t)P(X_s(k-1) = x_s, s \neq n_t) \quad (3.6)$$

where $\pi$ is the Gibbs measure.

For temperature $T$, the following logarithmic cooling schedule [39][40] [84] is followed.

$$T_k = \frac{c_c}{log(1+k)} \quad (3.7)$$

where $T_k$ is the temperature at the $k^{th}$ iteration and $c_c$ is a constant. The value of $c_c$ is determined by experiments.

In simulated annealing, for the first few iterations (30 or 40), the energy of the lattice increases and after that starts decreasing. The iteration process is terminated if for 50 successive iterations the change in energy is less than 2% i.e. if
3.3. The Algorithm of Geman et al

Let $S_I$ be the pixel lattice and $S_B$, the lattice of labels, located at the in between pixel positions. Lattice $S_B$ can be much sparser than $S_I$. This is determined by parameter $\sigma_I$, the label resolution (see Fig. 3.1). I associate label site $s = (i, j) \in S_B$ with the pixel $(i, j) \in S_I$ that is above and on the left of $s$. The label and the data configurations can be represented as follows.

Let

$$x = \{x_s, s \in S_B\},$$

$$g = \{g_s, 1 \leq s \leq m^2\}$$

denote respectively the label assignments and the data values.

$x_s$ is the label at site $s$, $s \in S_B$. Labels are binary representing the presence ($x_s = 1$) or absence ($x_s = 0$) of boundary elements. $g_s$ is the grey level of pixel $s$. The size of the lattice is $m \times m$.

For boundary finding, adjacent sites in $S_B$ define boundary segments, and are associated with pairs of pixel blocks, located across each other with respect to the segment. Figure 3.2(a) shows the pixels $s^*, t^*$ across the corresponding boundary segment $s, t$ while 3.2(b) and 3.2(c) show a pair of pixel blocks $B_{s^*}, B_{t^*}$ across the boundary segment for label resolution $\sigma_I = 3$ and $\sigma_I = 5$ respectively.

The interaction between label assignment $x$ and the set of data $g$ is defined in terms of an energy function:

$$\frac{|U_k - U_{k-1}|}{U_{k-1}} < 0.02 \quad (3.8)$$

where $U_k$ is the total energy at the $k^{th}$ iteration.
$U = U_1(x; g) + \alpha U_2(x)$ \hspace{1cm} (3.9)

where $\alpha$ is a constant.

The first term in this expression is defined as follows:

$$U_1(x; g) \equiv \sum_{i,j} (1 - x_{i,j} x_{i+\sigma_i,j} + \sigma_i)(\phi_{(i,j)}(i+\sigma_i,j)(g) + (1 - x_{i,j} x_{i+\sigma_i,j} + \sigma_i)(\phi_{(i,j)}(i+\sigma_i,j)(g)$$ \hspace{1cm} (3.10)

where $\phi_{(i,j)}(t,m)(g)$, is the measure of disparity between two blocks of pixel data on either side of a segment defined by label sites $x_{i,j}$ and $x_{t,m}$. Geman et al [39] used as disparity measure the Kolmogorov-Smirnov distance between the two histograms constructed from the pixels in the blocks adjacent to the boundary segment.

If I omit the constant terms, the first term of the cost function can be written as:

$$U_1(x; g) = - \sum_{i,j} \{x_{i,j} x_{i+\sigma_i,j} \phi_{(i,j)}(i+\sigma_i,j)(g) + x_{i,j} x_{i+\sigma_i,j} \phi_{(i,j)}(i+\sigma_i,j)(g)\}$$ \hspace{1cm} (3.11)
3.3. The Algorithm of Geman et al

The above term of the cost function promotes boundary maps $x$ which are faithful to the data $g$. Now I describe the “penalty function” $U_2(x)$ of the above cost function which will penalize the label configurations that represent very small structures which normally do not represent boundaries in an image.

The penalized patterns are the combinations of binary values of labels. They correspond to isolated edgels, small structures, cross junctions, and thick boundaries. These combinations of labels, if allowed, inhibit the formation of good boundaries in an image. Therefore these patterns are assigned zero probability when formulating the cost function. Mathematically I can represent the penalty function $U_2(x)$ as follows:

$$U_2(x) = \sum_{i,j} (V_1 + V_2 + V_3 + V_4 + V_5 + V_6 + V_7 + V_8 + V_9)$$

(3.12)

where, $V_1, \ldots, V_9$ are some functions defined as:

$$V_1 = \delta(x_{i,j} - 1)\delta(x_{i+\sigma_i,j})\delta(x_{i,j+\sigma_i})\delta(x_{i,j+\sigma_i+\sigma_i})$$

(3.13)

$$V_2 = \delta(x_{i,j} - 1)\delta(x_{i+\sigma_i,j})\delta(x_{i,j+\sigma_i})\delta(x_{i,j+\sigma_i})\delta(x_{i,j+\sigma_i} - 1)$$

(3.14)

$$V_3 = \delta(x_{i,j} - 1)\delta(x_{i+\sigma_i,j})\delta(x_{i,j+\sigma_i})\delta(x_{i,j+\sigma_i} - 1)\delta(x_{i,j+\sigma_i})$$

(3.15)
Chapter 3. Boundary Detection as a Global Optimization Problem

\[ V_4 \equiv \delta(x_{i,j} - 1)\delta(x_{i+\sigma_1,j})\delta(x_{i-\sigma_1,j} - 1)\delta(x_{i,j-\sigma_1})\delta(x_{i,j+\sigma_1}) \] 

\[ V_5 \equiv \delta(x_{i,j} - 1)\delta(x_{i+\sigma_1,j} - 1)\delta(x_{i-\sigma_1,j})\delta(x_{i,j-\sigma_1})\delta(x_{i,j+\sigma_1}) \] 

\[ V_6 \equiv \delta(x_{i,j})\delta(x_{i+\sigma_1,j} - 1)\delta(x_{i-\sigma_1,j} - 1)\delta(x_{i,j-\sigma_1} - 1)\delta(x_{i,j+\sigma_1} - 1) \] 

\[ V_7 \equiv \delta(x_{i,j} - 1)\delta(x_{i+\sigma_1,j} - 1)\delta(x_{i-\sigma_1,j} - 1)\delta(x_{i,j-\sigma_1} - 1)\delta(x_{i,j+\sigma_1} - 1) \] 

\[ V_8 \equiv \delta(x_{i,j} - 1)\delta(x_{i+\sigma_1,j} - 1)\delta(x_{i-\sigma_1,j} - 1)\delta(x_{i,j+\sigma_1} - 1) \] 

\[ V_9 \equiv \delta(x_{i,j} - 1)\delta(x_{i+2\sigma_1,j} - 1)\delta(x_{i,j-2\sigma_1} - 1)\delta(x_{i,j+2\sigma_1} - 1) \] 

Each one of these functions represents a local label configuration that is unaccept­
able. These unacceptable label configurations constitute the dictionary of penaliz-
able local label combinations and they are shown in figure 3.3. The penalty function 
\[ U_2(x) \] is incremented by 1 each time any of the penalty configurations occurs.

\[ \begin{array}{cccccc}
\circ & \circ & \circ & \circ & \circ & \circ \\
\circ & \circ & \circ & \circ & \circ & \circ \\
\end{array} \]

Figure 3.3: Forbidden combinations of labels.

Since many different textures have nearly identical grey level histograms [52] [60], 
we extract features from the data which involve higher order spatial statistics.

Let us represent the various transformations we perform as functions of the grey 
level values of the image \( g \) i.e. if \( \Gamma_l \) is the \( l^{th} \) transformation then the transformed 
image \( g^{(l)}_T \) is

\[ g^{(l)}_T = \Gamma_l(g) \] 

The features used are

\[ \Gamma_1(i,j) \equiv \max_{(k,l)\in W_s}\{g_{k,l}\} - \min_{(k,l)\in W_s}\{g_{k,l}\} \] 

where \( W_s \) is a window centered at pixel \((i,j)\) and \((k,l)\) is any pixel in this window 
with grey value \( g_{k,l} \). This transformation assigns to each pixel the range of grey 
values in a window around it.
Four extra features $\Gamma_2, \Gamma_3, \Gamma_4, \Gamma_5$, are also used. These are four local directional derivatives of the image:

\[
\begin{align*}
\Gamma_2(i,j) & = |g_{i,j} - 0.5(g_{i-1,j} + g_{i+1,j})| \\
\Gamma_3(i,j) & = |g_{i,j} - 0.5(g_{i,j-1} + g_{i,j+1})| \\
\Gamma_4(i,j) & = |g_{i,j} - 0.5(g_{i-1,j-1} + g_{i+1,j+1})| \\
\Gamma_5(i,j) & = |g_{i,j} - 0.5(g_{i+1,j-1} + g_{i-1,j+1})|
\end{align*}
\] (3.24-3.27)

These five residuals are locally averaged over window $W_s$ yielding the final feature maps of the image: $g_T^{(1)}, g_T^{(2)}, g_T^{(3)}, g_T^{(4)}, g_T^{(5)}$.

The disparity measure is based on these five transformations (features).

Let us set

\[
\phi_{s,t}(g) = \phi(\Delta_{s,t}(g))
\] (3.28)

where $\Delta_{s,t}(g)$ measures the difference in properties between the two blocks of pixels sharing a common boundary segment defined by $s$ and $t$, and $\phi(\omega)$ is some function yet to be defined.

The disparity measure $\Delta_{s,t}(g)$ is calculated by comparing the two histograms corresponding to two data sets which are on either side of the boundary segment $s, t$.

Let us assume that we have two data sets

\[
\begin{align*}
x^{(1)} & = \{x_1^{(1)}, x_2^{(1)}, \ldots, x_{n1}^{(1)}\} \\
x^{(2)} & = \{x_1^{(2)}, x_2^{(2)}, \ldots, x_{n2}^{(2)}\}
\end{align*}
\]

First we have to obtain the histograms from these two data sets.
We find the minimum and maximum grey level values in $x^{(1)}$ and $x^{(2)}$. Let $A_{\text{min}}$ be the minimum grey level value in $x^{(1)}$ and $x^{(2)}$ and $A_{\text{max}}$ be the maximum grey level value in $x^{(1)}$ and $x^{(2)}$.

Let

$$t = \frac{A_{\text{max}} - A_{\text{min}}}{N_b}$$

(3.29)

where, $N_b$ is the number of bins. $t$ determines the range of grey level values that contribute to a bin.

Let $B_i$ represent the frequency of occurrence of grey level values in $x^{(1)}$ which lie in the range from $[A_{\text{min}} + (i-1)t, A_{\text{min}} + it)$ and $b_i$ represent the frequency of occurrence of grey level values in $x^{(2)}$ in the same range.

Thus $B_i$ and $b_i$ for $i = 1, \ldots, N_b$ represent the two histograms. Once we have two histograms, we can compare them and find their difference by using the Kolmogorov-Smirnov measure.

This statistic compares two cumulative distribution functions corresponding to two data sets and assigns the maximum absolute difference between these two cumulative distribution functions as their Kolmogorov-Smirnov (K-S) distance.

Let $B^c_j$ denote the cumulative frequency of occurrence of event $B_i$ then

$$B^c_j = \sum_{i \leq j} B_i$$

(3.30)

where, $i, j$ vary from 1 to $N_b$. Similarly we can define $b^c_j$.

Thus, we define the Kolmogorov-Smirnov distance ($D$) as
3.3. The Algorithm of Geman et al

\[
D = \frac{1}{k_b} \max_j | (B_j^c - b_j^c) | \tag{3.31}
\]

where, \( k_b = \sum_i B_i \), and \( i, j = 1, \ldots, N_b \).

A high value of Kolmogorov-Smirnov distance provides a test statistic for the hypothesis that the two data sets \( x^{(1)} \) and \( x^{(2)} \) are the samples from different underlying probability distributions [3]. This distance is assigned to the disparity measure \( \Delta_{st}(g) \).

Each disparity measure computed for the pixels of an image is normalized by division with a threshold. This threshold is selected such that it distinguishes between boundary encouraging differences from non boundary encouraging differences. Thus if the normalized disparity value \( \Delta_{st}(g) \) at any boundary segment \( s, t \) is greater than 1, then the label of the segment leads to boundary formation at that position, otherwise no boundary is formed.

Geman et al [39] calculated the various normalizing constants as follows:

1. For each feature \( i \) and for each texture \( c \) expected to be found in the texture mosaics, they computed the histograms of combined horizontal and vertical Kolmogorov-Smirnov distances between blocks that belong to the same texture. Then they computed the normalizing constant for feature \( i \) and texture \( c \) as \( \lambda_{i,c} = 100(1 - \gamma) \) percentile of that histogram. The value of \( \gamma \) was selected in the range of 0.01 to 0.03, thus keeping 97% to 99% of the K-S distances below \( \lambda_{i,c} \) in each texture type \( c \).

2. They choose the maximum value of \( \lambda_{i,c} \) over all textures to be the normalizing constant of feature \( i \), i.e:

\[
\lambda_i = \max_c \{ \lambda_{i,c} \} \tag{3.32}
\]

where, \( 1 \leq i \leq 5 \) and \( 1 \leq c \leq C \), where \( C \) is the total number of textures in the training set.
Function $\phi(\omega)$, used in equation 3.28, has to be chosen in such a way that when the normalized disparity is less than 1 the formation of a boundary is discouraged, while if it is more than 1 the formation of boundary is encouraged. Geman et al [39] proposed the following function:

$$
\phi(\omega) = \begin{cases} 
- (\omega - 1)^2, & \text{for } 0 \leq \omega \leq 1 \\ 
(\omega - 1)^2, & \text{for } 1 < \omega 
\end{cases}
$$

(3.33)

A plot of this function is shown in figure 3.4.

![Figure 3.4: Function $\phi(\omega)$ used in equation 3.28.](image)

### 3.3.1 Stochastic Formulation

The prior probability distribution given in equation 3.3 is modified when constraints are imposed and can be expressed as:

$$
\pi(x) = \frac{1}{Z} \delta(u_2=0)(x) \exp \left\{ - \frac{U(x)}{T} \right\}
$$

(3.34)
3.3. The Algorithm of Geman et al

where $Z$ is a normalizing constants called partition function.

The constraints place infinite energy barriers in the "energy landscape". They assumed that the grey levels are observed (uncorrupted) whereas the labels are only observed. $\pi(g \mid x)$ is singular and thus the posterior distribution is defined as

$$
\pi(x \mid g) = \frac{1}{Z_1} \delta(u_2=0)(x) \exp\{-\frac{U(x,g)}{T}\}
$$

(3.35)

where $Z_1$ is some constant.

The MAP estimate of the problem is made which maximizes the posterior probability density $\pi(x \mid g)$ thereby minimizing the global cost function. The method of simulated annealing [39] [102] [16] was used to optimize the global cost function.

3.3.2 Results and Discussion

To test Geman et al’s algorithm I used six collages of texture images from the Brodatz database [13] and some ceramic tiles images. The segmentation results with Geman et al’s [39] approach as described above are given in figure 3.5. The disparity measure is based on the Kolmogorov-Smirnov statistic. The range and the 4 directional residuals are extracted as features. The range for each pixel is calculated over a window of $7 \times 7$ centered at that pixel. The four residuals are derived using equations 3.24-3.27. The final five feature maps of the images are obtained by locally averaging these residuals over a window $W_s$ of $7 \times 7$. These five feature images are used for all disparity calculations. I have taken $B_s^*$ and $B_t^*$ (see Fig. 3.2) as $21 \times 21$ blocks of pixels for the Kolmogorov-Smirnov statistic. The two blocks of data are used to find the disparity value by comparing their corresponding histograms over 8 bins. Label resolution $\sigma_l$ is taken as 5. All the images used are $256 \times 256$ pixels in size.

For all my experiments, I scanned the input image in raster format and used asynchronous mode of updating when applying the simulated annealing optimizing method.
Chapter 3. Boundary Detection as a Global Optimization Problem

Figure 3.5: Textural boundary estimation in images. Results with the original cost function and a linear schedule for $\alpha$ with $\Delta\alpha = 0.02$. The Kolmogorov-Smirnov statistic is used as disparity measure.

[102] [82] [73] [11] as discussed in an earlier section. A linear schedule is used for imposing the constraints on the cost function. In the linear schedule $\alpha$, (see equation 3.9) is zero during the first iteration of annealing and then it is increased in steps of $\Delta\alpha$ at every iteration. I choose $\Delta\alpha = 0.02$ and $c_c$ as 0.01. It was found experimentally that value of $c_c$ in the range 0.01 to 0.1 works well. The maximum number of iterations of simulated annealing required for the convergence was about 300.

Figure 3.6 shows the plots of individual terms of the energy function of equation 3.9 against the number of iterations of simulated annealing for the result of figure 3.5(b). The disparity measure is based on the Kolmogorov-Smirnov statistic.

The experimental results show that the said algorithm is rather non-robust. Very fine tuning of the parameters was necessary for obtaining acceptable results with different images. From this preliminary set of experiments the following problems were identified:
3.3. The Algorithm of Geman et al

The energy plots for the Kolmogorov-Smirnov statistic with the explicit dictionary

Figure 3.6: The energy function and its components as a function of the number of iterations of simulated annealing for results of figure 3.5(b). Linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$ is used.

- The cost function of equation 3.10 is not appropriate. The cost function does not appear to express correctly faithfulness to the data. This will become clearer in section 3.4. The function $\phi$ as defined by equation 3.33 is unbounded. This may cause problems in the value of the cost function.

- The selection of normalizing constants is not appropriate as I found that for many composite textures this way of selection does not work well and a lot of boundary information is lost. The selection of normalization constants is highly dependent on the type of composite texture to be segmentated.

- The Kolmogorov-Smirnov statistic alone is not adequate in distinguishing all pairs of neighbouring textures.

- The dictionary of penalizable label configurations of figure 3.3 did not seem adequate. It was felt that too many “bad” quality local label configurations were let to slip through.
• The training of the algorithm was done only on the textures that were known a priori to be present in the composite image. It is rather unrealistic to expect that one will have such prior knowledge for every image.

In the next sections I offer solutions to these problems.

3.4 The Modified Cost Function

The first term, $U_1$, of the cost function must express faithfulness to the data. So, if the disparity measure indicates the presence of a boundary between positions $(i, j)$ and $(i + \sigma_1, j)$, the labels we give in these two sites must comply with this. Overall, the faithfulness to the data is expected to be in the least square error sense. So, for the cost function we propose the following form:

$$U = U_{mo} + \alpha U_2$$

(3.36)

where $U_{mo}$ is given by:

$$U_{mo} = \sum_{i,j} \left\{ (x_{i,j}x_{i+\sigma_1,j} - \Phi_{(i,j)(i+\sigma_1,j)}(g))^2 + (x_{i,j}x_{i,j+\sigma_1} - \Phi_{(i,j)(i,j+\sigma_1)}(g))^2 \right\}$$

(3.37)

on expanding the above expression, we get

$$U_{mo} = \sum_{i,j} \left\{ x_{i,j}^2 x_{i+\sigma_1,j}^2 + \Phi_{(i,j)(i+\sigma_1,j)}^2(g) - 2x_{i,j}x_{i+\sigma_1,j}\Phi_{(i,j)(i+\sigma_1,j)}(g) + x_{i,j}^2 x_{i,j+\sigma_1}^2 + \Phi_{(i,j)(i,j+\sigma_1)}^2(g) - 2x_{i,j}x_{i,j+\sigma_1}\Phi_{(i,j)(i,j+\sigma_1)}(g) \right\}$$

(3.38)

The terms $\sum_{i,j} \Phi_{(i,j)(i+\sigma_1,j)}^2(g)$ and $\sum_{i,j} \Phi_{(i,j)(i,j+\sigma_1)}^2(g)$ are the data terms and are constants. These two terms being constants do not play any role in optimization and
3.4. The Modified Cost Function

hence they can be eliminated from the above cost function. Thus the above expression reduces to:

$$U_{mo} = \sum_{i,j} \left\{ x_{i,j}^2 x_{i+1,j}^2 - 2x_{i,j} x_{i+1,j} (i+\sigma_i,j) (g) + x_{i,j}^2 x_{i+1,j}^2 - 2x_{i,j} x_{i+1,j} (i+\sigma_i,j) (g) \right\}$$

(3.39)

Geman et al omitted the terms $\sum_{i,j} \{ x_{i,j}^2 x_{i+1,j}^2 + x_{i,j}^2 x_{i+1,j}^2 \}$. This is wrong because these terms are not constant. For example, from one iteration to the next we may have different number of sites with label equal to 1.

The unbounded nonlinear transformation of equation 3.33 is not appropriate for this modified cost function. Instead, we suggest a bounded linear transformation of the form:

$$\phi(\omega) = \begin{cases} 
\left( \frac{\omega}{2(\omega_{\text{max}} - 1)} \right), & \text{for } 0 \leq \omega \leq 1 \\
\frac{1}{2} & \text{for } 1 < \omega \leq \omega_{\text{max}} 
\end{cases}$$

(3.40)

where now $\phi(\cdot)$ lies between 0 and 1 with minimum and maximum values as 0 and 1 respectively and $\omega_{\text{max}}$ is the maximum disparity value estimated from the data. So when $\omega = \omega_{\text{max}}$ then $\phi(\omega)$ is 1. The plot of this function is shown in figure 3.7.

3.4.1 Results and Discussion

I performed experiments with the new cost function. The rest of the algorithm was left as Geman et al defined it. A linear schedule for $\alpha$ with $\Delta \alpha = 0.02$ is used and $c_c$ is set to 0.01. I calculated the normalizing constants as per equation 3.32 but many boundaries were missing in the segmentated results I obtained and so I selected a lower set of thresholds such that at least 80% of all the horizontal and vertical disparity values for each disparity measure over a homogeneous test image are less
than the corresponding normalizing constant. The highest value of normalizing constant \( \lambda_i \) for a particular feature \( i \) among the textures present in the texture mosaic is taken as the final threshold value for that feature. Table 3.1 gives the values of normalization constants used for results of Fig. 3.8.

<table>
<thead>
<tr>
<th>Image in Fig. 3.8</th>
<th>( \lambda_1 )</th>
<th>( \lambda_2 )</th>
<th>( \lambda_3 )</th>
<th>( \lambda_4 )</th>
<th>( \lambda_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a),(d)</td>
<td>0.424</td>
<td>0.433</td>
<td>0.440</td>
<td>0.523</td>
<td>0.428</td>
</tr>
<tr>
<td>(b),(e)</td>
<td>0.541</td>
<td>0.471</td>
<td>0.453</td>
<td>0.451</td>
<td>0.489</td>
</tr>
<tr>
<td>(c),(f)</td>
<td>0.480</td>
<td>0.453</td>
<td>0.445</td>
<td>0.327</td>
<td>0.416</td>
</tr>
</tbody>
</table>

Table 3.1: The different parameter settings for results of figure 3.8.

The segmentation results are shown in figure 3.8. The results are only marginally better than the results I obtained with the unmodified cost function. The missing boundaries in the results are due to explicit declaration of penalty patterns in the algorithm. The quantitative analysis of the segmentated results is given in section 3.8.

Because the results of the two cost functions are so close, in all the remaining ex-
3.5 The Disparity Measures

Experiments I will present, both cost functions will be used. It is not easy to explain why the original cost function, although it is clearly wrong, gives similar results to the new function. The fact that the original cost function assumes that the total number of edge labels is constant, which is not true, means that from one iteration to the next we try to optimize a different cost function. On the other hand, simulated annealing is not really applied fully according to the rules that would make it too slow. It seems that these two inaccuracies of applying the theory compensate for each other and the net result is that the mistake in the cost function becomes almost irrelevant.

3.5 The Disparity Measures

As mentioned earlier, the Kolmogorov-Smirnov statistic alone is not appropriate in distinguishing different textures. Here, I introduce three extra disparity measures, in addition to the Kolmogorov-Smirnov one.
Chapter 3. Boundary Detection as a Global Optimization Problem

3.5.1 Linear Correlation Coefficient

The linear correlation coefficient $\rho$ is a measure of the strength of linear association between two variables.

We apply this measure for finding the strength of linear correlation between the two binned data sets derived from two data sets $x^{(1)}$ and $x^{(2)}$ as discussed earlier. It provides a test statistic for the hypothesis that the two data sets are samples from the same underlying probability distribution. The correlation coefficient in our case can be represented as

$$
\rho \equiv \frac{\sum_i (B_i - \bar{B})(b_i - \bar{b})}{\sqrt{\sum_j (B_j - \bar{B})^2 \sum_k (b_k - \bar{b})^2}}
$$

where, $\bar{B}$ is the mean of $B_i$'s and $\bar{b}$ is the mean of $b_i$'s and $i, j, k$ vary from 1 to $N_b$.

The value of $\rho$ lies between -1 and 1. When $\rho$ has value zero then the two sets of data are uncorrelated and when it is -1 they are anti correlated.

As we are interested in defining a measure of disparity which takes maximum value for maximum difference between the two blocks we compare, and minimum value when they are most similar, we define $\Delta_{s,t}(g)$ as:

$$
\Delta_{s,t}(g) = (1 - \rho)/2
$$

3.5.2 Chi-Square Test

The Chi-square statistic is defined by

$$
\chi^2 \equiv \sum_i \frac{(B_i - b_i)^2}{B_i + b_i},
$$

1

[3] 2

[20]
where, \( i \) varies from 1 to \( N_b \).

A large value of \( \chi^2 \) shows that the two data sets have different probability distributions. The Chi square statistic works best when all the \( B_i \) and \( b_i \) have value at least 5 and so if any of them are smaller than 5, we join neighbouring bins together to meet this condition.

### 3.5.3 Contraharmonic Filter

Instead of comparing blocks of pixel values on either side of a candidate edge element, we may use any other non-linear edge detection filter that has been proposed in the literature. One such filter is the Contraharmonic filter which may be used to assign a value that indicates how edge-like a site is. It is defined by [91]

\[
Y_{CH} = Y_{CHp}(x_1, x_2, ..., x_N) - Y_{CH-\rho}(x_1, x_2, ..., x_N) \tag{3.44}
\]

where

\[
Y_{CHp}(x_1, ..., x_N) = \frac{\sum_{i=1}^{N} x_i^{p+1}}{\sum_{i=1}^{N} x_i^p}. \tag{3.45}
\]

In this case \((x_1, x_2, ..., x_N)\) are the values of the pixels inside a square window centered at the label site to which \(Y_{CH}\) refers and \(p\) is an integer number.

The value of \(Y_{CH}\) is high when the window contains an edge and low otherwise.

Let us define that the values of \(Y_{CH}\) at any label position \((i, j)\) and its four nearest neighbours located at \((i-\sigma_l, j), (i+\sigma_l, j), (i, j-\sigma_l), (i, j+\sigma_l)\) are \(Y_{CHs}, Y_{CHsl}, Y_{CHsr}, Y_{CHsl}\) and \(Y_{CHsb}\) respectively. Then we define the disparity measures associated with the corresponding label segments as follows:
\[ \Delta_{(i,j),(i-\sigma_i,j)}(g) = \frac{Y_{CHS} + Y_{CHst}}{2} \]  
(3.46)

\[ \Delta_{(i,j),(i+\sigma_i,j)}(g) = \frac{Y_{CHS} + Y_{CHst}}{2} \]  
(3.47)

\[ \Delta_{(i,j),(i,j-\sigma_j)}(g) = \frac{Y_{CHS} + Y_{CHst}}{2} \]  
(3.48)

\[ \Delta_{(i,j),(i,j+\sigma_j)}(g) = \frac{Y_{CHS} + Y_{CHst}}{2} \]  
(3.49)

### 3.5.4 Combining the disparity measures

The idea is not only to use disparity measures computed as the same statistic from the various image transforms, but also to use different statistics. We were motivated in this by the observation that different statistics could discriminate between different textures. At the end, each candidate edge segment will be assigned only a single disparity value, the maximum over all disparities calculated from all the feature maps and with all statistics. For 5 features and 4 statistics, the maximum value of the disparity measure for any edge segment is selected from the 20 possible values.

### 3.5.5 Selection of normalization constants

After some experimentation we concluded that the best way to select the thresholds from the training was to require that at least 80% of the values of the disparity measures are below them when both blocks of pixels compared belong to the same texture. As we use five features for each image, we have five different disparity values computed for each segment. We choose the largest of the normalized disparities as the disparity of the segment. Once the disparity values are known, we look for the calculation of \( \phi(\omega) \).
3.5.6 Selection of $\phi(\omega)$

Earlier, we used two different expressions for $\phi(\omega)$: One given by Geman et al of equation 3.33 and the other one we suggested in the equation 3.40 for use with the modified cost function of equation 3.39.

Here, we also use a simple linear function for use with the original cost function:

$$\phi(\omega) \equiv \omega - 1$$  \hspace{1cm} (3.50)

From the experiments we found that $\phi(\omega)$ proposed by Geman et al and given by equation 3.33 works well with the Kolmogorov-Smirnov statistic but it does not give satisfactory results when used with the three other disparity measures. The linear function works well with all four disparity measures. We found virtually the same results with either function for the Kolmogorov-Smirnov statistic. Our aim is to develop a texture segmentation technique which relies on multiple statistics for segmentation and therefore we opted for the linear function which works well with all four statistics we use.

3.5.7 Results and Discussion

I used all four disparity measures namely, Kolmogorov-Smirnov statistic, linear correlation coefficient, Chi-square and Contraharmonic filters for the disparity calculations and then all four disparities were combined for assigning a maximum disparity value to any boundary segment. I present the segmentation results for the following:

- Individual disparity measure and the joint disparity measure when the original cost function of equation 3.9 is used (see figures 3.9 and 3.10).

- Individual disparity measure and the joint disparity measure when the modified cost function of equation 3.36 is used (see figures 3.11 and 3.12).
The features and block size for disparity estimation are taken as discussed earlier. A linear schedule with $\Delta x = 0.02$ is used for penalizing the illegal configurations and $c_c$ is set to 0.01. For the Contraharmonic filter, we have $N = 64$ (block of $8 \times 8$ pixels) and $p$ equal to 2 (see equation 3.45). For label resolution of 5, I scan the image and find values of $Y_{CH}$ for every fifth label location, considering the values of the pixels around it.

Table 3.2 gives details of different parameter settings used for the results of figures 3.9-3.12.

<table>
<thead>
<tr>
<th>Normalizing constants $\lambda_i$ for the test images</th>
<th>Image in Fig 3.9-3.12</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td></td>
<td>0.474</td>
<td>0.153</td>
<td>0.187</td>
<td>0.201</td>
<td>0.179</td>
</tr>
<tr>
<td>(b)</td>
<td></td>
<td>0.520</td>
<td>0.325</td>
<td>0.311</td>
<td>0.378</td>
<td>0.352</td>
</tr>
<tr>
<td>(c)</td>
<td></td>
<td>0.229</td>
<td>0.120</td>
<td>0.142</td>
<td>0.172</td>
<td>0.165</td>
</tr>
<tr>
<td>(d)</td>
<td></td>
<td>253.28</td>
<td>174.64</td>
<td>165.59</td>
<td>160.61</td>
<td>180.18</td>
</tr>
<tr>
<td>(e)</td>
<td></td>
<td>287.98</td>
<td>222.26</td>
<td>189.06</td>
<td>234.91</td>
<td>200.29</td>
</tr>
<tr>
<td>(f)</td>
<td></td>
<td>254.45</td>
<td>207.34</td>
<td>182.34</td>
<td>218.95</td>
<td>190.41</td>
</tr>
<tr>
<td>(g)</td>
<td></td>
<td>4.10</td>
<td>1.03</td>
<td>0.628</td>
<td>2.00</td>
<td>1.40</td>
</tr>
<tr>
<td>(h)</td>
<td></td>
<td>17.13</td>
<td>4.01</td>
<td>3.07</td>
<td>8.87</td>
<td>8.07</td>
</tr>
<tr>
<td>(i)</td>
<td></td>
<td>2.59</td>
<td>1.58</td>
<td>1.04</td>
<td>2.90</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Table 3.2: The different parameter settings for the results of figures 3.9-3.12.

The experimental results shown in figures 3.9 and 3.10 with the original cost function and in figures 3.11 and 3.12 with the modified cost function show that the algorithm when the multiple statistics are used performs better than when a single statistic is used. I want to emphasise here that my aim is not to compare various disparity measures but to improve the performance of the algorithm using the concept of multiple statistics. The results show that although a single statistic may give good results on some images, the results are consistently better with multiple statistics. The results with the modified cost function are marginally better than with the old cost function. I shall discuss their quantitative analysis in section 3.8.
Figure 3.9: Textural boundary estimation in some test images. Results with the original cost function and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure.
Figure 3.10: Textural boundary estimation in images. Results with the original cost function and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure.
3.5. The Disparity Measures

Figure 3.11: Textural boundary estimation in some test images. Results with the modified cost function and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure.
Figure 3.12: Textural boundary estimation in images. Results with the modified cost function and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the linear correlation coefficient measure, (d)(e)(f): Results with the Chi-square measure, (g)(h)(i): Results with the Contraharmonic filter, (j)(k)(l): Results with the joint disparity measure.
3.6. The Implicit Dictionary

The results with the images of figure 3.10 and 3.12 are rather poor. In particular, the results in figures 3.10(k) and 3.12(k) are poor even with multiple statistics which highlights that:

- The penalizable configurations of figure 3.3 are highly dependent on the configuration of the composite texture to be segmented.

3.6 The Implicit Dictionary

The basic shortcoming of the explicit dictionary of penalizable configurations is that the inclusion of a penalty term depends on the image on which boundary detection is to be performed. For example, Geman et al [39] included all penalty patterns as defined earlier in some images while for others they dropped the last penalty term. Thus, some prior knowledge of the image on which segmentation is to be performed is required for defining penalty patterns.

Instead of defining penalty patterns directly, we adopt a more general approach which does not require any prior information about the input image. We create a dictionary which contains those label configurations that promote the boundary formation. Such a dictionary will consists of the set of all $3 \times 3$ permissible label configurations and is a simplified version of the dictionary introduced by Hancock and Kittler [46]. Their dictionary is based on the assumption that edges are one pixel wide and that permissible edges propagate continuously in one direction or undergo orientation changes by no more than $\pi/2$. This dictionary reduces the total number of possible combinations of labels in a $3 \times 3$ neighbourhood from $2^9$ to 45 [46]. We have modified this dictionary to suit our cost function which detects edges with highest sensitivity in horizontal or vertical directions. In our case therefore, edges can only be horizontal or vertical. This reduces the possible combinations of labels from 45 to 31. Examples of entries which are included in the dictionary are shown in figure 3.13.
Chapter 3. Boundary Detection as a Global Optimization Problem

Our cost function now will include many more penalty terms than before, all those combinations of labels which are *not* part of the dictionary. Thus our dictionary of penalized configurations is only implicitly defined. The penalty function is incremented by 1 each time the relevant pattern is not part of the dictionary of allowable configurations.

To deal with the fact that the number of penalty terms is much larger now than it was before, the value of parameter $\alpha$ in equation 3.9 has to be appropriately modified.

Figure 3.13: Example entries of the dictionary. "1" indicates the presence of an edge while blank indicates no edge present.

### 3.6.1 Results and Discussion

I present segmentation results for the following:

- Individual disparity measure and the joint disparity measure with the implicit dictionary used for penalization when the original cost function is used (see figures 3.14, 3.15, 3.16, and 3.20).

- Individual disparity measure and the joint disparity measure with the implicit dictionary used for penalization when the modified cost function is used (see figures 3.21 and 3.22).
3.6. The Implicit Dictionary

First, I describe the segmentation results I achieved when the original cost function of equation 3.9 is used. The same set of normalization constants for all the disparity measures as given in tables 3.1 and 3.2 are used. Two "cooling" schedules have been experimented with: linear and parabolic.

In the linear schedule $\alpha$ is zero during the first iteration of annealing and then it is increased in steps of $\Delta \alpha$ at every iteration.

In the parabolic schedule $\alpha_k = k^{0.5}/10$, where $k$ is the iteration number.

In experiments with the linear schedule, I found that $\alpha$ can be kept constant (without affecting the final output) after initially increasing it for a few iterations until the total value of the penalty terms in the cost function reduces to a constant value.

Figures 3.14 and 3.15 show the segmentation results when the original cost function is used with the implicit dictionary and a linear cooling for $\alpha$, while figure 3.16 shows the results when the linear schedule for $\alpha$ is replaced by the parabolic cooling schedule.

To get a feeling about the influence of the cooling schedule of $\alpha$, figures 3.17, 3.18 and 3.19 show the plots of individual terms of the energy function of equation 3.9 for a linear cooling schedule with $\Delta \alpha = 0.02$, $\Delta \alpha = 0.01$ and for a parabolic schedule for images 3.14(n), 3.15(n) and 3.16(n) respectively.

From the experimental results, I observed that the lower value of linear schedule ($\Delta \alpha = 0.01$) gives better segmentation results than the higher value of $\Delta \alpha$ or the parabolic schedule and therefore I use linear schedule with $\Delta \alpha = 0.01$ for the remaining set of experiments.

The results of experiments run with the same set of normalization constants of tables 3.1 and 3.2 for images consisting of the same homogeneous textures but arranged to form different composite textures, are shown in the figure 3.20.

The experimental results with the modified cost function for different disparity measures and the joint disparity measure and with implicit dictionary for penal-
Figure 3.14: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
3.6. The Implicit Dictionary

Figure 3.15: Textural boundary estimation in images. Results with the original cost function, implicit dictionary, and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.16: Textural boundary estimation in images. Results with the original cost function, implicit dictionary, and a parabolic cooling schedule for $\alpha$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
3.6. The Implicit Dictionary

Figure 3.17: The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.14(n). The implicit dictionary is used and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

Figure 3.18: The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.15(n). The implicit dictionary is used and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. 
Figure 3.19: The original cost function and its components as a function of the number of iterations of simulated annealing for results of figure 3.16(n). The implicit dictionary is used with a parabolic cooling schedule for $\alpha$.

These experiments again confirm that multiple statistics with the implicit dictionary give better segmentation than any single statistic with either the explicit or the implicit dictionary. However, the algorithm is still highly dependent on the normalization constant selection procedure which requires prior knowledge of the homogeneous textures present in the composite texture to be segmented.

### 3.7 The Common Set of Thresholds

At this stage we have not abandoned the training stage of the algorithm. During this stage the appropriate thresholds are determined, with the help of uniform texture images that represent the textures that are expected to be found in the patchwork textured images we have to segment. This set of experiments are aimed at eliminating the training phase (normalization constant selection procedure) of this algorithm.
Figure 3.20: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.21: Textural boundary estimation in images. Results with the modified cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
3.7. The Common Set of Thresholds

Figure 3.22: Textural boundary estimation in images. Results with the modified cost function, implicit dictionary and a linear schedule for \( \alpha \) with \( \Delta \alpha = 0.01 \).

- (a)(b)(c): Results with the Kolmogorov-Smirnov statistic,
- (d)(e)(f): Results with the linear correlation coefficient measure,
- (g)(h)(i): Results with the Chi-square measure,
- (j)(k)(l): Results with the Contraharmonic filter,
- (m)(n)(o): Results with the joint disparity measure.
For this purpose I incorporate the idea of a database of homogeneous textures for the calculation of the normalization constants. I performed various experiments by randomly generating composite textures from the homogeneous textures of Fig 3.23 and decided to select a common set of normalization constants for individual features, for each statistics I use, over this training database and the same set of constants to be used for all the test images. I found, however, that the procedure suggested by Geman et al [39] for normalization constants selection cannot incorporate the concept of selection of a common set of normalization constants. My experiments in this direction showed that many boundary labels are lost in different test images if a common set of thresholds is computed according to equation 3.32. I adopted a different approach to solve this problem.

Robust Normalizing Constants

I selected $\lambda_{d,c}$ (see equation 3.32) for an individual texture of the database as the 90 percentile of the total horizontal and vertical disparity values derived for any of the disparity measures. Then instead of adopting the maximum value to $\lambda_{d}$, I used the average value of $\lambda_{i}$ over all textures of the database.

$$\lambda_{d} = \frac{1}{C_{d}} \sum_{j=1}^{C_{d}} \lambda_{i,j} \quad \forall i \quad (3.51)$$

where $C_{d}$ is the total number of homogeneous textures in the image database of figure 3.23 and $1 \leq i \leq 5$.

The advantage of using the above approach is:

- The selection of normalization constant is made independent of the textures present in the texture mosaic to be segmentated. The training phase I use here is for a whole class of mosaic images and over all textures that may or may not be present in a single composite image.
3.7. The Common Set of Thresholds

Figure 3.23: Ten homogeneous textures used for the calculation of the common set of normalization constants for each disparity measure. All images are $128 \times 128$ in size.
• The normalizing constants selected by my approach will always be having lower value than the normalizing constants selected by the approach proposed by [39]. This means that in my case more boundary segments will have disparity values above the threshold. However, it is expected that the virtual dictionary which incorporates lots of penalty terms will cope with any unwanted boundary labels.

For each disparity measure, the normalization constants \( \lambda_i \), where \( 1 \leq i \leq 5 \) were derived separately for each homogeneous texture of figure 3.23. The average value (see equation 3.51) of \( \lambda_i \) was then selected as the common normalization constants for that disparity measure. These common normalization constants for each disparity measure are independent from the composite textures on which the segmentation experiment is to be performed. Tables 3.3, 3.4, 3.5, and 3.6 give the individual values of \( \lambda_i \) for the homogeneous textures of the database shown in figure 3.23 for the Kolmogorov-Smirnov statistic, the linear correlation coefficient measure, the Chi-square values and the Contraharmonic filter respectively. The set of average values is also listed at the end of each table.

<table>
<thead>
<tr>
<th>Normalizing constants ( \lambda_i ) based on the Kolmogorov-Smirnov statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homogeneous Textures of Fig 3.23</td>
</tr>
<tr>
<td>(a)</td>
</tr>
<tr>
<td>(b)</td>
</tr>
<tr>
<td>(c)</td>
</tr>
<tr>
<td>(d)</td>
</tr>
<tr>
<td>(e)</td>
</tr>
<tr>
<td>(f)</td>
</tr>
<tr>
<td>(g)</td>
</tr>
<tr>
<td>(h)</td>
</tr>
<tr>
<td>(i)</td>
</tr>
<tr>
<td>(j)</td>
</tr>
<tr>
<td>Average</td>
</tr>
</tbody>
</table>

Table 3.3: The normalizing constants for 5 features (Range and 4 Directional residuals).
3.7. The Common Set of Thresholds

### Normalizing constants $\lambda_i$ based on the Linear correlation coefficient

<table>
<thead>
<tr>
<th>Homogeneous Textures of Fig 3.23</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.538</td>
<td>0.050</td>
<td>0.097</td>
<td>0.124</td>
<td>0.154</td>
</tr>
<tr>
<td>(b)</td>
<td>0.285</td>
<td>0.168</td>
<td>0.119</td>
<td>0.163</td>
<td>0.084</td>
</tr>
<tr>
<td>(c)</td>
<td>0.446</td>
<td>0.117</td>
<td>0.105</td>
<td>0.180</td>
<td>0.161</td>
</tr>
<tr>
<td>(d)</td>
<td>0.360</td>
<td>0.185</td>
<td>0.239</td>
<td>0.257</td>
<td>0.256</td>
</tr>
<tr>
<td>(e)</td>
<td>0.371</td>
<td>0.129</td>
<td>0.187</td>
<td>0.188</td>
<td>0.117</td>
</tr>
<tr>
<td>(f)</td>
<td>0.462</td>
<td>0.358</td>
<td>0.344</td>
<td>0.428</td>
<td>0.392</td>
</tr>
<tr>
<td>(g)</td>
<td>0.342</td>
<td>0.206</td>
<td>0.187</td>
<td>0.222</td>
<td>0.222</td>
</tr>
<tr>
<td>(h)</td>
<td>0.575</td>
<td>0.246</td>
<td>0.298</td>
<td>0.382</td>
<td>0.294</td>
</tr>
<tr>
<td>(i)</td>
<td>0.501</td>
<td>0.020</td>
<td>0.096</td>
<td>0.137</td>
<td>0.175</td>
</tr>
<tr>
<td>(j)</td>
<td>0.553</td>
<td>0.028</td>
<td>0.113</td>
<td>0.235</td>
<td>0.173</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>0.443</td>
<td>0.151</td>
<td>0.178</td>
<td>0.232</td>
<td>0.203</td>
</tr>
</tbody>
</table>

Table 3.4: The normalizing constants for 5 features (Range and 4 Directional residuals).

3.7.1 Results and Discussion

I present segmentation results for the following:

- Individual disparity measure and the joint disparity measure with the implicit dictionary and the common set of thresholds for each of the disparity measures and the joint disparity measure when the original cost function is used (see figures 3.24-3.27).

- Individual disparity measure and the joint disparity measure with the implicit dictionary and the common set of thresholds for each of the disparity measures and the joint disparity measure when the modified cost function is used (see figures 3.28-3.31.)

The experiments were performed in order to clarify

- How the texture segmentation algorithm works when some of the textures of the training database may or may not be present in the composite image.
### Normalizing constants $\lambda_i$ based on Chi-square distribution

<table>
<thead>
<tr>
<th>Homogeneous Textures of Fig 3.23</th>
<th>$\lambda_1$</th>
<th>$\lambda_2$</th>
<th>$\lambda_3$</th>
<th>$\lambda_4$</th>
<th>$\lambda_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>294.12</td>
<td>93.30</td>
<td>86.36</td>
<td>128.59</td>
<td>163.75</td>
</tr>
<tr>
<td>(b)</td>
<td>216.44</td>
<td>117.49</td>
<td>96.62</td>
<td>126.89</td>
<td>75.73</td>
</tr>
<tr>
<td>(c)</td>
<td>273.44</td>
<td>116.84</td>
<td>110.95</td>
<td>190.42</td>
<td>167.38</td>
</tr>
<tr>
<td>(d)</td>
<td>211.61</td>
<td>161.92</td>
<td>174.88</td>
<td>172.08</td>
<td>171.44</td>
</tr>
<tr>
<td>(e)</td>
<td>223.24</td>
<td>113.10</td>
<td>148.58</td>
<td>127.41</td>
<td>107.38</td>
</tr>
<tr>
<td>(f)</td>
<td>282.90</td>
<td>263.10</td>
<td>186.57</td>
<td>179.85</td>
<td>229.94</td>
</tr>
<tr>
<td>(g)</td>
<td>205.85</td>
<td>135.29</td>
<td>137.98</td>
<td>124.85</td>
<td>147.55</td>
</tr>
<tr>
<td>(h)</td>
<td>328.73</td>
<td>212.93</td>
<td>223.36</td>
<td>256.09</td>
<td>211.75</td>
</tr>
<tr>
<td>(i)</td>
<td>287.86</td>
<td>67.47</td>
<td>85.68</td>
<td>145.61</td>
<td>154.45</td>
</tr>
<tr>
<td>(j)</td>
<td>355.51</td>
<td>87.98</td>
<td>111.58</td>
<td>237.95</td>
<td>196.04</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td>267.97</td>
<td>136.94</td>
<td>136.26</td>
<td>168.97</td>
<td>162.54</td>
</tr>
</tbody>
</table>

Table 3.5: The normalizing constants for 5 features (Range and 4 Directional residuals).

- How the texture segmentation algorithm works when textures that were not included in the training database are present in an image.

Experimental results concerning the first point are shown in Fig 3.24 and 3.25. Experimental results concerning the second point are shown in Fig 3.26 and 3.27.

Note that in image of Fig 3.26(a), only one homogeneous texture is from the database of Fig 3.23. In image of Fig 3.26(b), all homogeneous textures are from the database. Image of Fig 3.26(c) consists of four Gaussian Markov random field (GMRF) [58] textures and none of the individual textures is part of the database.

Similarly for image of figure 3.27(a), both the homogeneous textures are from the database while for image of figure 3.27(b), only one homogeneous texture is from the database. Note that these two images have only diagonal and anti-diagonal textured edges while our both cost functions incorporate only horizontal and vertical boundary segments.

The experimental results when the modified cost function is used are shown in figures 3.28–3.31.
3.7. The Common Set of Thresholds

Figure 3.24: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contra-harmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.25: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
3.7. The Common Set of Thresholds

Figure 3.26: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$.(a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.27: Textural boundary estimation in images. Results with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b): Results with the Kolmogorov-Smirnov statistic, (c)(d): Results with the linear correlation coefficient measure, (e)(f): Results with the Chi-square measure, (g)(h): Results with the Contraharmonic filter, (i)(j): Results with the joint disparity measure.
3.7. *The Common Set of Thresholds*

Figure 3.28: Textural boundary estimation in images. Results with the modified cost function and the implicit dictionary. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.29: Textural boundary estimation in some test images. Results with the modified cost function and the implicit dictionary. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.30: Textural boundary estimation in some test images. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
Figure 3.31: Textural boundary estimation in some test images. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used. (a)(b)(c): Results with the Kolmogorov-Smirnov statistic, (d)(e)(f): Results with the linear correlation coefficient measure, (g)(h)(i): Results with the Chi-square measure, (j)(k)(l): Results with the Contraharmonic filter, (m)(n)(o): Results with the joint disparity measure.
3.7. The Common Set of Thresholds

| Normalizing constants $\lambda_i$ based on the Contraharmonic filter |
| Homogeneous Textures of Fig 3.23 | $\lambda_1$ | $\lambda_2$ | $\lambda_3$ | $\lambda_4$ | $\lambda_5$ |
| (a) | 3.09 | 0.171 | 0.303 | 0.406 | 0.234 |
| (b) | 0.981 | 0.856 | 0.809 | 1.58 | 1.24 |
| (c) | 8.42 | 0.35 | 0.31 | 0.61 | 0.63 |
| (d) | 7.24 | 1.73 | 1.02 | 3.03 | 2.16 |
| (e) | 1.38 | 0.20 | 0.26 | 0.55 | 0.28 |
| (f) | 7.08 | 0.42 | 0.32 | 0.84 | 0.64 |
| (g) | 7.14 | 3.15 | 1.98 | 5.30 | 3.20 |
| (h) | 6.89 | 1.71 | 2.59 | 4.50 | 2.97 |
| (i) | 5.28 | 2.30 | 1.41 | 4.33 | 2.04 |
| (j) | 20.60 | 7.48 | 5.48 | 11.06 | 9.55 |
| **Average** | 6.81 | 1.83 | 1.44 | 3.22 | 2.29 |

Table 3.6: The normalizing constants for 5 features (Range and 4 Directional residuals).

The results with the common set of thresholds are worse than those presented in the previous sections because the segmentation algorithm has not been tuned to each particular image separately. The results with the multiple statistics for both the cost functions are certainly better than the results with any single statistic for most of the images.

Although the results of the Contraharmonic filter in figures 3.26 and 3.30 are poor, they are marginally better with the modified cost function. This also indicate that the Contraharmonic filter works well only when prior information about texture categories are known.

The results of figures 3.27 and 3.31 look very similar. I shall present the quantitative analysis of these results in the next section.
3.8 Overview of the Results

I calculate under and over segmentation errors for each of the segmented results for quantifying the quality of the results I obtained with different approaches. The errors are calculated as follows:

Let \( N_g \) be the total number of boundary labels (labels having value "1") in the ground truth of the test image.

Scan the whole ground truth image in raster scan and for each boundary label \( x_{ij}^g \) look for a corresponding boundary label in a window of 3 x 3 centred at \((i,j)\) in the output image. Increment a counter by 1 if there is a boundary label in this window. Let \( N_o \) be the final value of this counter.

Then

\[
Under\ detection\ error = \frac{N_g - N_o}{N_g} \quad (3.52)
\]

To calculate the over segmentation error, find the total number \( n_o \) of boundary labels \( x_{ij}^o \) in the output image. Scan the output image in raster scan fashion and look for a corresponding boundary label in a window of 3 x 3 centred at \((i,j)\) in the ground truth image. Increment a counter by 1 if there is a boundary pixel inside this window. Let \( n_g \) be the final value of this counter.

Then

\[
Over\ detection\ error = \frac{n_o - n_g}{n_o} \quad (3.53)
\]

The under and over detection errors for each of the segmentation results shown earlier are given in tables 3.7–3.26
### 3.8. Overview of the Results

<table>
<thead>
<tr>
<th>Image of Fig. 3.5</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>75.27%</td>
<td>76.19%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>77.27%</td>
<td>77.51%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>83.01%</td>
<td>80.37%</td>
</tr>
<tr>
<td>(d)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>64.95%</td>
<td>76.19%</td>
</tr>
<tr>
<td>(e)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>79.73%</td>
<td>70.04%</td>
</tr>
<tr>
<td>(f)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>50.66%</td>
<td>48.40%</td>
</tr>
</tbody>
</table>

Table 3.7: Errors for Geman et al’s approach. The Kolmogorov-Smirnov statistic is used as disparity measure with a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

<table>
<thead>
<tr>
<th>Image of Fig. 3.8</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>73.63%</td>
<td>75.11%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>76.18%</td>
<td>76.38%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>77.09%</td>
<td>75.86%</td>
</tr>
<tr>
<td>(d)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>49.33%</td>
<td>40.58%</td>
</tr>
<tr>
<td>(e)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>82.78%</td>
<td>80.19%</td>
</tr>
<tr>
<td>(f)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>57.11%</td>
<td>55.70%</td>
</tr>
</tbody>
</table>

Table 3.8: Errors with the modified cost function. The Kolmogorov-Smirnov statistic is used as a disparity measure with a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

<table>
<thead>
<tr>
<th>Image of Fig. 3.9</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Linear Correlation Coefficient</td>
<td>56.18%</td>
<td>59.94%</td>
</tr>
<tr>
<td>(d)</td>
<td>Chi Square</td>
<td>74.54%</td>
<td>75.03%</td>
</tr>
<tr>
<td>(g)</td>
<td>Contra-Harmonic Filter</td>
<td>72.90%</td>
<td>73.01%</td>
</tr>
<tr>
<td>(j)</td>
<td>Joint Disparity</td>
<td>44.36%</td>
<td>45.89%</td>
</tr>
<tr>
<td>(b)</td>
<td>Linear Correlation Coefficient</td>
<td>53.45%</td>
<td>54.62%</td>
</tr>
<tr>
<td>(e)</td>
<td>Chi Square</td>
<td>83.09%</td>
<td>81.99%</td>
</tr>
<tr>
<td>(h)</td>
<td>Contra-Harmonic Filter</td>
<td>76.28%</td>
<td>87.92%</td>
</tr>
<tr>
<td>(k)</td>
<td>Joint Disparity</td>
<td>46.54%</td>
<td>47.83%</td>
</tr>
<tr>
<td>(c)</td>
<td>Linear Correlation Coefficient</td>
<td>47.81%</td>
<td>46.72%</td>
</tr>
<tr>
<td>(f)</td>
<td>Chi Square</td>
<td>69.45%</td>
<td>71.55%</td>
</tr>
<tr>
<td>(i)</td>
<td>Contra-Harmonic Filter</td>
<td>81.45%</td>
<td>84.59%</td>
</tr>
<tr>
<td>(l)</td>
<td>Joint Disparity</td>
<td>46.90%</td>
<td>46.04%</td>
</tr>
</tbody>
</table>

Table 3.9: Errors for various disparity measures. The original cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$. 
### Table 3.10: Errors for various disparity measures. The original cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

<table>
<thead>
<tr>
<th>Image of Fig. 3.10</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Linear Correlation Coefficient</td>
<td>27.57%</td>
<td>33.48%</td>
</tr>
<tr>
<td>(d)</td>
<td>Chi Square</td>
<td>50.83%</td>
<td>59.19%</td>
</tr>
<tr>
<td>(g)</td>
<td>Contra-Harmonic Filter</td>
<td>68.43%</td>
<td>66.12%</td>
</tr>
<tr>
<td>(j)</td>
<td>Joint disparity</td>
<td>23.92%</td>
<td>31.85%</td>
</tr>
<tr>
<td>(b)</td>
<td>Linear Correlation Coefficient</td>
<td>58.63%</td>
<td>42.19%</td>
</tr>
<tr>
<td>(e)</td>
<td>Chi Square</td>
<td>68.77%</td>
<td>70.56%</td>
</tr>
<tr>
<td>(h)</td>
<td>Contra-Harmonic Filter</td>
<td>47.00%</td>
<td>76.18%</td>
</tr>
<tr>
<td>(k)</td>
<td>Joint disparity</td>
<td>59.30%</td>
<td>66.89%</td>
</tr>
<tr>
<td>(c)</td>
<td>Linear Correlation Coefficient</td>
<td>44.01%</td>
<td>50.14%</td>
</tr>
<tr>
<td>(f)</td>
<td>Chi Square</td>
<td>63.45%</td>
<td>60.43%</td>
</tr>
<tr>
<td>(i)</td>
<td>Contra-Harmonic Filter</td>
<td>64.79%</td>
<td>66.02%</td>
</tr>
<tr>
<td>(l)</td>
<td>Joint disparity</td>
<td>41.09%</td>
<td>48.18%</td>
</tr>
</tbody>
</table>

### Table 3.11: Errors for various disparity measures. The modified cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

<table>
<thead>
<tr>
<th>Image of Fig. 3.11</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Linear Correlation Coefficient</td>
<td>60.36%</td>
<td>61.16%</td>
</tr>
<tr>
<td>(d)</td>
<td>Chi Square</td>
<td>78.54%</td>
<td>80.00%</td>
</tr>
<tr>
<td>(g)</td>
<td>Contra-Harmonic Filter</td>
<td>72.90%</td>
<td>74.54%</td>
</tr>
<tr>
<td>(j)</td>
<td>Joint Disparity</td>
<td>41.09%</td>
<td>42.36%</td>
</tr>
<tr>
<td>(b)</td>
<td>Linear Correlation Coefficient</td>
<td>45.27%</td>
<td>49.00%</td>
</tr>
<tr>
<td>(e)</td>
<td>Chi Square</td>
<td>62.90%</td>
<td>60.24%</td>
</tr>
<tr>
<td>(h)</td>
<td>Contra-Harmonic Filter</td>
<td>67.45%</td>
<td>49.64%</td>
</tr>
<tr>
<td>(k)</td>
<td>Joint Disparity</td>
<td>44.54%</td>
<td>45.90%</td>
</tr>
<tr>
<td>(c)</td>
<td>Linear Correlation Coefficient</td>
<td>51.81%</td>
<td>50.65%</td>
</tr>
<tr>
<td>(f)</td>
<td>Chi Square</td>
<td>81.27%</td>
<td>82.14%</td>
</tr>
<tr>
<td>(i)</td>
<td>Contra-Harmonic Filter</td>
<td>60.00%</td>
<td>60.69%</td>
</tr>
<tr>
<td>(l)</td>
<td>Joint Disparity</td>
<td>44.54%</td>
<td>46.44%</td>
</tr>
</tbody>
</table>
### 3.8. Overview of the Results

<table>
<thead>
<tr>
<th>Image of Fig. 3.12</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Linear Correlation Coefficient</td>
<td>31.11%</td>
<td>40.65%</td>
</tr>
<tr>
<td>(d)</td>
<td>Chi Square</td>
<td>80.46%</td>
<td>81.68%</td>
</tr>
<tr>
<td>(g)</td>
<td>Contra-Harmonic Filter</td>
<td>62.91%</td>
<td>73.02%</td>
</tr>
<tr>
<td>(j)</td>
<td>Joint Disparity</td>
<td>28.14%</td>
<td>34.14%</td>
</tr>
<tr>
<td>(b)</td>
<td>Linear Correlation Coefficient</td>
<td>40.06%</td>
<td>43.94%</td>
</tr>
<tr>
<td>(e)</td>
<td>Chi Square</td>
<td>50.33%</td>
<td>54.28%</td>
</tr>
<tr>
<td>(h)</td>
<td>Contra-Harmonic Filter</td>
<td>59.10%</td>
<td>70.81%</td>
</tr>
<tr>
<td>(k)</td>
<td>Joint Disparity</td>
<td>57.78%</td>
<td>45.11%</td>
</tr>
<tr>
<td>(c)</td>
<td>Linear Correlation Coefficient</td>
<td>36.58%</td>
<td>40.98%</td>
</tr>
<tr>
<td>(f)</td>
<td>Chi Square</td>
<td>60.26%</td>
<td>56.85%</td>
</tr>
<tr>
<td>(i)</td>
<td>Contra-Harmonic Filter</td>
<td>42.54%</td>
<td>47.19%</td>
</tr>
<tr>
<td>(l)</td>
<td>Joint Disparity</td>
<td>30.90%</td>
<td>30.90%</td>
</tr>
</tbody>
</table>

Table 3.12: Errors for various disparity measures. The modified cost function is used with the explicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.02$.

<table>
<thead>
<tr>
<th>Image of Fig 3.14</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>62.90%</td>
<td>60.12%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>58.18%</td>
<td>51.75%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>74.18%</td>
<td>70.24%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>74.90%</td>
<td>78.83%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>55.27%</td>
<td>53.94%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>51.45%</td>
<td>54.30%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>41.74%</td>
<td>41.63%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>68.00%</td>
<td>67.30%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>87.45. %</td>
<td>83.09%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>40.72%</td>
<td>41.12%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>70.90%</td>
<td>65.00%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>66.72%</td>
<td>64.79%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>68.00%</td>
<td>67.30%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>69.81%</td>
<td>69.31%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>58.00%</td>
<td>57.18%</td>
</tr>
</tbody>
</table>

Table 3.13: Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$. 
### Table 3.14: Errors for various disparity measures. The original cost function is used with the implicit dictionary with a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$.

<table>
<thead>
<tr>
<th>Image of Fig 3.15</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>61.27%</td>
<td>61.80%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>56.00%</td>
<td>57.07%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>68.72%</td>
<td>70.70%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>60.00%</td>
<td>61.18%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>41.63%</td>
<td>41.88%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>46.72%</td>
<td>49.63%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>48.18%</td>
<td>47.81%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>58.36%</td>
<td>56.54%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>70.72%</td>
<td>72.64%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>38.18%</td>
<td>38.00%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>72.18%</td>
<td>70.09%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>72.18%</td>
<td>70.09%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>72.18%</td>
<td>70.09%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>72.18%</td>
<td>70.09%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>72.18%</td>
<td>70.09%</td>
</tr>
</tbody>
</table>

### Table 3.15: Errors for various disparity measures. The original cost function is used with the implicit dictionary and the parabolic cooling schedule for $\alpha$.

<table>
<thead>
<tr>
<th>Image of Fig 3.16</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>76.36%</td>
<td>68.93%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>51.81%</td>
<td>54.25%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>64.36%</td>
<td>65.44%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>72.72%</td>
<td>72.05%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>51.09%</td>
<td>51.55%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>70.90%</td>
<td>68.55%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>46.36%</td>
<td>48.13%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>84.72%</td>
<td>78.65%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>78.54%</td>
<td>76.89%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>43.27%</td>
<td>44.23%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>81.45%</td>
<td>73.12%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>75.81%</td>
<td>69.21%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>68.72%</td>
<td>64.88%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>76.54%</td>
<td>73.71%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>58.90%</td>
<td>57.29%</td>
</tr>
</tbody>
</table>
### 3.8. Overview of the Results

**Table 3.16:** Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$.

<table>
<thead>
<tr>
<th>Image of Fig 3.20</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>58.97%</td>
<td>60.28%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>39.53%</td>
<td>38.85%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>53.32%</td>
<td>54.45%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>51.16%</td>
<td>52.18%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>22.09%</td>
<td>21.60%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>25.91%</td>
<td>31.59%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>47.34%</td>
<td>47.94%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>49.33%</td>
<td>49.50%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>46.84%</td>
<td>54.80%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>14.11%</td>
<td>14.83%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>76.74%</td>
<td>76.17%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>59.83%</td>
<td>57.86%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>59.30%</td>
<td>58.26%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>63.28%</td>
<td>64.30%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>33.88%</td>
<td>33.88%</td>
</tr>
</tbody>
</table>

**Table 3.17:** Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$.

<table>
<thead>
<tr>
<th>Image of Fig 3.21</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>75.45%</td>
<td>73.23%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>50.00%</td>
<td>52.23%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>47.27%</td>
<td>46.74%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>66.90%</td>
<td>65.63%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>42.72%</td>
<td>43.45%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>47.63%</td>
<td>50.29%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>49.09%</td>
<td>49.22%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>73.63%</td>
<td>74.81%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>69.45%</td>
<td>52.11%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>44.00%</td>
<td>42.54%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>80.72%</td>
<td>76.92%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>63.45%</td>
<td>59.41%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>70.90%</td>
<td>72.58%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>47.27%</td>
<td>46.07%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>52.36%</td>
<td>52.11%</td>
</tr>
</tbody>
</table>
### Chapter 3. Boundary Detection as a Global Optimization Problem

#### Image of Fig 3.22

<table>
<thead>
<tr>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Kolmogorov-Smirnov Statistic</td>
<td>42.71%</td>
<td>37.62%</td>
</tr>
<tr>
<td>(d) Linear Correlation Coefficient</td>
<td>59.27%</td>
<td>54.96%</td>
</tr>
<tr>
<td>(g) Chi Square</td>
<td>55.62%</td>
<td>59.75%</td>
</tr>
<tr>
<td>(j) Contra-Harmonic Filter</td>
<td>48.67%</td>
<td>48.24%</td>
</tr>
<tr>
<td>(m) Joint disparity</td>
<td>39.07%</td>
<td>40.22%</td>
</tr>
<tr>
<td>(b) Kolmogorov-Smirnov Statistic</td>
<td>67.05%</td>
<td>67.85%</td>
</tr>
<tr>
<td>(e) Linear Correlation Coefficient</td>
<td>40.19%</td>
<td>47.91%</td>
</tr>
<tr>
<td>(h) Chi Square</td>
<td>55.46%</td>
<td>58.67%</td>
</tr>
<tr>
<td>(k) Contra-Harmonic Filter</td>
<td>44.37%</td>
<td>39.16%</td>
</tr>
<tr>
<td>(n) Joint disparity</td>
<td>33.94%</td>
<td>34.48%</td>
</tr>
<tr>
<td>(c) Kolmogorov-Smirnov Statistic</td>
<td>59.43%</td>
<td>59.09%</td>
</tr>
<tr>
<td>(f) Linear Correlation Coefficient</td>
<td>43.27%</td>
<td>47.81%</td>
</tr>
<tr>
<td>(i) Chi Square</td>
<td>68.87%</td>
<td>66.72%</td>
</tr>
<tr>
<td>(l) Contra-Harmonic Filter</td>
<td>51.82%</td>
<td>52.15%</td>
</tr>
<tr>
<td>(o) Joint disparity</td>
<td>41.22%</td>
<td>40.90%</td>
</tr>
</tbody>
</table>

Table 3.18: Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta\alpha = 0.01$.

#### Image of Fig 3.24

<table>
<thead>
<tr>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Kolmogorov-Smirnov Statistic</td>
<td>72.72%</td>
<td>71.77%</td>
</tr>
<tr>
<td>(d) Linear Correlation Coefficient</td>
<td>45.81%</td>
<td>47.14%</td>
</tr>
<tr>
<td>(g) Chi Square</td>
<td>80.72%</td>
<td>76.36%</td>
</tr>
<tr>
<td>(j) Contra-Harmonic Filter</td>
<td>66.72%</td>
<td>67.10%</td>
</tr>
<tr>
<td>(m) Joint disparity</td>
<td>60.90%</td>
<td>60.12%</td>
</tr>
<tr>
<td>(b) Kolmogorov-Smirnov Statistic</td>
<td>65.27%</td>
<td>64.99%</td>
</tr>
<tr>
<td>(e) Linear Correlation Coefficient</td>
<td>58.90%</td>
<td>57.78%</td>
</tr>
<tr>
<td>(h) Chi Square</td>
<td>88.18%</td>
<td>86.76%</td>
</tr>
<tr>
<td>(k) Contra-Harmonic Filter</td>
<td>80.18%</td>
<td>78.98%</td>
</tr>
<tr>
<td>(n) Joint disparity</td>
<td>73.27%</td>
<td>72.52%</td>
</tr>
<tr>
<td>(c) Kolmogorov-Smirnov Statistic</td>
<td>78.00%</td>
<td>77.57%</td>
</tr>
<tr>
<td>(f) Linear Correlation Coefficient</td>
<td>78.54%</td>
<td>71.99%</td>
</tr>
<tr>
<td>(i) Chi Square</td>
<td>75.81%</td>
<td>71.82%</td>
</tr>
<tr>
<td>(l) Contra-Harmonic Filter</td>
<td>62.00%</td>
<td>60.51%</td>
</tr>
<tr>
<td>(o) Joint disparity</td>
<td>68.54%</td>
<td>67.36%</td>
</tr>
</tbody>
</table>

Table 3.19: Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling for $\alpha$ with $\Delta\alpha = 0.01$. The common set of thresholds are used.
### 3.8. Overview of the Results

<table>
<thead>
<tr>
<th>Image of Fig 3.25</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>54.98%</td>
<td>54.81%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>51.66%</td>
<td>51.82%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>50.33%</td>
<td>49.58%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>62.45%</td>
<td>55.94%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>23.75%</td>
<td>23.28%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>44.01%</td>
<td>43.71%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>41.02%</td>
<td>41.36%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>59.80%</td>
<td>58.97%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>60.79%</td>
<td>65.35%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>39.36%</td>
<td>39.02%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>69.60%</td>
<td>68.84%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>67.27%</td>
<td>65.73%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>74.75%</td>
<td>73.88%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>64.79%</td>
<td>66.02%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>59.63%</td>
<td>57.49%</td>
</tr>
</tbody>
</table>

Table 3.20: Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.

<table>
<thead>
<tr>
<th>Image of Fig 3.26</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>0.0%</td>
<td>29.41%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>52.34%</td>
<td>53.49%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>72.37%</td>
<td>73.41%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>98.51%</td>
<td>98.51%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>49.87%</td>
<td>79.20%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>25.18%</td>
<td>25.18%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>90.27%</td>
<td>90.60%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>29.62%</td>
<td>32.14%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>60.98%</td>
<td>62.38%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>99.25%</td>
<td>98.79%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>60.70%</td>
<td>61.21%</td>
</tr>
</tbody>
</table>

Table 3.21: Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.
### Table 3.22: Errors for various disparity measures. The original cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.

<table>
<thead>
<tr>
<th>Image of Fig 3.27</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>77.64%</td>
<td>72.22%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>78.22%</td>
<td>74.09%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>78.80%</td>
<td>74.09%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>82.85%</td>
<td>79.51%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>82.46%</td>
<td>78.25%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>83.62%</td>
<td>80.63%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>79.83%</td>
<td>84.20%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>83.23%</td>
<td>79.03%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>81.88%</td>
<td>77.70%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>82.65%</td>
<td>78.22%</td>
</tr>
</tbody>
</table>

### Table 3.23: Edge detection errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.

<table>
<thead>
<tr>
<th>Image of Fig 3.28</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>72.18%</td>
<td>74.14%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>50.36%</td>
<td>54.51%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>74.18%</td>
<td>73.34%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>62.54%</td>
<td>57.30%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>59.63%</td>
<td>58.59%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>57.09%</td>
<td>60.05%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>57.81%</td>
<td>56.67%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>75.63%</td>
<td>72.14%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>79.27%</td>
<td>75.51%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>50.36%</td>
<td>50.62%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>73.09%</td>
<td>70.24%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>69.63%</td>
<td>65.32%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>85.09%</td>
<td>83.69%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>63.45%</td>
<td>59.61%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>55.09%</td>
<td>53.88%</td>
</tr>
</tbody>
</table>
### 3.8. Overview of the Results

<table>
<thead>
<tr>
<th>Image of Fig 3.29</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>33.94%</td>
<td>39.33%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>38.24%</td>
<td>38.41%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>53.31%</td>
<td>56.61%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>54.96%</td>
<td>55.17%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>23.84%</td>
<td>23.84%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>51.98%</td>
<td>52.38%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>34.78%</td>
<td>35.18%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>62.74%</td>
<td>62.43%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>50.33%</td>
<td>70.19%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>26.82%</td>
<td>26.98%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>42.05%</td>
<td>35.42%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>43.70%</td>
<td>47.60%</td>
</tr>
<tr>
<td>(l)</td>
<td>Chi Square</td>
<td>52.98%</td>
<td>53.95%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>28.31%</td>
<td>22.89%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>43.54%</td>
<td>43.54%</td>
</tr>
</tbody>
</table>

Table 3.24: Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear schedule for $\alpha$ with $\Delta\alpha = 0.01$. The common set of thresholds are used.

<table>
<thead>
<tr>
<th>Image of Fig 3.30</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>47.78%</td>
<td>50.23%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>0.0%</td>
<td>0.0%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>49.50%</td>
<td>49.50%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>24.50%</td>
<td>24.50%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>86.13%</td>
<td>86.55%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>49.00%</td>
<td>48.35%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>14.60%</td>
<td>16.86%</td>
</tr>
<tr>
<td>(c)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>22.46%</td>
<td>24.33%</td>
</tr>
<tr>
<td>(f)</td>
<td>Linear Correlation Coefficient</td>
<td>52.22%</td>
<td>57.26%</td>
</tr>
<tr>
<td>(i)</td>
<td>Chi Square</td>
<td>50.24%</td>
<td>36.87%</td>
</tr>
<tr>
<td>(l)</td>
<td>Contra-Harmonic Filter</td>
<td>82.67%</td>
<td>68.60%</td>
</tr>
<tr>
<td>(o)</td>
<td>Joint disparity</td>
<td>54.63%</td>
<td>63.96%</td>
</tr>
</tbody>
</table>

Table 3.25: Errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta\alpha = 0.01$. The common set of thresholds are used.
Chapter 3. Boundary Detection as a Global Optimization Problem

<table>
<thead>
<tr>
<th>Image of Fig 3.31</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>72.06%</td>
<td>67.23%</td>
</tr>
<tr>
<td>(d)</td>
<td>Linear Correlation Coefficient</td>
<td>71.09%</td>
<td>65.15%</td>
</tr>
<tr>
<td>(g)</td>
<td>Chi Square</td>
<td>69.74%</td>
<td>64.15%</td>
</tr>
<tr>
<td>(j)</td>
<td>Contra-Harmonic Filter</td>
<td>71.86%</td>
<td>65.71%</td>
</tr>
<tr>
<td>(m)</td>
<td>Joint disparity</td>
<td>69.74%</td>
<td>65.07%</td>
</tr>
<tr>
<td>(b)</td>
<td>Kolmogorov-Smirnov Statistic</td>
<td>72.63%</td>
<td>67.03%</td>
</tr>
<tr>
<td>(e)</td>
<td>Linear Correlation Coefficient</td>
<td>78.03%</td>
<td>72.13%</td>
</tr>
<tr>
<td>(h)</td>
<td>Chi Square</td>
<td>72.44%</td>
<td>65.59%</td>
</tr>
<tr>
<td>(k)</td>
<td>Contra-Harmonic Filter</td>
<td>50.28%</td>
<td>49.26%</td>
</tr>
<tr>
<td>(n)</td>
<td>Joint disparity</td>
<td>73.98%</td>
<td>66.15%</td>
</tr>
</tbody>
</table>

Table 3.26: Edge detection errors for various disparity measures. The modified cost function is used with the implicit dictionary and a linear cooling schedule for $\alpha$ with $\Delta \alpha = 0.01$. The common set of thresholds are used.

The following conclusions can be drawn from the quantitative analysis.

- Tables 3.7 and 3.8 give the under and over detection errors for Geman et al's approach and for Geman et al's approach with the modified cost function respectively. The Kolmogorov-smirnov statistic is used as disparity measure. We observe that there is some improvement in the segmentation errors for 4 out of the 6 segmentation results when the modified cost function is used.

- Tables 3.9 and 3.10 list the under and over detection errors when different disparity measures and the joint disparity measure are used with the original cost function. The explicit dictionary of figure 3.3 is used for penalization of illegal boundary patterns and a linear schedule for $\alpha$ with $\Delta \alpha = 0.02$ is used for penalization. We observe that the under/over detection errors are least (except for 3.10(k)) with the joint disparity measure. This confirms the benefit of using multiple statistics instead of a single statistic.

- Tables 3.11 and 3.12 give the errors for different disparity measure and joint disparity measure when the modified cost function is used. The explicit dictionary of figure 3.3 is used for penalization of illegal boundary patterns and
3.8. Overview of the Results

A linear schedule for $\alpha$ with $\Delta \alpha = 0.02$ is used for penalization. Again the segmentation errors are least when the multiple statistic is used. The segmentation errors with the modified cost function and the multiple statistics are less than the errors obtained with the old cost function and the multiple statistics for most of the images. Tables 3.10 and 3.12 also show that the errors are least (except for 3.10(k) and 3.12(k)) when the multiple statistics is used for segmentation. However the errors are not consistently lower with the modified cost function than what we have with the old cost function.

- Tables 3.13 and 3.14 list the under and over detection errors for various disparity measures and the joint disparity measure when the original cost function is used with the implicit dictionary and a linear schedule for penalization. Table 3.15 list errors when the linear schedule is replaced by a parabolic schedule. We notice that the segmentation errors are least in all the three cases with the multiple statistics. The comparison of tables 3.9 and 3.13 shows that the implicit dictionary does not give less error than the explicit dictionary, in all the three images of Fig 3.9 and 3.14 when a linear schedule with $\Delta \alpha = 0.02$ is used. This is the same schedule we used with experiments with the explicit dictionary. The fact that the results are worse can be justified by the fact that we have very few penalty configurations in the explicit dictionary while we have many penalty configurations in the implicit dictionary. The constraints should always be imposed on the model (data) initially slowly and afterwards (in our case after few iterations of annealing) rapidly. As now I consider many illegal boundary configurations in my implicit dictionary, I adopted a slower linear schedule with $\Delta \alpha = 0.01$. The under/over detection errors (see table 3.14) obtained when this slower schedule used are lower than those obtained with the explicit dictionary (see table 3.9). The results with the parabolic schedule (see table 3.15) again confirm the merit of the multiple statistics but the under and over detection errors are
not lower than what we had with the explicit dictionary. The results (see figure 3.16 (a), (h), (k), and (l)) have missing boundaries which indicate that the parabolic schedule imposes the constraints rapidly because of its exponential nature which causes the cost function to be trapped in a local minimum.

- Although the errors obtained with multiple statistics and implicit dictionary are least for Fig 3.15, there is not a great improvement in the errors which were obtained with individual statistics. The reason for this is that both our cost functions incorporate only horizontal and vertical boundary segments and hence they approximate any edge which is diagonal or antidiagonal by either vertical or horizontal segments. The errors for images of Fig 3.20 (see table 3.16) where only vertical or horizontal edges are present, show a significant improvement when multiple statistics are used with the original cost function, implicit dictionary and a linear schedule for $\alpha$ with $\Delta \alpha = 0.01$. These errors are less than those obtained with the explicit dictionary (see table 3.10). The relatively lower errors with the implicit dictionary indicate that it is better to promote boundary patterns which lead to boundary formation instead of penalizing the illegal boundary patterns. It also shows that a priori information about textured images is not required for deciding legal and illegal patterns for segmentation.

- Tables 3.17 and 3.18 give the segmentation errors when the modified cost function is used with the implicit dictionary and a linear schedule with $\Delta \alpha = 0.01$. The segmentation errors in table 3.17 show the importance of multiple statistics but the segmentation errors for the image of figure 3.21(o) with multiple statistics are higher than those with the Contraharmonic filter (see figure 3.21(l)). Although tables 3.17 and 3.18 show the benefit of using multiple statistics but the errors obtained with multiple statistics are not lower than what we obtained with the original cost function, the implicit dictionary (see table 3.14) and the multiple statistics.
3.8. Overview of the Results

- Tables 3.19–3.22 list the segmentation errors when the original cost function is used with the implicit dictionary and a linear penalization schedule of $\Delta \alpha = 0.01$. The common set of thresholds are used for determining the normalization constants used for each of the test images. Table 3.19 does not show the benefit of using multiple statistics but table 3.20 shows significant improvement in the errors with multiple statistics. The segmentation results with the Contraharmonic filters are not good which shows that some form of training phase is required with the Contraharmonic filter.

Table 3.21 shows that the multiple statistic gives the least error (except for image of figure 3.26(o)) as compared to other statistics. The image of figure 3.26(o) used the common set of thresholds derived from the database which does not include any homogeneous texture present in the image. The results with the Contraharmonic filter (see figure 3.26) are poor which again indicates that the Contraharmonic filter works better with the tuned thresholds.

Table 3.22 does not show the benefit of using the multiple statistics. We can see that the images of figure 3.27 have only diagonal or antidiagonal edges and so our algorithm approximates the edges by vertical or horizontal edges. We expect that the estimated errors can further be reduced if diagonal and antidiagonal edgels are also included in the cost function.

- Tables 3.23–3.26 list the segmentation errors for the images of figure 3.28–3.31 where the modified cost function is used with the implicit dictionary. The normalizing constants are derived from the database. The under/over detection errors are not least for all the three images of figure 3.28 with multiple statistics but errors are relatively lower than compared to our earlier results of figure 3.24 where the original cost function is used with the implicit dictionary and the common set of thresholds.

Table 3.24 shows the advantage of having the multiple statistics. Note that the segmentation errors here with the Contraharmonic filter for the image of
Figure 3.29(l) are least but the segmentation result shows missing boundaries. Also the segmentation errors achieved are less for most of the segmentation results when the modified cost function is used in place of the original cost function with the implicit dictionary.

Table 3.25 again shows that the segmentation errors can be reduced if the multiple statistic is used with the implicit dictionary. Although the image of figure 3.30(o) does not give the least error with multiple statistics it shows that if the size of the database is increased for finding the common set of thresholds, a better result can be achieved.

Table 3.26 although does not show the benefit of using the multiple statistic, the under/over detection errors achieved are significantly lower for images of figure 3.31 than our results of figure 3.27 (see table 3.22). This set of experiments shows that the modified cost function provides better interaction between label and data.

3.9 Summary

I discussed in this chapter the advantage of multiple statistics over a single static for texture segmentation. I showed by estimating under/over detection errors that multiple statistics with the implicit dictionary for penalization of illegal boundary labels helps in improving the segmentation of composite textures. I highlighted few shortcomings of earlier algorithm and then eliminated them in steps. I introduced the concept of a database for making the algorithm unsupervised to a large extent. I showed results of segmentation of composite textures.
Chapter 4

The Global Boundary Detection: Incorporating Edge Orientation

4.1 Introduction

Texture segmentation can be achieved in many different ways. My main aim here beyond developing another approach for texture segmentation is to incorporate edge orientation.

In order to achieve the above requirements, I need to associate edgel labels and disparity strengths with individual label sites rather than with pairs of label sites as in chapter 3. I present here a different topology for label lattice which includes labels with 4 different orientations. I also present a different segmentation scheme which takes into account the orientation of edgels alongside their magnitudes.

In section 4.2, I describe the relevant theory for the problem formulation.

In section 4.3, I present the extended virtual dictionary which also incorporates diagonal and anti diagonal edgels. I present the segmentation scheme used, in section 4.4.

I present the segmentation results in section 4.5. In section 4.6, I give the overview of the segmentation results in terms of their under/over detection errors and discussion. I give summary of the work in section 4.7.
4.2 Theory

Consider the dual grid of pixels and edgels as shown in figure 4.1.

Every edgel \((i,j)\) carries a line label, that consists of a strength value \(x_{i,j}\) and an orientation value \(\theta_{i,j}\). At any level resolution \(l\), I consider the four types of windows shown in figure 4.2 for the calculation of disparity values associated with any edgel. Let \(H, V, D_1\) and \(D_2\) denote horizontal, vertical, diagonal, and anti-diagonal disparity values associated with any edgel respectively.

Inside every window, I calculate three statistics namely the Kolmogorov-Smirnov...
statistic, the linear correlation coefficient, the Chi-square value and the Contra-
harmonic filter value. The values of each statistic inside the two panels of the same
window are compared to produce a value for the disparity measure (see figure 4.2).
The disparity value for the Contraharmonic filter is calculated in a slightly different
way than the earlier approach. Here, I assign the average value of the Contra-
harmonic filter of the two nearest horizontal, vertical, diagonal and antidiagonal
edgels present on either side of the central edgel \((i, j)\) to the horizontal, vertical,
diagonal and antidiagonal disparity values for the central edgel \((i, j)\) respectively. I
calculate the normalized disparity values corresponding to each individual statistic
and choose the maximum disparity value for that edgel. This way I calculate four
disparity values for each edgel. Thus at the end, every edgel carries 4 disparity val-
ues corresponding to the four blocks of figure 4.2: \(\Delta_{Hi,i,j}, \Delta_{Vi,i,j}, \Delta_{Di,i,j}\) and \(\Delta_{Di2,i,j}\).

At resolution level \(l\), I consider all the edgels that are \(2^l\) units apart.

The following cost function is used:

\[
U^l = \sum_i \sum_j (A + B + C + D + U_2^l) \tag{4.1}
\]

where

\[
A = \{x_{i,j}^2 - 2x_{i,j}\phi_H(i,j)(g)\} \cos(\theta_{i,j}) \tag{4.2}
\]

\[
B = \{x_{i,j}^2 - 2x_{i,j}\phi_V(i,j)(g)\} \sin(|\theta_{i,j}|) \tag{4.3}
\]

\[
C = \{x_{i,j}^2 - 2x_{i,j}\phi_D1(i,j)(g)\} \cos(0_{i,j} - 45^0) \tag{4.4}
\]

\[
D = \{x_{i,j}^2 - 2x_{i,j}\phi_D2(i,j)(g)\} \cos(0_{i,j} + 45^0) \tag{4.5}
\]

\(U_2^l\) is the contribution from the virtual dictionary penalty terms.

where \(\phi(\cdot)\) is the linear function used in equation 3.40 and can take values between
0 and 1 inclusive. \(U_2^l\) is the contribution from the virtual dictionary penalty terms.
Label $x$ is either 0 or 1 and $-90^\circ < \theta_{i,j} \leq 90^\circ$. As $\theta_{i,j}$ may take negative values, I had to use its absolute values in equation 4.3 so that deviation away from $\theta = 90^\circ$ is treated the same way either it is towards the diagonal or the antidiagonal directions.

The variable $\theta_{i,j}$ representing label orientation may take continuous values, but in practice I discretize it. For example, in my experiments I use $\theta_{i,j}$ to be $-45^\circ, 0^\circ, 45^\circ,$ and $90^\circ$. Finer quantization of orientation may be needed in general but such restriction to four orientations is made for computational efficiency of the algorithm, and is sufficient for discriminating many textures.

### 4.3 Virtual Dictionary

The virtual dictionary used here is different from the one used earlier in section 3.6 (see figure 3.13). As my cost function now has two variables $x_{i,j}$ and $\theta_{i,j}$, each entry in the dictionary incorporates two terms corresponding to edgel magnitude and edgel orientation. Figure 4.3 shows an arbitrary dictionary entry and its corresponding new dictionary entry. Example entries of such a dictionary are shown in Figure 4.4.

![Figure 4.3: A dictionary entry and the corresponding new dictionary.](image)

The total number of permissible configurations in this dictionary entries increases to 53 as compared to the 31 in the previous virtual dictionary of figure 3.13.
4.4 Estimation of the Penalty Term

Let us assume a local configuration of the form shown in figure 4.5. When the strength of an edgel is zero, its orientation is undetermined. The orientation of weak edgels is also of low significance as such edgels are probably due to noise.

I assume therefore that the weaker the edge, the least important the value of orientation is. The stronger the edge, the more weight the orientation value should carry.

From each entry of the dictionary, I create then a vector by stacking the columns of the entry one under the other, and by using \( x_0 \) instead of \( \theta \). In order to make \( x_0 \) and \( x \) to be of the same order of magnitude, I measure \( \theta \) in radians i.e.

\[
0 \leq x \leq 1 \quad \text{and} \quad -1.57 < \theta < 1.57.
\]

So from the example dictionary entry of figure 4.3, I create the following vector:
Chapter 4. The Global Boundary Detection: Incorporating Edge Orientation

\[ V_D = (1, -\pi/4, 0, 0, 0, 0, 0, 0, 1, -\pi/4, 0, 0, 0, 0, 0, 0, 1, -\pi/4) \tag{4.6} \]

From the example configuration of figure 4.5, I create the following vector:

\[ V_x = (0.6, \frac{30}{180}\pi \times 0.6, 0.2, \frac{45}{180}\pi \times 0.2, 0.1, \frac{70}{180}\pi \times 0.1, 0.3, \frac{30}{180}\pi \times 0.3, 0.7, \frac{45}{180}\pi \times 0.7, \frac{60}{180}\pi \times 0.2, 0.2, \frac{25}{180}\pi \times 0.2, 0.1, \frac{20}{180}\pi \times 0.1, 0.8, \frac{50}{180}\pi \times 0.8) \tag{4.7} \]

This is only an example, as in practice I assign only binary values and one of 4 possible orientations to edgels.

Once the dictionary vectors have been created, I compare the configuration vector \( V_x \) with all the dictionary vectors in turn. To avoid expensive computations I simply define the difference of the two vectors using the Manhattan metric, ie I sum up the absolute values of the differences of individual components. Then I choose the minimum such value over all the dictionary entries. This minimum value \( V_m \) is taken as the penalty term value. To have some control over it, I use a constant \( \alpha \) with which I multiply it, before I add it to the cost function. Thus the term \( U_2 \) of the
cost function is given by:

\[ U_2^l = \alpha \times V_m \]  

(4.8)

I have described in section 4.5 how the value of \( \alpha \) is to be calculated.

I optimize the cost function using the simulated annealing optimization technique [39] [40]. At the end of optimization I assign to each edgel a \( \theta_{i,j} \) and \( x_{i,j} \).

**4.5 Experimental Results**

Experiments are performed for boundary estimation on composite textures at level resolution \( l = 3 \). The same set of features is used as discussed in chapter 3. At level resolution \( l = 3 \), edgels are \( 2^3 \) units apart. I have taken \( 22 \times 22 \) blocks of pixels for the Kolmogorov-Smirnov statistic, the linear correlation coefficient and the Chi square test. The two blocks of data are used to find the disparity value by comparing their corresponding histograms over 8 bins. For keeping the total number of pixels approximately the same in vertical/horizontal and the diagonal blocks, I have taken \( 15 \times 15 \) blocks of pixels for diagonal blocks. For the Contraharmonic filter only one block of \( 8 \times 8 \) is considered. The maximum normalized disparity values are calculated and are assigned to \( H, V, D_1 \) and \( D_2 \) for each edgel. The normalization constants are derived using \( 21 \times 21 \) blocks of pixels in the training phase. I found experimentally that the common set of normalization constants for each of the statistics derived earlier were too low to give good segmentation with this scheme and therefore I adopted a higher set of thresholds again derived from the database of figure 3.23. This new common set of thresholds are derived by taking the 99 percentile of all horizontal and vertical disparity values for each homogeneous texture and then averaging the values (see equation 3.51). Table 4.1 lists the common set of values for each of the statistics.
4.5.1 Finding an initial label assignment

Before starting the relaxation, I assign initial values to label magnitude and label orientation as follows.

**Initial edgel magnitudes**

The theory of simulated annealing says that one can start from an arbitrary configuration for label lattice and keep on updating the label lattice as $k \to \infty$ where $k$ is the number of iteration of label lattice for obtaining a global optimum solution. But in practice, a better initial guess helps in finding a global optimum solution.

Edgels are assigned a magnitude according to the strength of the disparity values. If any of the disparity values $\phi()$ is greater than 0.6 then I assign that particular edgel a value one. The value of 0.6 was selected by trial and error. Any value in the range 0.6-0.7 works well.

**Initial edgel orientations**

The four disparity values $(H, V, D_1, D_2)$ for each edgel $(i, j)$ at each level resolution are compared among themselves.

If the horizontal disparity value $H$ is maximum then I assign zero degree orientation to this edgel. I convert all degrees into radians. If the vertical disparity value $V$ is maximum then I assign 90 degrees (1.57 radians) to that edgel. If the diagonal disparity value $D_1$ is maximum then assign 45 degrees (0.785 radians) to that edgel.
4.5. Experimental Results

If the diagonal disparity value $D_2$ is maximum then I assign -45 degrees (-0.785 radians) to that edgel.

A modified version of simulated annealing is used for optimization. Labels are either 0 or 1 and their orientations can be 0°, 45°, -45° and 90°. In conventional simulated annealing, a new state of the whole configuration is created by simply changing the label of a single site. This approach was tried here and found to be very slow. So, a modified version was adopted: instead of changing the label of a single site at a time, a whole 3 x 3 tile of labels are replaced by a tile from the dictionary. The criteria for accepting or rejecting the replacement are the same as for ordinary simulated annealing.

The logarithm cooling schedule of equation 3.7 is used and $c_c$ is set to 1.0. Any value of $c_c$ in the range 0.1 to 1.0 works well. The value of $c_c$ decides the starting temperature of the cooling schedule. The value of $c_c$ should be selected such that for the first few iterations of annealing many unacceptable configurations (configurations having higher energy) are accepted by the Metropolis criteria and then after a few iterations, i.e. at lower temperature, very few unacceptable configurations are accepted and finally none of the unacceptable configurations are accepted.

I replaced the linear schedule for penalization used in chapter 3 with an adaptive schedule in which the value of $\alpha$ is derived from the ratio of the total value of data terms over the total value of penalty terms of the cost function. The value of controlling factor $\alpha$ for the penalization of illegal configurations is calculated for each iteration of annealing as follows.

$$\alpha = k_{v} \times \frac{\sum_i \sum_j A + B + C + D}{\sum_i \sum_j U_2^i}$$

where

$k_v$ is a constant which controls the contribution of penalty terms with respect to the data term in simulated annealing updating process. A value of $k_v$ in the range 0.6
-0.8 works very well.

For the first iteration, the value of $\alpha$ is calculated from the initial lattice of edgels derived from the strength of individual disparity values at each edgel as described above. The value of $\alpha$ is kept constant for one complete iteration of the image lattice. For the next iteration, a new value of $\alpha$ is derived from the lattice configuration obtained after the previous iteration.

Experimental results for the level resolution of 3 are shown in figures 4.6–4.8.

### 4.6 Overview of the Results and Discussion

I calculate under and the over detection errors for each of the segmentated results I obtained and are listed below in table 4.2.

<table>
<thead>
<tr>
<th>Image</th>
<th>Method</th>
<th>Under detection error</th>
<th>Over detection error</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6 (a)</td>
<td>Multiple statistics</td>
<td>87.81%</td>
<td>88.04%</td>
</tr>
<tr>
<td>4.6 (b)</td>
<td>Multiple statistics</td>
<td>73.09%</td>
<td>74.56%</td>
</tr>
<tr>
<td>4.6 (c)</td>
<td>Multiple statistics</td>
<td>73.09%</td>
<td>74.17%</td>
</tr>
<tr>
<td>4.6 (d)</td>
<td>Multiple statistics</td>
<td>97.68%</td>
<td>98.03%</td>
</tr>
<tr>
<td>4.6 (e)</td>
<td>Multiple statistics</td>
<td>98.50%</td>
<td>98.38%</td>
</tr>
<tr>
<td>4.6 (f)</td>
<td>Multiple statistics</td>
<td>98.01%</td>
<td>97.77%</td>
</tr>
<tr>
<td>4.7 (a)</td>
<td>Multiple statistics</td>
<td>100.00%</td>
<td>100.00%</td>
</tr>
<tr>
<td>4.7 (b)</td>
<td>Multiple statistics</td>
<td>99.50%</td>
<td>99.21%</td>
</tr>
<tr>
<td>4.7 (c)</td>
<td>Multiple statistics</td>
<td>99.50%</td>
<td>98.99%</td>
</tr>
<tr>
<td>4.7 (d)</td>
<td>Multiple statistics</td>
<td>69.45%</td>
<td>71.55%</td>
</tr>
<tr>
<td>4.7 (e)</td>
<td>Multiple statistics</td>
<td>71.67%</td>
<td>69.26%</td>
</tr>
<tr>
<td>4.7 (f)</td>
<td>Multiple statistics</td>
<td>64.73%</td>
<td>66.19%</td>
</tr>
<tr>
<td>4.8 (a)</td>
<td>Multiple statistics</td>
<td>75.72%</td>
<td>73.29%</td>
</tr>
<tr>
<td>4.8 (b)</td>
<td>Multiple statistics</td>
<td>79.76%</td>
<td>77.22%</td>
</tr>
<tr>
<td>4.8 (c)</td>
<td>Multiple statistics</td>
<td>73.21%</td>
<td>73.63%</td>
</tr>
<tr>
<td>4.8 (d)</td>
<td>Multiple statistics</td>
<td>71.09%</td>
<td>69.01%</td>
</tr>
</tbody>
</table>

Table 4.2: Errors with the multiple statistics for various images. Errors are estimated over a window of 3 x 3. An adaptive cooling schedule for $\alpha$ is used.

I present here the comparative study of the algorithm in chapter 3 and this new
Figure 4.6: Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ in size.
Figure 4.7: Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ in size.
Figure 4.8: Textural boundary estimation in images. Results at $l = 3$. Edgels are 8 units apart. All images are $256 \times 256$ pixels in size.
• The algorithm of the previous chapter incorporated only horizontal and vertical boundary segments and therefore the implicit dictionary was having very few legal boundary patterns. Thus the algorithm penalizes many illegal boundary patterns and leads to global optimum of the cost function. The algorithm worked very well with low values of common set of thresholds. The algorithm presented in this chapter incorporated diagonal and anti-diagonal edgels alongside the horizontal and vertical edgels. This algorithm is based on the logic that the data should penalize the illegal boundary patterns and the extended implicit dictionary will govern the edges to have suitable directions depending on the disparity values associated with different edgels. As the algorithm is data based, I took a relatively higher common set of thresholds in order to eliminate the unwanted boundary patterns.

• The result of figure 4.7(c) shows missing boundaries because of selecting high values for the common set of thresholds.

• There are two quantitative parameters on which the performance of the two algorithms can be judged.

1. The under and over detection errors estimated for each of the segmented results shown in table 4.2 are large as compared to the errors achieved with the previous technique of chapter 3. The large errors obtained with the images of figure 4.6 and 4.7(a)(b)(c) can be justified as follows:

The results obtained with the proposed technique are at level resolution of 3 which means that the labels are $2^3$ units apart. Thus if we start relaxation at $(i, j)$ then every $8^{th}$ label takes part in the relaxation process. Now if any true edge lies on this sparse label lattice then that edge is correctly placed on the segmented output image. If the true edge point say at $(i, j)$ does not lie on this sparse label lattice, then the nearest label on the label lattice
located either at \((i - 8, j), (i, j - 8), (i + 8, j)\) or \((i, j + 8)\) is labelled as edge point, depending on the disparity values associated with those edgels by the algorithm. The under detection errors are calculated over a window of \(3 \times 3\) labels centered at the true edge point. Thus if any true edge point does not lie on the label lattice, the error counting program does not find any edge point in the window of \(3 \times 3\) labels and this leads to high errors. For example for the images of figures 4.7(a),(b),(c), true edges do not lie on the label lattice and therefore under/over detection errors are very large. Similarly the over detection errors are high.

The under and over detection errors for the cross images of figures 4.7(d),(e),(f) and 4.8 which include only diagonal or anti diagonal edges are relatively low and are close to the errors I achieved with the previous algorithm.

2. The second quantitative parameter is the cpu time required by the two algorithms. The average cpu time required by the algorithm of chapter 3 is 210 seconds while average cpu time required by this new algorithm is 270 seconds for an image of \(256 \times 256\) pixels. The experiments were run on a solaries machine. The time required in this new algorithm is more because:

1. I calculate two extra disparity values for diagonal and antidiagonal edgels.
2. I calculate the penalty term by taking into account the actual magnitude and orientation of the edgels which requires considerable amount of computation time.

It will not be very correct to compare their cpu times as the two algorithms work for different edgel resolution. The first algorithm work for label resolution of 5 where every \(5^{th}\) label takes part in the relaxation process while in the second algorithm the label resolution is 8 i.e every \(8^{th}\) label takes part in the relaxation.

I faced the following difficulties while running algorithm at level resolution of 2.
Chapter 4. The Global Boundary Detection: Incorporating Edge Orientation

- The disparity values associated with neighbouring edgels at the true edge at level resolution of 2 (edgels 4 units apart) are very similar and therefore the algorithm finds it difficult to locate the actual edge point. In other words, the statistical disparity measures are not sensitive enough to cope with very fine resolutions.

- The output of filter preprocessor assigns value one to many edgels around true edges, just because the disparity values at those edgels are above the threshold values. During relaxation when a tile of $3 \times 3$ labels is replaced by a new tile randomly taken from the dictionary, the new tile is selected if its penalty value is less than the penalty value of the old tile. Thus for the first few iterations, the decision of updating the tile is totally dependent on the contribution of the penalty term as the data terms do not play any major role. Although the simulated annealing optimization technique leads to the global optimum of the cost function, in practice there are many “globally” optimal solutions most of which are wrong.

4.7 Summary

In this chapter, I presented a texture segmentation algorithm based on multiple statistics and the virtual dictionary. The dictionary incorporated diagonal and antidiagonal boundary labels and was an extended version of our earlier implicit dictionary. I presented the segmentation scheme which takes into account the magnitude and the orientation of edgels for the constraint optimization. I showed results of segmentation for level of resolution $l = 3$. A comparative study was made between two algorithms in terms of their under/over detection errors and computation time.
Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, the problem of texture segmentation has been addressed. A review of existing texture segmentation techniques was carried out. This review was made in order to examine the existing techniques and their merits and demerits. We selected an existing texture segmentation technique based on a statistical approach which uses a global optimization strategy for texture segmentation post processing. The simulated annealing (SA) has the potential of bringing a smoothly varying energy function with multiple minima to its global minimum regardless of initial condition. Another advantage of such a stochastic relaxation algorithm was the design flexibility. SA can be implemented on Markov Random Fields (MRF) which provide a great measure of freedom in energy function definition. Thus complex decision rules (constraints) can be easily incorporated. One such texture segmentation technique by Geman et al [39] was developed. Ideally, any texture segmentation algorithm should segment such that there is one to one correspondence between the segmented output edgels and the ground truth edgels (derived from the input textured image manually) and so we focussed on finding the effectiveness of the algorithm by estimating the under and over detection errors for each segmented output image. The main objective of this work was how to modify this existing algorithm so as to significantly reduce the errors and modify it so that it has the
potential for unsupervised texture segmentation. The major shortcomings of the ex­isting algorithm we pointed out and the solutions we proposed can be summarized as follows:

1. The cost function proposed by Geman et al does not appear to express correctly faithfulness to the data. The faithfulness to the data is expected to be in the least square error sense. So, we proposed a modified cost function which also incorporated a bounded linear transformation as opposed to a unbounded transformation for disparity value estimation. The segmentation results we found with the two cost functions were quite similar which pointed us to look for the other shortcomings of their algorithm.

2. Geman et al used the Kolmogorov-Smirnov statistic in distinguishing different textures. We found that Kolmogorov-Smirnov statistic alone is not appropriate in distinguishing different textures. We suggested three extra disparity measures, namely the linear correlation coefficient measure, Chi-square measure and the Contraharmonic filter in addition to the Kolmogorov-Smirnov statistic. We were moti­vated by the observation that different statistics could discriminate between different textures and therefore we proposed the concept of joint disparity measure for accurate segmentation. We found that multiple statistics gives consistently better segmentation results than any single statistic [86]. Also the segmentation errors achieved with multiple statistics were least as opposed to when single statistic is used.

3. The basic shortcoming of their algorithm was that they explicitly defined penal­izable boundary configurations in which the inclusion of a penalty term was de­pendent on the test image on which boundary detection is to be performed. Thus defining these configurations explicitly makes the algorithm supervised. We adop­ted a more general approach which does not require any prior information about the input image. We created a dictionary which contained those label configurations that promote the boundary formation. Such a dictionary consisted of the set of all
5.1. Conclusions

3 × 3 permissible label configurations. The segmentation results we achieved when the original and the modified cost function are used with this implicit dictionary showed consistently improved results. The segmentation errors we achieved with multiple statistics and the implicit dictionary were least for most of the segmentation results as compared to when a single statistic was used with either the explicit or the implicit dictionary.

4. The normalization constant selection procedure proposed by Geman et al was not appropriate. We found that many boundaries were missing in the segmented results with their approach. Also the normalization constants were derived from the homogeneous textures present in the test image to be segmented which made their algorithm highly dependent on the prior knowledge of homogeneous textures present in the composite texture to be segmented. We incorporated the idea of a database of homogeneous textures for calculation of normalization constants which to a large extent made the algorithm suitable for unsupervised segmentation although the segmentation errors achieved were relatively larger.

5. We conclude that the segmentation errors are large for some test images even with the multiple statistics because both cost functions incorporated only vertical and horizontal boundary segments. The errors can further be brought down by including diagonal and anti-diagonal boundary segments along with horizontal and vertical segments in the cost functions.

We presented a different topology for label lattice incorporating edge orientation. A new segmentation scheme is introduced which takes into account edgels with four orientations (horizontal, vertical, diagonal and anti-diagonal) to overcome the shortcoming of the earlier algorithm. The segmentation results were shown for level resolution \( l = 3 \).
5.2 Future Directions

Throughout the thesis we addressed the problem of texture segmentation for grey level images. Coloured texture segmentation is a new area of research. The modified algorithm can further be extended to solving coloured texture segmentation problems with multiple statistics. A 3-dimensional histogram can be generated by observing the number of occurrences of the red, green, and blue sensor values of a colour images for disparity value calculations for any edgel on the image lattice.

Presently, we proposed the concept of multiple statistics with four different measures with a fixed set of features. The technique can further be explored to check and improve segmentation results by incorporating other disparity measures e.g. entropy, with different feature sets.

We introduced the concept of a database of just ten homogeneous textures for the estimation of the normalization constants (training phase). A large set of database for training phase could further make the algorithm suitable for a wider class of textures.

The proposed approach can be further developed for image database retrieval in which the training phase will estimate the under and over detection errors. Thus any test image having minimum errors with respect to the images of the database would be able to retrieve similar images from the database.


