

Surface-Based Segmentation of Volume Data Using Texture Features

by

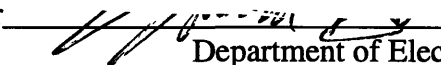
Edward H. Baik


Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degrees of
Bachelor of Science in Electrical Engineering and Computer Science
and Master of Engineering in Electrical Engineering and Computer Science
at the
Massachusetts Institute of Technology

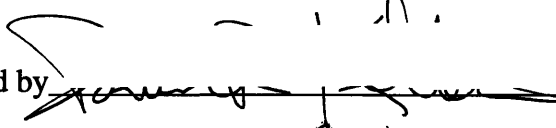
February, 1997

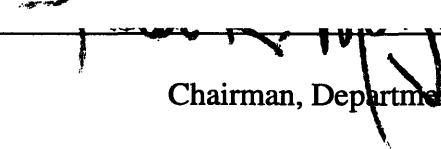
© Edward H. Baik, 1997. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce
and to distribute publicly paper and electronic copies of this thesis
and to grant others the right to do so.

Author 
Department of Electrical Engineering and Computer Science
February 7, 1997

Certified by 
Andrew B. Dobrzeniecki
Thesis Supervisor

Certified by 
Darlene R. Ketten
Thesis Supervisor

Accepted by 
Frederic R. Morgenthaler
Chairman, Department Committee on Graduate Theses

Surface-Based Segmentation of Volume Data Using Texture Features

by

Edward H. Baik

Submitted to the
Department of Electrical Engineering and Computer Science

February 7, 1997

In Partial Fulfillment of the Requirements for the Degrees of
Bachelor of Science in Electrical Engineering and Computer Science
and Master of Engineering in Electrical Engineering and Computer Science

Abstract

Much research has been devoted to the segmentation of 2D images which has long been a challenging problem due to the complexity of the images. Three-dimensional (3D) data sets representing spatial volumes, or volume data, may be produced by imaging modalities such as computed tomography (CT) and magnetic resonance imaging (MRI). Relatively recently, more effort has been devoted to the application of image segmentation to such 3D images. In 1987, Lorensen and Cline introduced the marching cubes algorithm which efficiently constructs surfaces, made up of a set of connected polygonal triangles, between constant gray level objects in given input data. This thesis proposes a method for the surface-based segmentation of volume data by modifying the marching cubes algorithm and incorporating texture features computed within defined local volumes. The texture features are computed from first-order statistical methods, 3D Fourier power spectra, 3D co-occurrence matrices, and 3D Gaussian Markov random field (GMRF) models. The surface-based segmentation method developed is tested on both synthetic textured images and regions of interest from an MRI image of *Phocoena phocoena* (harbor porpoise). The results are evaluated qualitatively and are found to be encouraging. Surfaces representing boundaries between different textures are approximately constructed in many instances.

Thesis Supervisor: Andrew B. Dobrzeniecki
Title: Lecturer, Harvard Medical School

Thesis Supervisor: Darlene R. Ketten
Title: Assistant Professor, Harvard Medical School

Acknowledgments

First of all, I would like to thank my thesis supervisors, Andy Dobrzeniecki and Darlene Ketten, for their guidance, help, and understanding. Without them, this thesis would not exist. I am very grateful to them for providing me with the opportunity to work on this project.

I spent my time working on this thesis in the Whitaker College of Biomedical Imaging and Computation Laboratory (WCBICL). Thanks to Professor Jacqueline Yanch who is the director of the lab, An Lu who I hope everything works out for, M.J. Belanger who I hope will get better, Aina Rogel who I hope continues to enjoy her experience here at MIT, and Melissa Lambeth who I hope will enjoy her future graduate school endeavors for their interest and kindness. Thanks also to Scott Cramer in Darlene Ketten's lab at the MEEI for his help and friendly attitude.

I would say that some people come to MIT with many expectations of what it will be like and that others come with very few or no expectations at all. I guess that I was sort of in the middle of both extremes. Overall, I've enjoyed the interesting experiences I've had during my five and a half years here. I've grown a lot and learned a lot and not just because of the academic aspects of life at MIT that began once upon a time in 26-100. I think a lot of the growth is also due to the people that I have had the opportunity to meet here and, of course, merely due to the irreversible fact of growing older.

I want to thank some of the people who have had an influence on my life here and who I consider my friends, especially those who I have lived with at the "house": Mark Eastley, Franz Busse, Sridhar Kalluri, Jeff Reneau, Mark West, Mike Chatwin, Andrew Beharrie, Chris Hirsch, Greg Thomas, Richard Sann and his family, Mike Thomas, Jordan Roberts, Matt Wall, Rick Patterson, Keith Fife, and Noah Olson. Special thanks go to Donald Greer, who has been in most of my classes, for working together many late nights on problem sets and for always seeming to possess a good nature and sense of humor even at 2 am; to Dan Theobald who is one of the few people I knew a while back who is still around and who possesses a good deal of patience and is an all-around great guy who actually likes seaweed; to Mark Spilker with whom I share more common thoughts and perspectives than you might think upon first observation and who is a wild man on the road bike and has a good work ethic even though he doesn't think it at times; to Pete Crompton who taught me words like jejunum, sphincter, and beano and who, may he rest in peace, I hope gets to fulfill all his hopes when he gets his M.D.; to Jason Olson who is most likely the funniest person I know and has the loudest, weirdest laugh I've ever heard and who has been a good person to talk with about a great many things; to Neil Jenkins with whom I've had many good, deep conversations about religion and who always seems to pop his head into my room, leave, and then return 30 seconds later.

Special thanks also go to Abbie Gouggerchian for his support and for helping me to pursue my educational goals. Lastly, I cannot express how much gratitude I have for my father, mother, and brother Howard. The story of how my father came to this country with only a hundred dollars in his hand reveals a journey that is nothing less than a miracle. My parents instilled in me a value for education and have constantly worked to provide me with opportunities they didn't have. I am deeply aware of what they have done for me, although I often feel that they think that I do not understand.

I don't think much inherently separates me from those individuals who have been dealt a bad hand of cards by life, except perhaps chance and circumstance. I need to count my blessings I guess; I hope this is something I never forget.

Contents

1	Introduction.....	9
1.1	Motivation and Objectives.....	9
1.2	Possible Applications.....	12
1.2.1	Acoustic Models of Marine Mammal Ear	13
1.2.2	Image-guided Surgery.....	14
1.3	Brief Summary of Results.....	15
1.4	Overview.....	16
2	Background.....	19
2.1	3D Image Segmentation: Existing Work and Current Approaches	19
2.2	The Marching Cubes Algorithm	23
2.3	Texture Segmentation	26
2.4	Neighborhood Definitions and Framework for Feature Computations	28
2.4.1	Neighboring Local Volumes for First-Order Statistics.....	29
2.4.2	Neighboring Local Volumes for Other Texture Features	31
2.5	Synthetic and Real Image Data Sets	32
2.5.1	Using GMRF Model to Create Textured Images.....	33
2.5.2	Using the Gaussian Distribution to Create Textured Images.....	34
2.5.3	MRI Image of a Porpoise.....	35
2.6	Summary	38
3	Texture Features and Our Approach.....	39
3.1	First-Order Statistical Methods.....	39

3.2	Gray Level Co-occurrence Matrix Features	43
3.2.1	2D Gray Level Co-occurrence Matrices	44
3.2.2	3D Gray Level Co-occurrence Matrices	47
3.3	Fourier Power Spectrum	50
3.3.1	2D Fourier Power Spectrum Features.....	51
3.3.2	3D Fourier Power Spectrum Features.....	55
3.4	Gaussian Markov Random Fields.....	57
3.4.1	2D GMRF Model.....	57
3.4.2	3D GMRF Model.....	60
3.5	Our Approach Using Texture Features	64
3.5.1	The Method Using a Feature Vector and a Similarity Measure	64
4	Experiments: Tests and Results	69
4.1	Synthetic and Real Volume Data Used.....	70
4.2	Standard Marching Cubes.....	73
4.3	Tests With Synthetic Images	76
4.3.1	Single Feature Tests.....	77
4.3.2	Using the Feature Vector with First-Order Statistical Features.....	82
4.3.3	Using the Feature Vector with 3D Co-occurrence Matrix Features	82
4.3.4	Using the Feature Vector with Fourier Power Spectrum Features	86
4.3.5	Using the Feature Vector with 3D GMRF Model Features.....	88
4.3.6	Using the Feature Vector with All Features	93
4.4	Tests with Real Images	99
4.4.1	Using the Feature Vector	101

4.5	Implementation Details.....	105
4.6	Summary.....	105
5	Conclusions.....	111
Bibliography	117

Chapter 1

Introduction

1.1 Motivation and Objectives

Image segmentation has been a topic of active research for the last thirty years. Segmentation can be regarded as the integral first step toward subsequent processing and analysis of an image and is still an unsolved problem despite the research effort that has been devoted to this field. The goal of image segmentation is to identify homogeneous regions in an image where the characteristics that determine homogeneity can be based on a number of factors that are context dependent.

To further describe image segmentation, it is necessary to address how an image is represented. In our case, two-dimensional (2D) images may be thought of as a finite 2D rectangular grid of sites where picture elements or pixels are located. Each pixel has an associated gray level value which may be represented as function of its position, $f(i,j)$. Analogously, we consider three-dimensional (3D) images as being a 3D grid of sites where voxels are located. Each voxel has a gray level value which may be represented as a function of its 3D location, $f(i,j,k)$. Each pixel or voxel gray level represents certain relevant properties associated with the location (i,j,k) in the modeled three-dimensional world depending on the method used to acquire the representation. For instance, gray levels may represent attributes of crop types in satellite images, distance estimates in range

images, x-ray attenuation of structures in computed tomography (CT) images, and proton density and mobility in magnetic resonance (MR) images.

The approaches to finding desired homogeneous regions in a 2D image by segmentation may be categorized into two groups, edge-detection methods and region-based methods. Edge-detection, or boundary-based, methods search for dissimilarities between pixels and thereby determine edges that distinguish boundaries between areas. Region-based methods search for similarities by which to merge pixels together into homogeneous regions. For 3D images, the approaches to segmentation are usually extensions of the 2D approaches to 3D in which the region-based approaches extract homogeneous volumes instead of areas, and the edge-based approaches identify surfaces forming volume boundaries. The main problem of segmenting 2D or 3D regions in images has been to find the suitable homogeneity properties [Monga, *et al.*, 1991].

Most of the research in image segmentation has focused on segmenting 2D images. Several papers showing the variety of techniques and results are available [Reed and Hans Du Buf, 1992; Bezdek, *et al.*, 1993; Haralick, 1979; Hu and Dennis, 1994; Geman, *et al.*, 1990]. Recently, however, more effort has been devoted to the application of image segmentation to 3D images (and to 3D image processing in general) due to the continuing development of imaging instruments, computer processing power, and graphics capabilities. 3D data sets representing spatial volumes, or volume data, arise in many scientific applications ranging from meteorological and astrophysical measurements to medicine, and it is likely that more applications will generate volume data in the future, continuing to provide information for further advances in many fields [Drebin, Carpenter, and Hanrahan, 1988]. In particular, access to the third dimension in medical imaging has

already provided valuable information for reparative surgery, radiotherapy treatment planning, and stereotactic neurosurgery [Joliot and Mazoyer, 1993].

More specifically, medicine has benefited greatly from imaging modalities such as x-ray computed tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET), all of which produce volume data that are typically stored as a sequence of parallel 2D slices. The top-level approaches to segmenting these volume data may be loosely categorized into two approaches. One approach applies various 2D segmentation techniques on each cross-sectional slice of the volume data [Muzzolini, Yang, and Pierson, 1994; Chakraborty, *et al.*, 1994; Joliot and Mazoyer, 1993]. Using 2D techniques on 3D data involves first segmenting each 2D image in the sequence of slices and then combining the results in some manner to get the 3D segmentation. Apart from not directly utilizing all of the information in the volume data, such 2D methods give rise to various ambiguity and connectivity problems between the 2D segmented planes [Joliot and Mazoyer, 1993; Cappelletti and Rosenfeld, 1989].

The other approach works directly with the volume data as a whole, thereby utilizing all of the 3D information. Recently, more effort has been devoted to this approach [Muzzolini, Yang, and Pierson, 1994; Liou and Jain, 1991; Cohen, *et al.*, 1992; Cline, *et al.*, 1990; Bomans, *et al.*, 1990; Ehrlicke, 1990; Aach and Dawid, 1990; Strasters and Gerbrands, 1991; Choi, *et al.*, 1989; Gerig, *et al.*, 1992].

In 1987, Lorensen and Cline presented the marching cubes algorithm. This surface construction algorithm produces high quality images by generating a set of connected triangles to represent 3D surfaces of structures in the input data. This surface representation allows for 3D rendering of structures with the appropriate visualization software.

This technique has become popular and widely used due to its practicality and simplicity.

At the heart of the marching cubes algorithm is a simple thresholding procedure which determines the location of the surfaces. The volume data space is viewed as being partitioned into cubical elements for which each cube consists of voxel values located at its eight vertices. Inside these cubical elements, the decision is made as to whether a surface intersects the cube by comparing voxel values along a cube edge with a determined threshold value. If a surface is determined to intersect the cube, polygonal triangles are used to approximate this surface-cube intersection.

This thesis proposes that perhaps the marching cubes technique can be modified to tackle the larger problem of image segmentation of volume data and investigates how well local statistical and textural properties of the volume data may replace this simple thresholding procedure for the purpose of segmentation. In particular, the objective is to use the textural features computed from first-order statistical methods, local 3D Fourier power spectra, 3D co-occurrence matrices, and Gaussian Markov random field (GMRF) models to determine surface boundaries between different tissues and structural objects in volume data produced by magnetic resonance imaging (MRI) or computed tomography (CT). In summary, this thesis presents an approach for the segmentation of volume data, both real and synthetic, based on textural and statistical properties within a marching cubes type framework.

1.2 Possible Applications

The results of segmentation may be used for many applications since segmentation

is the first step in many image processing tasks. This section discusses two possibilities that serve to illustrate the value of effective 3D image segmentation methods.

1.2.1 Acoustic Models of Marine Mammal Ear

The first application relates to developing models of ear function of marine mammals. In particular, many functional aspects of the whale (Order Cetacea) ear are not fully understood. There is no satisfactory model of hearing in Cetacea [Ketten, 1992]. Since whales must function in water, a light-limited environment, sound is a fundamental sensory and communication tool. Whales are the only mammals with acute ears that are fully adapted to underwater hearing, and they have the broadest acoustic range of any known mammal group [Ketten, 1994].

The order Cetacea has two suborders, Odontoceti (toothed whales and dolphins) and Mysticeti (baleen whales). All odontocetes tested to date echolocate; that is, they sense their environment by analyzing echoes from a self-generated ultrasonic signal of up to 200 kHz [Ketten, 1992]. Some mysticetes produce infrasonic signals that may be used for long-range communication or navigation, but observations are consistent with the assumption that they do not use ultrasonics for echolocation. Outer ear adaptations to an aquatic environment are extensive in cetaceans, and the mechanisms of sound production and reception in odontocetes are still being debated. It is presently believed that multiple paths exist for sound conduction [Ketten, 1994].

One primary path for ultrasonic signals in odontocetes is believed to be located in the lower jaw, where waxy tissues overlain by the mandibular bone provide a low imped-

ance path that can guide sound to the middle and inner ear. Another path may be a thin ovoid region near the posterior of the odontocete mandible called the “pan bone” [Ketten, 1992]. Recent MRI data has shown that this broader, laterally directed, funnel-shaped channel aligns with the middle ear and may act as a low frequency conduit [Ketten, 1994].

Similar tissue channels to the ear in mysticetes have not been identified. A good method for the three-dimensional segmentation of the ear and the specific acoustic tissue paths mentioned above would contribute to the development of various acoustical and mechanical models that could help elucidate the mechanisms behind the hearing capabilities of odontocetes and perhaps marine mammals in general.

1.2.2 Image-guided Surgery

Another potential application is found in image-guided surgery. The goal of an image-guided surgery system is to allow a surgeon to view a patient and at the same time display in exact alignment with that view all desired internal structures before executing each stage in a surgical procedure. Although there are many groups working on image-guided surgery, we mention one in particular [Peters, *et al.*, 1996]. The Computer Vision Group of the MIT Artificial Intelligence Lab has been working in collaboration with the Surgical Planning Laboratory of Brigham and Women’s Hospital to develop tools to support image-guided surgery [Grimson, *et al.*, 1996]. Such tools allow surgeons to visualize internal structures through an automated overlay of 3D reconstructions of internal anatomy superimposed on live video views of a patient.

These tools are needed for constructing 3D models, laser scanning, performing

registration, implementing enhanced reality visualization, and doing other tasks that make up an image-guided surgery system. Creating segmentations of the input data (mostly MR scans) is embedded in the process of creating the 3D models. The initial stages of the system that the AI Computer Vision Group at MIT has implemented involves automatic intensity-driven labelling of MRI voxels by type, segmentation of these labeled voxels into distinct anatomical structures, and visualization by rendering of the anatomical structures. Clearly, 3D image segmentation methods play an important role in the performance of such a system.

1.3 Brief Summary of Results

Chapter 4 of this thesis will detail the experimental results of using the surface-based segmentation algorithm which incorporates texture features. But in order for the reader to have a sense of how our segmentation method performs, here a brief description of the general results is given in a manner that does not necessitate the explanation of implementation details that have not yet been presented. It should be noted that all results are qualitative because the computed polygonal surfaces are not closed and are therefore not amenable to quantitative assessment.

Testing was performed on a group of synthetic images and also on MRI data of a porpoise. The algorithm was applied using different combinations of texture features. Results for the synthetic images varied from poor to encouraging. In some cases, visible polygonal surfaces were constructed along the boundaries separating one volume of tex-

ture from another volume of texture. In other cases, the algorithm produced no visible surfaces separating volumes of texture, the output being randomly scattered polygons.

Two 3D regions of interest were selected from the MRI data of a harbor porpoise which had sufficient volumes of simple textures for testing. In a qualitative sense, the tests were encouraging in that the simpler 3D region of interest resulted in the construction of a visible surface between the different textured regions. The resulting polygonal surface is similar to those surfaces constructed using the synthetic data. The more complex 3D region which was extracted from the original MRI data generally yielded scattered polygons, demonstrating that the current approach is not sufficient to segment complex textured images.

1.4 Overview

This chapter gives the overall picture of the main ideas involved in this thesis. Chapter 2 gives further background information and continues a general presentation of ideas related to the issues at hand. Chapter 2 contains further discussions of existing work on 3D image segmentation, the marching cubes algorithm, textures and texture segmentation, neighborhood local volume definitions, and the synthetic and real images used for testing.

Chapter 3 describes each texture feature used and its associated implementation in sequence. In particular, it explains the first-order statistical features, the 3D co-occurrence matrix based features, the 3D Fourier power spectrum features, and the Gaussian Markov random field (GMRF) model based features. Chapter 3 also describes the manner in

which features are combined into the general algorithm's framework. Chapter 4 presents experimental results of applying these individual and combined features to synthetic data and real data. Finally, chapter 5 suggests possible future work.

Chapter 2

Background

This chapter covers the background for the ideas used in this thesis. In the first section, we present a summary of some recent work in 3D image segmentation. Section 2.2 discusses the original marching cubes algorithm and how it constructs surfaces from the input data using a simple user specified threshold. Section 2.3 gives an overview of various methods and features used to characterize textures for analysis and segmentation. Section 2.4 discusses local volumes within which various texture features may be computed. The last section describes how synthetic images were generated and shows examples of both synthetic images and real images.

2.1 3D Image Segmentation: Existing Work and Current Approaches

As indicated earlier, most research has addressed the problem of segmentation for 2D images although recently work has been done with 3D images. This section describes some of this recent work and the approaches to 3D image segmentation. This discussion covers merely a sampling of the research that has been done and is by no means exhaustive.

In 1990, Cline *et al.* described a 3D segmentation method that classifies 3D MR images of the head into separate tissue types and constructs a surface model for each tissue

by using two sets of MRI data acquired with different contrast protocols. The user must identify the different tissue classes and sample a sufficient number of points in appropriate locations. Then, without further user interaction, their method calculates a probability distribution for each tissue, creates a feature map of the most probable tissues, segments and smooths the data, and finally uses a connectivity algorithm to extract the surfaces of interest from the segmented data. This process allows for the selection of multiple surfaces for subsequent display. Cline *et al.* used MR images from patients with normal head anatomy and from patients with abnormalities such as multiple sclerosis lesions and brain tumors to demonstrate that their technique is able to adequately segment such lesions and tumors as well as segment normal tissues.

In 1990, Aach and Dawid presented a region-based three-stage method for segmenting MRI data sets of the head into 3D regions corresponding to different brain tissue classes, fluid containing structures, and skull. Their method first partitions the volume into a number of homogeneous regions based on global and local gray level statistics. These regions are then grouped into different categories by a Bayes classifier. Finally, knowledge of the spatial relationship between objects is used to correct possible misclassifications. Aach and Dawid assumed from the beginning that volume data is composed of 3D regions whose internal gray values may be described by a Gaussian random field model. Their method was applied to several MR images of the head and the resulting segmentations shown, but there was little qualitative or quantitative discussion of the results.

In 1991, Liou and Jain presented an algorithm which can segment 3D images into coherent volumes by α -partitioning and volume filtering in such a way that the gray level variations within each volume could be described by a regression model. Their model

allowed them to combine a probabilistic approach, in which a 3D image is modeled as a 3D random field, with a functional approach, in which the underlying gray level distribution is captured by using a family of smooth functions. Experimental tests were run on one CT data set and on two intensity image sequence data sets. Experimental results demonstrated that their algorithm produced successful segmentations in general but that for highly textured areas of the images the algorithm did not perform well.

Cohen *et al.* (1992) tried a different approach by using a deformable 3D shape model to extract reliable surfaces in 3D images. Their deformable model allowed them to characterize the boundaries in a 3D image by describing these boundaries as a set of surfaces. These surfaces were found by minimizing an appropriate energy function. The power of their approach was shown by a set of experimental results on some synthetic images and on some complex 3D medical images which include an MR image of a human heart and a human head. By using a variational approach and a conforming finite element method to express surfaces in a discrete basis of continuous functions, Cohen *et al.* were able to reduce computational complexity and ensure better numerical stability. The strength of their technique is that their method provides an analytical representation of the extracted surfaces which may be utilized for other computational purposes.

In 1992, Gerig *et al.* proposed the application of multivariate statistical classification techniques to the segmentation of dual-echo MR volume data of the human head. Both supervised methods (which require training and user interaction) and unsupervised methods (which are fully automated) were tested in segmenting the MR head volume data into different tissue types such as gray matter, white matter, and fluid spaces. For their supervised method, Gerig *et al.* used both a maximum likelihood classifier and a non-para-

metric Parzen window classifier to obtain cluster statistics for segmentation. For their unsupervised method, Gerig *et al.* applied two different clustering algorithms to obtain cluster statistics for comparison with the cluster statistics of the supervised method. It was found that the estimated parameters were very similar to those obtained by supervised parameter learning. Their algorithms were applied to a clinical study comprising sixteen volume acquisitions of the human brain. Their methods illustrated the robustness of their method in segmenting white brain matter, gray brain matter, and cerebrospinal fluid. The supervised method had a 93.8% successful classification rate, and the unsupervised method had a 87.5% success rate.

In 1993, Joliot and Mazoyer proposed an almost fully automated method for the 3D segmentation and interpolation of anisotropic MRI brain data for improved definition of the brain surface. This segmentation process involved three distinct steps. First, a gray level thresholding of the white and gray matter tissue was performed on the raw brain MRI data. Then, an automatic global white matter segmentation was performed with a global 3D connectivity algorithm which takes into account the anisotropy of the MRI voxel. Lastly, the gray matter was segmented with a local 3D connectivity algorithm. Joliot and Mazoyer used mathematical morphology tools (which are used to manipulate shapes of objects in an image) to interpolate the white matter and brain tissues in adjacent slices, producing binary representations of both white and gray matter which were used for 3D surface rendering. Their method was applied to four MRI data sets and compared to a manual segmentation. The results were similar, but their method in general would be difficult to extend to other types of image data besides MRI brain data.

In 1994, Muzzolini *et al.* proposed a 3D Multi-resolution Texture Segmentation

algorithm (MTS). They stated the importance of incorporating information in the third dimension into the segmentation process when the data used is inherently 3D and proceeded to extend a 2D multi-dimensional texture segmentation algorithm to 3D. Their MTS method uses the texture present in the image to determine homogeneous regions. In addition, their technique utilized an octree structure to represent the 3D data, creating a framework for combining a split and merge process with simulated annealing. The splitting and merging of homogeneous regions was performed by characterizing $N \times N \times N$ blocks of voxels with an appropriate set of features. This provided satisfactory qualitative segmentation results on synthetic images of geometrical objects as well as a $256 \times 256 \times 128$ ultrasound image of a human fetus.

2.2 The Marching Cubes Algorithm

In 1987, Lorensen and Cline presented a new algorithm called marching cubes that creates triangle models of constant density surfaces from 3D medical data. The algorithm processes the 3D medical data in scan-line order, having abstracted the data into logical cubes created from eight pixels, four each from two adjacent slices as shown in figure 2.1. The algorithm decides whether a surface is located in a cube and then “marches on” to the next cube.

To find the surface intersection in a cube, we look at each edge of the cube and compare vertex values with a predetermined value, or threshold, of the surface under construction. If a vertex value exceeds or equals the threshold value, the vertex is thought to be inside or on the surface, and it is assigned a value of one. If the vertex value is below

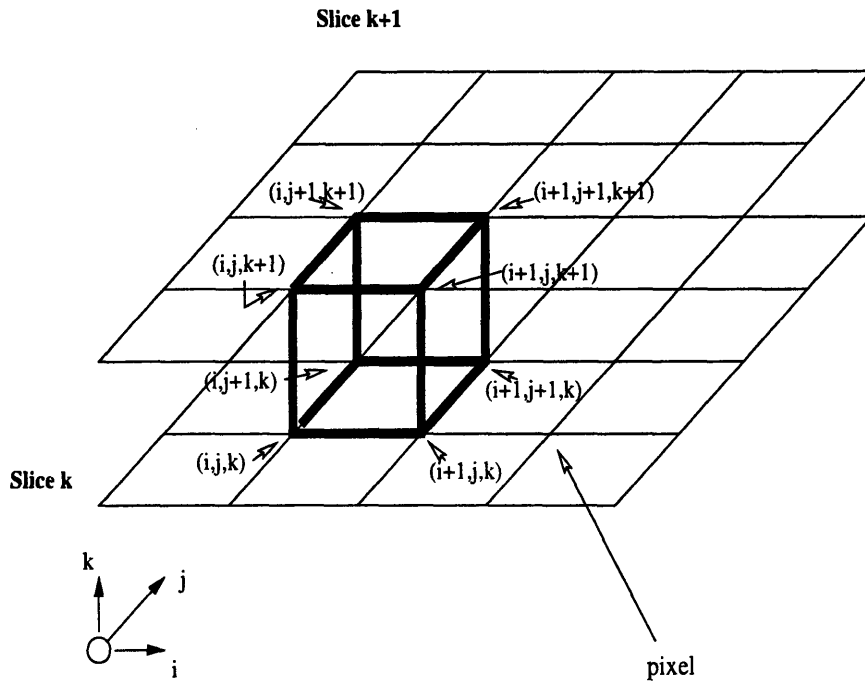


Figure 2.1: The marching cube [Lorensen and Cline, 1987]

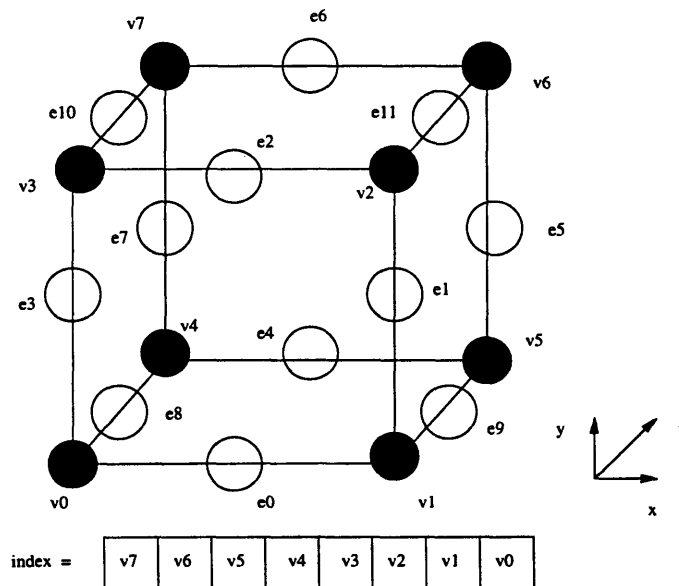


Figure 2.2: Cube numbering and the 8-bit index [Lorensen and Cline, 1987]

the threshold, the vertex is thought to be outside the surface and is assigned a value of zero. Having done this, the topology of the surface in the cube can be determined by considering the possible combinations of different values the eight vertices may take on and by approximating surface-cube intersections with triangular polygons. Lorensen and Cline observed that there were only 256 ways a surface could intersect the cube since each cube has eight vertices and each vertex has two states. They created a table containing a list of the edges intersected for each of these 256 cases, using the vertex values of the cube as an 8-bit index into this edge table (see figure 2.2).

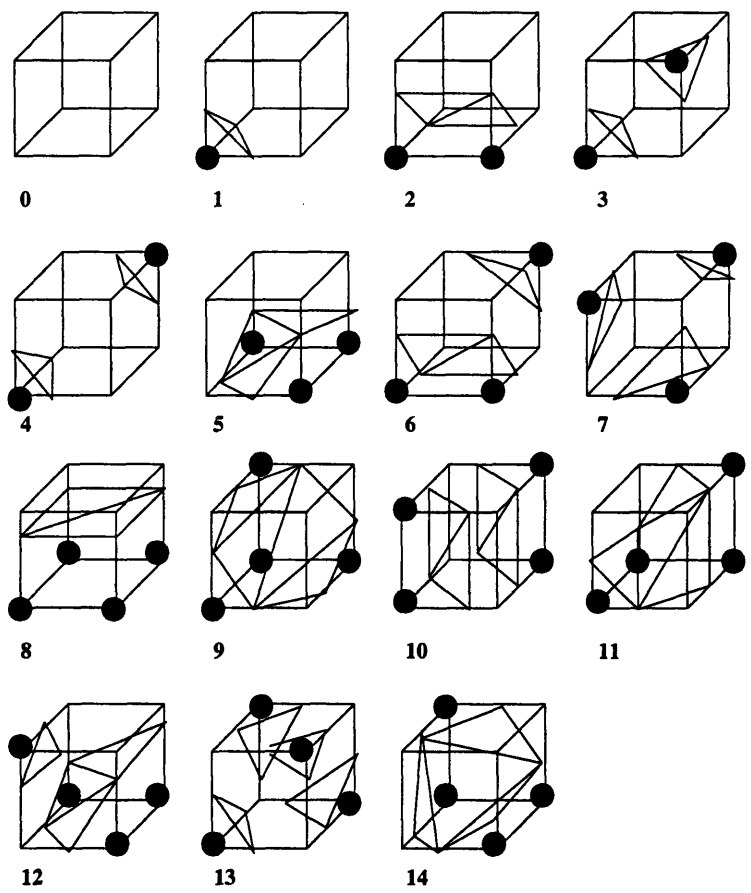


Figure 2.3: Triangulated cube cases [Lorensen and Cline, 1987]

Using complementary and rotational symmetry, it was observed that the 256 cases could be reduced to 15 cases which simplified the creation of the lookup table. Figure 2.3 shows the 15 basic cases. Linear interpolation of the cube vertex values is used to determine exactly where the surface-edge intersection point (or equivalently, the triangle vertex) is located along a cube edge. The final step in the algorithm calculates a unit normal for each triangle vertex for shading used in rendering algorithms.

2.3 Texture Segmentation

Texture is an important perceptual property, and classification and segmentation of textured images has been a topic of research for quite some time. Yet it is interesting that there is no single, unambiguous definition of what the term texture means. Many sources give different definitions of texture. Carstensen (1992) defines texture as a region in 2D or 3D that can be perceived as being spatially homogeneous in some sense. Hu and Dennis (1994) describe texture informally as an image feature that has structure composed of a large number of more or less ordered elements or patterns, without one of these drawing special attention. Gonzalez and Wintz (1987) define texture as a descriptor providing a measure of properties such as smoothness, coarseness, and regularity. The IEEE Standard Glossary of Image Processing and Pattern Recognition Terminology (1990) defines texture as an attribute representing the spatial arrangement of the gray levels of the pixels in a region. This IEEE definition seems to capture the idea best. That is, we think of texture as referring to the spatial distribution of gray level variations in an image.

There have been several different methods used for texture analysis which include

statistical methods, structural methods, spectral methods, model based methods, and multi-scale methods [Reed and Hans Du Buf, 1993]. All these methods allow for the computation of features that are utilized to differentiate and classify textures. These features can be computed by combining pixel or voxel intensities in many different ways, thereby generating a set of numbers that can hopefully represent textures in a meaningful way.

For example, residual features form one type of statistical measure. Residuals are computed by taking a pixel or voxel gray level value of interest and subtracting some fraction of the gray level values of its neighbors (located in specified directions) from its current value. The result is assigned as the new value of the pixel or voxel of interest [Muzzolini, *et al.*, 1994; Geman, *et al.*, 1990]. The average deviation of a region or volume of residuals, or what we will call a residual feature, may then be computed and used as a texture feature. Other more standard statistical approaches include the computation of simple first-order gray level statistics such as mean, variance, skewness, and energy; second-order statistical features from gray level co-occurrence matrices; and higher-order statistical features from gray level run-length matrices. Haralick (1979), Weszka *et al.* (1976), and Carstensen (1992) describe such statistical methods in detail.

Structural methods describe textures by a subpattern or primitive and the placement of these primitives (texture elements) in the image. The structural approach is useful in describing deterministic or regular textures. Spectral methods involve computations of features from a texture's Fourier power spectrum, thereby utilizing the frequency domain representation of an image. Model based methods include fitting simultaneous autoregressive models, Markov random field models, and fractal models to the textured image and

use the model parameters as features [Mao and Jain, 1992]. Multi-scale methods have currently been of great interest and have produced encouraging results, especially in the application of wavelet based multi-scale features in the classification and segmentation of textured images [Unser, 1995; Porter and Canagarajah, 1996; Chang and Kuo, 1993].

Among these various methods and techniques used for analyzing texture for segmentation and classification, in our approach to 3D segmentation of volume data we have chosen to apply first-order statistical measures (loosely grouping mean, variance, gradient, and residual feature calculations into this category), co-occurrence matrix based features, 3D Fourier power spectrum based features, and Gaussian Markov random field (GMRF) model based features. Descriptions of the details of such features and their implementations are given in subsequent chapters.

2.4 Neighborhood Definitions and Framework for Feature Computations

As described in section 2.2, the marching cubes framework allows us to process the volume data in scan-line order and lets us think of the data set as abstracted into logical cubes created from eight pixels, four each from two adjacent slices as shown in figure 2.1. In this section, we define neighboring $N \times N \times N$ local volumes of voxels which are associated with each vertex of the cube and which are used to compute local texture features.

Remember that throughout this thesis the term voxel refers to discrete scalar values in the 3D lattice of points that make up the volume data and does not refer to its other often used meaning of being a volume element in a 3-dimensional space (which results from partitioning the space by 3 sets of mutually orthogonal planes) [Udupa, *et al.*, 1982].

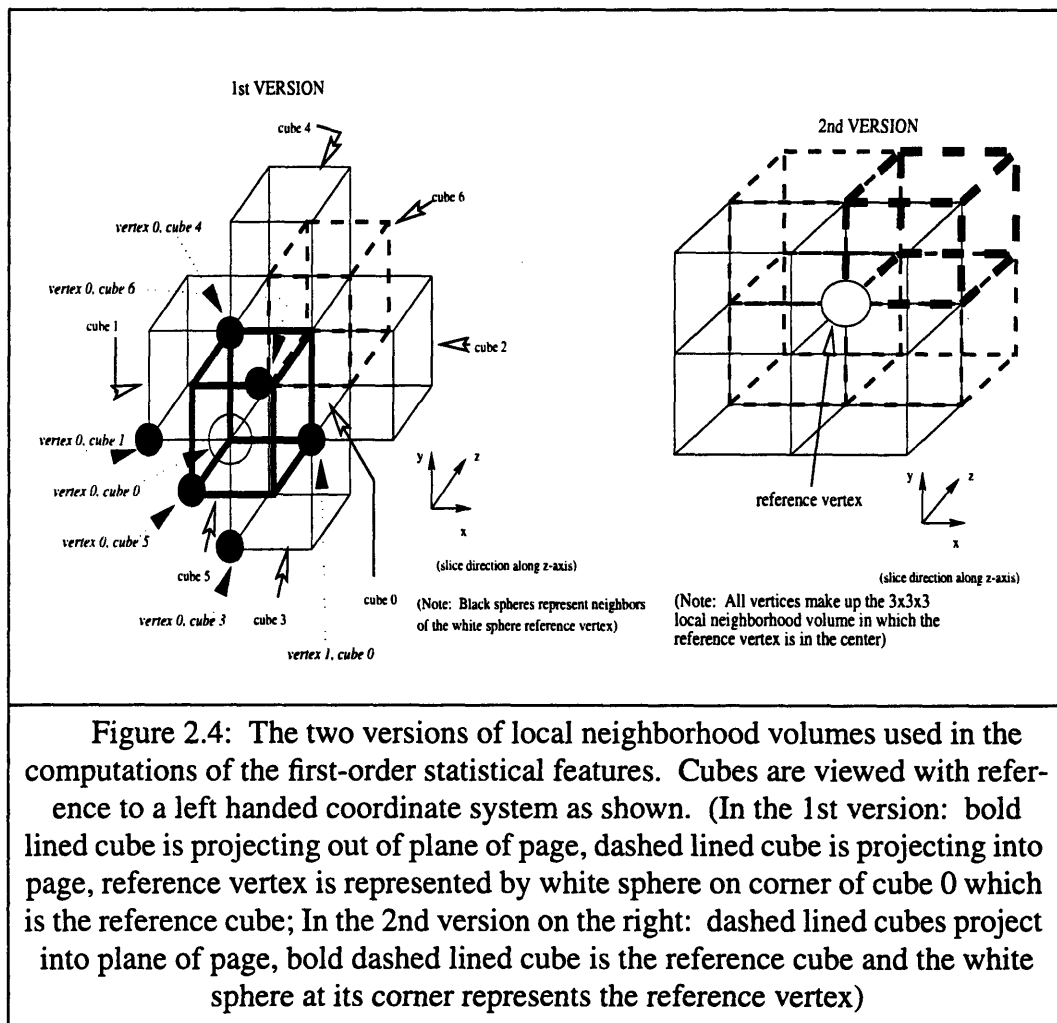
In other words, this alternative definition of a voxel is equivalent to the logical cube created from eight pixels, four each from two adjacent slices as mentioned above.

Since texture is a local picture element interaction phenomenon, it can be characterized within a local area or volume [Ehricke, 1990; Muzzolini, *et al.*, 1994; Aach and Dawid, 1990]. In 1994, Hu and Dennis presented a way to achieve unsupervised image segmentation based on clustering by calculating textural features, which were the parameters of an autoregressive model, in small non-overlapping blocks of the images used. In a statistical framework, Geman *et al.* (1990) compared blocks of pixels for finding boundaries and for partitioning scenes by computing textural and statistical features within the blocks. Clearly, computing texture features within local areas or volumes is not a new idea.

2.4.1 Neighboring Local Volumes For First-Order Statistics

Two versions of neighboring local volumes associated with each cube vertex are used for the calculation of residuals, mean, variance, and gray level gradient first-order statistics (see figure 2.4). The first version of a 3D local neighborhood for a voxel may be thought of as an extension to 3D of the so-called 2D first order neighborhood often used in dealing with Markov random field models [Carstensen, 1992; Mui, 1995; Chellappa, *et al.*, 1985]. We may imagine marching along the data set where the current cube of interest is labeled cube 0 in figure 2.4. We will call the current cube of interest the reference cube. We evaluate the necessary features for each vertex by computing the statistics using the vertex value and the values of its six surrounding neighbors. Specifically, if vertex 0 in

cube 0 is the vertex of interest (or what we will call the reference vertex), the neighboring voxels are as shown in figure 2.4. The numbering system used to refer to and access the neighboring voxel values is also specified. The local calculations are made for each vertex in the cube, the results are assigned and stored for each vertex, and then the procedure is repeated for the next cube.

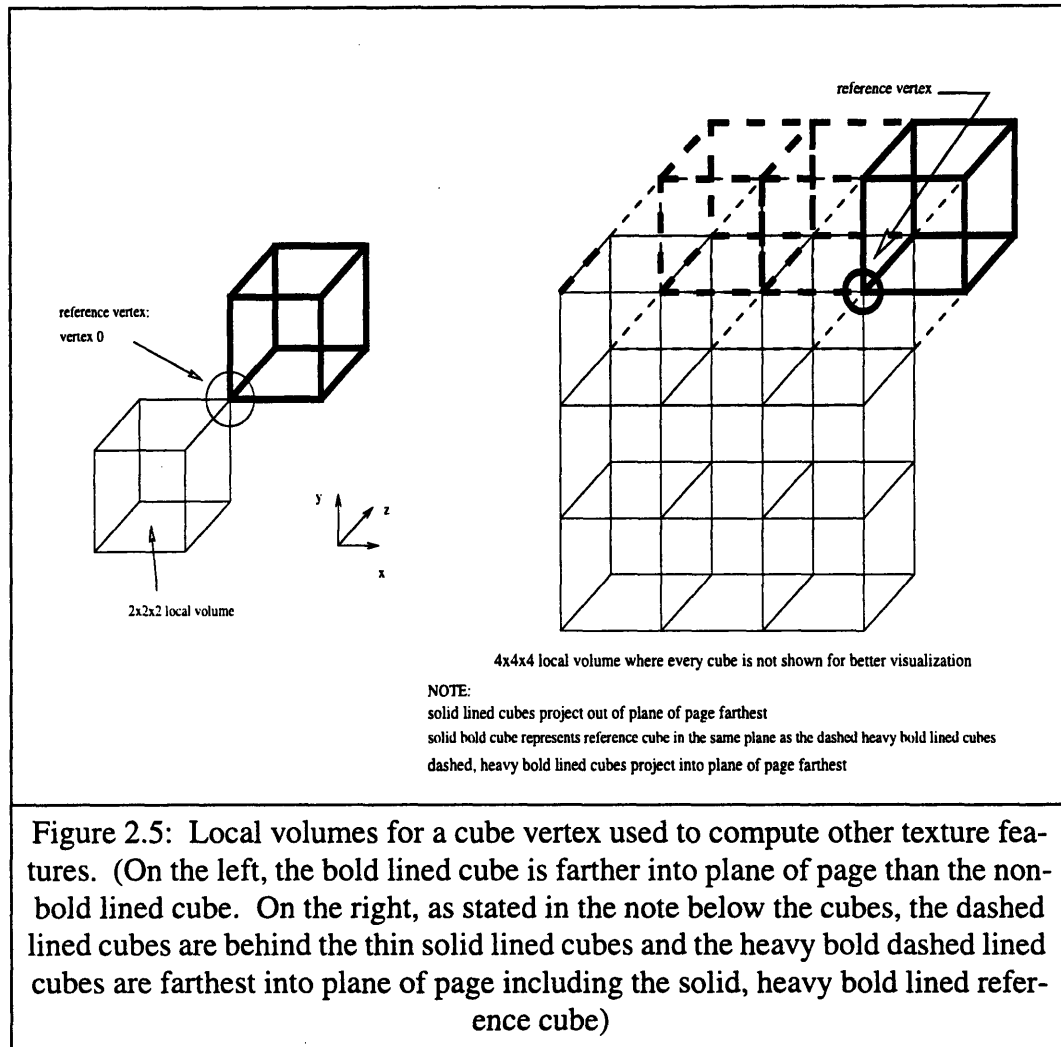


The second version of a neighboring local volume is shown in the right of figure 2.4. Here, we use a 3x3x3 volume of voxel values to compute features associated with

each cube vertex. In the figure, the vertex of interest is labeled with a white sphere, but its neighbors are not labeled with black spheres as in the 1st version figure to eliminate clutter. The vertex value of interest, or the reference vertex, is located in the center of the $3 \times 3 \times 3$ volume. Thus, although the local volumes associated with the other vertices are not explicitly shown, we can easily imagine the other cases by picturing $3 \times 3 \times 3$ volumes surrounding each reference vertex of the reference cube.

2.4.2 Neighboring Local Volumes for Other Texture Features

For the computation of 3D co-occurrence matrix based features, 3D Fourier power spectrum based features, and Gaussian Markov random field (GMRF) model based features in our approach to 3D segmentation of volume data, we use local volumes associated with each vertex of a cube that are basically extensions of the $3 \times 3 \times 3$ local volumes (shown in the right-hand side of figure 2.4) to the larger $4 \times 4 \times 4$ and $6 \times 6 \times 6$ volumes. It should be noted that we also implement a $2 \times 2 \times 2$ local volume but in a different manner as seen in figure 2.5. These $2 \times 2 \times 2$ volumes can only be used for 3D Fourier power spectrum based features due to the nature of the computations involved for both co-occurrence matrix features and GMRF features. Figure 2.5 shows a specific case of $4 \times 4 \times 4$ local volume for the shown reference vertex. Just as for the $3 \times 3 \times 3$ volume described in section 2.4.1, all the voxels in this $4 \times 4 \times 4$ volume are considered neighbors of the reference vertex. $N \times N \times N$ local volumes of other vertices may be mentally pictured given this example.



2.5 Synthetic and Real Image Data Sets

Both synthetic and real images are used to test this approach to 3D segmentation. Numerous researchers have created synthetic 2D textured images. It has been shown that the Markov random field binomial model may be used to generate blurry, sharp, line-like, and blob-like textures [Cross and Jain, 1983]. In addition, other Markov random field

models and even co-occurrence matrices have been successfully used to synthesize textured images that closely resemble real images [Lohmann, 1995; Kashyap and Chellappa, 1983]. In the following subsections, we give a brief description of how we generate the synthetic test images that are used. Two different types of synthetic images are generated. The first type of texture is created by modelling the image intensities with a 3D Gaussian Markov random field. The second type is created by assuming that the image intensities can be modelled by a Gaussian distribution. In the last subsection, the real images used for testing are discussed and examples of the synthetic and real images are shown.

2.5.1 Using GMRF Model to Create Textured Images

The 2D GMRF model was extended to 3D to create textured geometric objects within a different texture. The 2D GMRF model and the extended 3D model are described in chapter 3. Here, we do not go into detail on how the images are created. Figure 2.6 shows an example of a textured 3D image of a sphere created using two different GMRF models, one model to generate the inside texture and a second model to create the outside texture. A sectional slice of the volume data is shown using the visualization software SegmentVIEW™ created by TechnoData Software, Inc. In the section shown, a circle containing a smooth, blurry type of texture can be seen to be contained within an outside texture with a bumpy or grainy appearance. By changing the parameters of the GMRF model, we may alter the corresponding texture's characteristics. The parameters of the models used to generate the image in the figure are given in the caption. Detailed explanations of these model parameters are found in chapter 3.

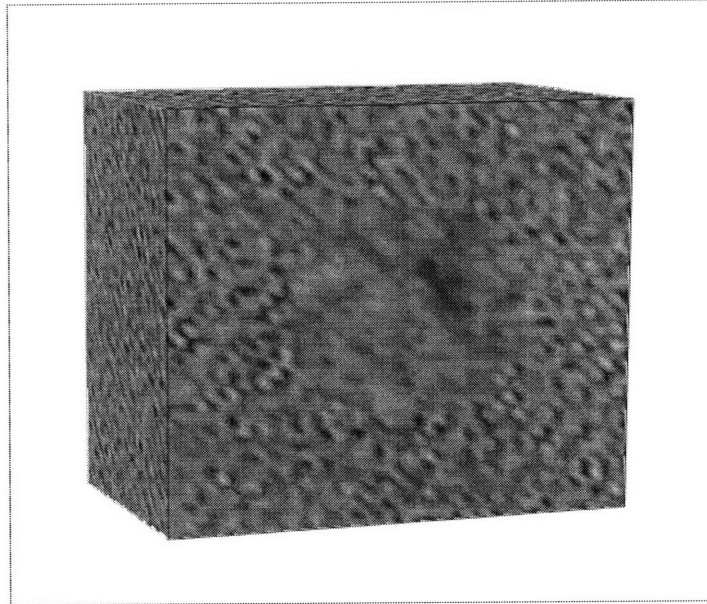


Figure 2.6: An example of a textured sphere within another texture using the 3D GMRF model. Inside texture model parameters are $\{b_0, b_1, b_2\} = \{0.2, 0.1, 0.0\}$; Outside texture model parameters are $\{b_0, b_1, b_2\} = \{-0.4, -0.4, 0.1\}$.

2.5.2 Using the Gaussian Distribution to Create Textured Images

The second form of synthetic textured images may be generated by assuming the image intensities adhere to a Gaussian distribution. In this case, the intensities may be modeled by the following equation for a Gaussian density.

$$f(y_i) = \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\left\{\frac{(y_i - \mu_k)^2}{2\sigma_k^2}\right\}} \quad (2.5.1)$$

In equation 2.5.1, μ_k and σ_k are the mean and standard deviation of volume (or region) k and y_i is the intensity value at voxel (or pixel) i .

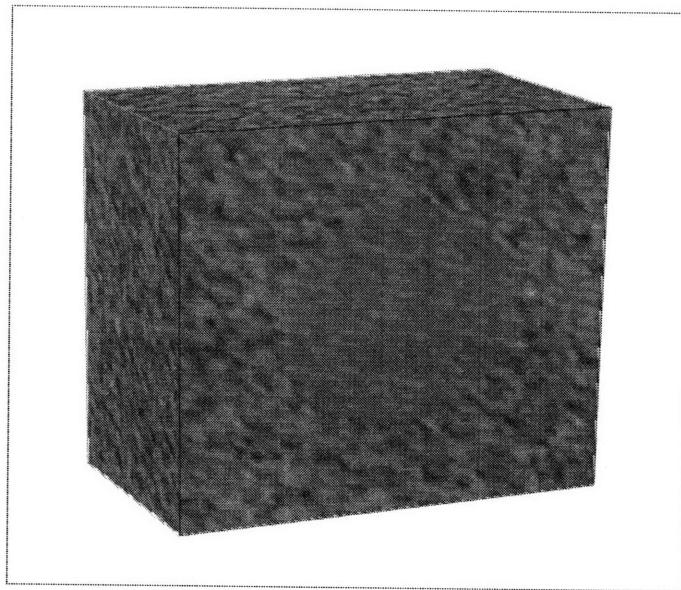


Figure 2.7: An example of a textured sphere within another texture generated using the gaussian density with mean 180.0 and variance 400.0 for the inside texture and mean 120.0 and variance 2,500.0 for the outside texture.

Figure 2.7 shows a textured sphere within another texture by using different means and variances for the inside and outside textures in the image. Other textured geometrical objects and regions besides spheres may be created. By varying the means and variances of the Gaussian intensity process, various different textures may be generated.

2.5.3 MRI Image of a Porpoise

MRI volume data of a porpoise (*Phocoena phocoena*) were provided courtesy of Dr. Darlene Ketten at the Massachusetts Eye and Ear Infirmary, Harvard Medical School. Regions of interest (ROI) may be taken from the 3D data set and tested. In figure 2.8, slice 25 of the 256x256x36 MRI data set is shown. The location shown circled by the con-

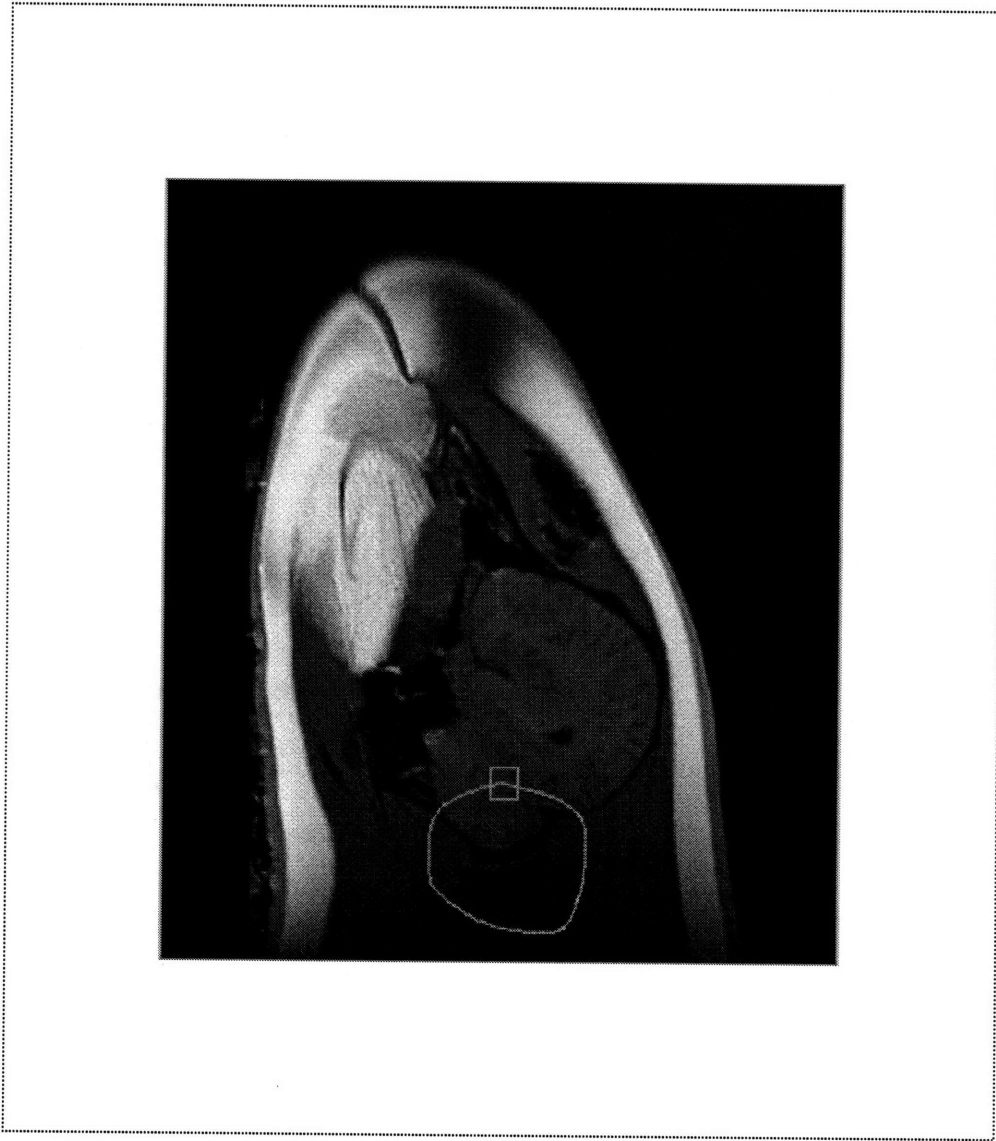


Figure 2.8: Slice of the 256x256x36 MRI data of a porpoise. The contour encircles a particular region of interest where brain tissue meets other tissues.

tour toward the bottom of the figure shows where a $64 \times 77 \times 29$ ROI was obtained for use in testing (as will be described in chapter 4). A representation of this ROI is shown in figure 2.9. This ROI was specifically selected because it contains a desirable boundary between brain, fluid, and bony tissue. A $26 \times 82 \times 22$ ROI was also selected from a portion of the $256 \times 256 \times 36$ MRI data located to the upper-right of the location indicated by the contour in figure 2.8. This ROI was selected because it contains a simpler boundary between less complex regions of texture than what may be found in the $64 \times 77 \times 29$ ROI. More discussion and details are given in chapter 4.



Figure 2.9: A $64 \times 77 \times 29$ ROI from the original $256 \times 256 \times 36$ MRI data of a porpoise. This ROI corresponds to the location indicated by the contour in the larger image of figure 2.8.

2.6 Summary

This chapter has provided information on past research in 3D image segmentation. The marching cubes algorithm was presented in detail in section 2.2. The definition of texture and various methods and features used to segment textures were described in section 2.3. Section 2.4 defines the local volumes within which texture features were computed. The last section introduced the reader to the images used to test the segmentation algorithm of this thesis. The background presented in this chapter provides sufficient information to lead into the discussion in chapter 3. Chapter 3 describes in detail the texture features used and also the modification of the marching cubes algorithm to directly segment textured 3D images.

Texture Features and Our Approach

In this chapter, the texture features used are explained in detail as well as the approach used to segment 3D images by utilizing these textural features. Four categories of texture features were formed. In the first category, we loosely group together the first-order statistical measures and the gray level gradient and residual features. These are described in section 3.1. Section 3.2 discusses the co-occurrence matrices and their associated features. Co-occurrence matrices have long been applied to analyze textures in 2D images. Here, the notions are extended to 3D. Section 3.3 presents the 3D Fourier power spectrum features used for analysis of images in the frequency domain. Section 3.4 explains the Gaussian Markov random field (GMRF) model for texture that researchers have applied to 2D textured images. Simple methods to extend the GMRF model to 3D are presented. Finally, we combine the features and discuss the method used to utilize these features to directly segment volume data within the marching cubes paradigm.

3.1 First-Order Statistical Methods

First-order statistics refer to measures of the gray level distribution of one pixel (or voxel). Similarly, second-order statistics refer to measures of the joint gray level distribu-

tion of pairs of pixels (or voxels) [Carstensen, 1992]. Although gradient and residual features are not technically first-order statistics, in this thesis the mean, variance, gradient, and residual features are regarded as first-order statistical methods for simplicity.

As explained in chapter 2, there are two versions of local volumes that are used to calculate these statistics. The local volumes used to compute these features will be discussed with the relevant features to clarify the method. For means and variances, we can describe the computations in the same manner given one of two types of local volumes.

Local Mean Intensity:

$$\mu_i = \frac{1}{N_{n_i}} \sum_{n_{ij}=1}^{N_{n_i}} y_j \quad (3.1.1)$$

Local Variance:

$$\sigma_i^2 = \frac{1}{N_{n_i}} \sum_{n_{ij}=1}^{N_{n_i}} (y_j - \mu_i)^2 \quad (3.1.2)$$

N_{n_i} is the number of voxels in the local volume, i.e., the number of neighbors of the voxel of interest, i . The symbol n_i indicates the local neighbors or local volume of voxel i . So we compute two means and two variances, one of each for the two specified local volumes giving a total of four features (see figure 2.4).

Given a gray level $f(x,y,z)$, the gradient at that Cartesian coordinate location (x,y,z) is defined as the vector:

$$G[f(x,y,z)] = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} = \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} \quad (3.1.3)$$

Gradients are computed differently depending on the type of local volume used. For the first version of local volume, voxels on either side of a voxel of interest are subtracted from each other and placed into the appropriate location of the gradient vector. In this case, neighboring voxels constitute the set of voxel pairs located one voxel away from the reference voxel, in the three directions corresponding to the x, y, and z axes of a Cartesian coordinate system. Figure 3.1 illustrates how the components of the gradient vector are computed.

Since the second version is a 3x3x3 local volume, all the voxels in the 3x3x3 volume are involved in computing the gradient vector components. We use three 3D gradient operators, or masks, to do the computation, capitalizing on the ideas of Zucker and Hummel (1981). Zucker and Hummel were the first to introduce a 3D gradient operator based on NxNxN voxel neighborhoods [Cappelletti and Rosenfeld, 1989]. They provided mathematical foundations to derive three basis functions that define their local operator. Discrete approximations to these basis functions can form three 3x3x3 masks which may be used to compute the components of the gradient vector for the voxel located at the center of the mask.

In figure 3.1, the 3x3x3 mask used to compute G_x is shown. The masks for computing G_y and G_z are not explicitly shown because they are basically a rotation in space of the same set. The analogous 2D versions of these masks are commonly referred to as

Sobel operators [Gonzalez and Wintz, 1987; Zucker and Hummel, 1981]. In actuality, the magnitude of the gradient vector is used as the feature. Given that there are two local volume versions, we now have two gradient features.

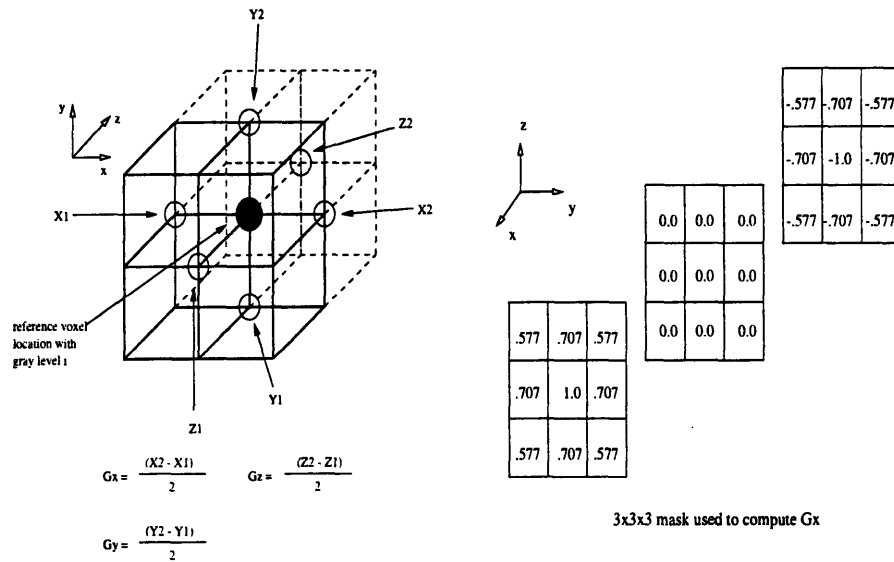


Figure 3.1: The two ways by which the components of the gradient vector are computed.

Lastly, we have been using the term residual features for what are actually average deviation measures of residuals. In other words, the gray level values of a given $N \times N \times N$ volume are transformed into residuals, and then the average deviation of the volume of residuals is taken as the texture feature. Similar features have been used by Muzzolini *et al.* (1994). Note that here we only use $N \times N \times N$ local volumes for these computations. The local residual is defined in equation 3.1.4, given that y_{ijk} is the gray level at location (i, j, k) , M represents the number of voxels in the local $3 \times 3 \times 3$ volume with (i, j, k) located in the

center, and μ_{ijk} represents the mean of the gray level values in the 3x3x3 volume.

Local Residual:

$$r_{ijk} = |y_{ijk} - \mu_{ijk}|$$

$$\text{where } \mu_{ijk} = \frac{1}{M} \sum_i \sum_j \sum_k y_{ijk} \quad (3.1.4)$$

These computed residual values are then taken and used to compute the average deviation of the volume of interest where $\overline{r_{ijk}}$ is the mean of the residuals in the volume of N^3 voxels.

$$AD(r_{ijk}) = \frac{1}{N^3} \sum_i \sum_j \sum_k |r_{ijk} - \overline{r_{ijk}}| \quad (3.1.5)$$

3.2 Gray Level Co-occurrence Matrix Features

Haralick *et al.* (1973) were the first to introduce the concept of creating gray level co-occurrence matrices from which to derive texture features from images. Gray level co-occurrence matrices (GLCM) are also called gray level spatial-dependence matrices.

These co-occurrence matrices represent second-order gray level statistics; that is, these matrices can be formulated to characterize the gray levels of pairs of pixels in an image.

In this section, the 2D GLCM and its associated features are described, followed by a discussion of the extension of these ideas to 3D.

3.2.1 2D Gray Level Co-occurrence Matrices

Haralick *et al.* defined a matrix of relative frequencies with which two pixels separated by distance d at a specified angle α occur on the image [Haralick, *et al.*, 1973; Ohanian and Dubes, 1992]. The terms and symbols used here to describe co-occurrence matrices will be borrowed from Carstensen (1992). A GLCM, \mathbf{c} , is defined with respect to a (row, column) displacement \mathbf{h} (which captures the distance d and specified angle α mentioned before). Element (i,j) of \mathbf{c} will be denoted by c_{ij} which is the number of times a point having gray level j occurs in position \mathbf{h} relative to a point having gray level i . To obtain a normalized GLCM, which will be denoted \mathbf{C} , we divide each element of \mathbf{c} by the total number of pairs in the matrix, $N_{\mathbf{h}}$. Each element of \mathbf{C} is therefore represented by $C_{ij} = c_{ij}/N_{\mathbf{h}}$. If the image has G gray levels, then the GLCM will be of size $G \times G$.

The following examples will clarify these ideas. In figure 3.2, the GLCM, \mathbf{c} , for the given displacement \mathbf{h} is shown for the given gray level image. The sample image has four gray level values and so $\mathbf{c}(\mathbf{h})$ is 4×4 . To obtain $\mathbf{C}(\mathbf{h})$, we would divide each element of $\mathbf{c}(\mathbf{h})$ by $N_{\mathbf{h}}$, which is 20 in this case.

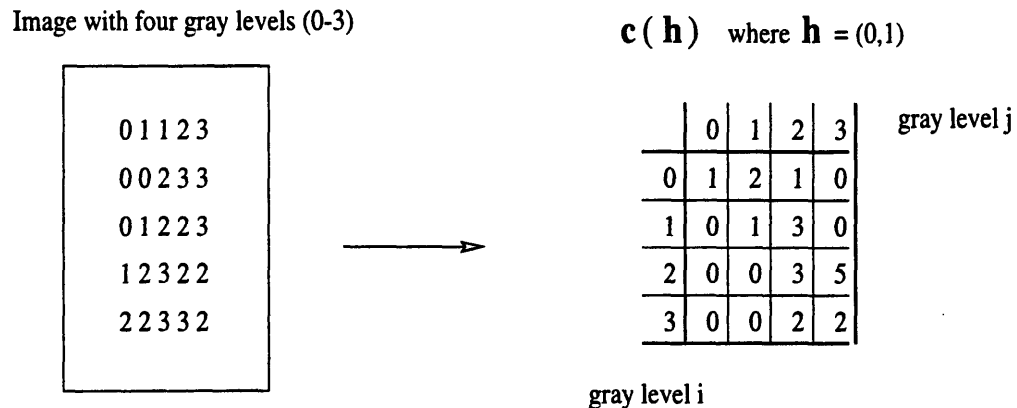


Figure 3.2: Example of computing 4×4 GLCM.

If a texture has directionality, i.e., coarser in one direction than another, then the degree of spread of values about the main diagonal in a GLCM will vary according to the angle associated with the displacement \mathbf{h} . Thus texture directionality can be analyzed by comparing co-occurrence matrices computed using displacements \mathbf{h} of the same length but of different angles.

It is convenient to use a symmetric GLCM, $\mathbf{c}_s(\mathbf{h})$, in which pairs of gray levels at separation \mathbf{h} and $-\mathbf{h}$ are counted. Pooling the frequencies, or counts, together, we have

$$\mathbf{c}_s(\mathbf{h}) = \mathbf{c}(\mathbf{h}) + \mathbf{c}(-\mathbf{h})$$

and it follows that the normalized GLCM is

$$\mathbf{C}_s(\mathbf{h}) = (1/2)[\mathbf{C}(\mathbf{h}) + \mathbf{C}(-\mathbf{h})].$$

Co-occurrence matrices provide an enormous amount of data due to the possible choices for \mathbf{h} . Carstensen (1992) suggests two ways to reduce the amount of data. One, we can pool the frequencies of more than two symmetric co-occurrence matrices of different displacements, \mathbf{h} , having approximately the same length. In this case, they are approximately the same length because a displacement of $\mathbf{h} = (1,-1)$ is associated with a location at a distance of $\sqrt{2}$ and an angle of 45° with respect to the reference location, but we take the actual $\sqrt{2}$ distance to be the same as a distance of 1. Thus one possible pooled GLCM (where \mathbf{h} is length one and includes the angles 0° , 45° , 90° , 135° and also 180° , 225° , 270° , 315° due to symmetry) is

$$\mathbf{c}(1) = \mathbf{c}_s(0,1) + \mathbf{c}_s(1,0) + \mathbf{c}_s(1,1) + \mathbf{c}_s(-1,1)$$

for which the corresponding normalized GLCM would be

$$\mathbf{C}(1) = (1/4)[\mathbf{C}_s(0,1) + \mathbf{C}_s(1,0) + \mathbf{C}_s(1,1) + \mathbf{C}_s(-1,1)]$$

The second way to reduce the data is by reducing the number of gray levels; that is, we may reduce G of the original image by binning the gray level values [Mui, 1995; Ohanian and Dubes, 1992]. The GLCM should be smaller in size than the original image to avoid having too sparse a matrix, and a reasonable value of G should be chosen such that sufficient resolution is retained to discriminate the textures in the original image being analyzed.

The GLCM is rarely used directly. Rather, features are computed from the co-occurrence matrices. Both Haralick *et al.* (1973) and Weszka *et al.* (1976) suggest the use of only four out of the many possible features for a co-occurrence matrix; consequently, only four features are used here.

The contrast feature (CON) is a natural measure of the degree of spread of the matrix values and is a measure of the local variation present in an image. It is defined as the following:

$$\text{CON} = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} (i-j)^2 C_{ij} \quad (3.2.1)$$

The angular second moment (ASM) is a measure of the homogeneity of the image. It is smallest when the C_{ij} are all as equal as possible, indicating a fine texture.

$$\text{ASM} = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} C_{ij}^2 \quad (3.2.2)$$

Entropy (ENT) is a measure of uniformity. It is largest for equal C_{ij} and smallest when the C_{ij} are unequal. Entropy is defined as follows:

$$\text{ENT} = -\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} C_{ij} \log(C_{ij}) \quad (3.2.3)$$

The correlation feature (COR) measures gray-level linear dependencies in the image. In equation 3.2.4, μ_x and σ_x are the mean and standard deviation of the row sums of matrix $\mathbf{C}(\mathbf{h})$ and μ_y and σ_y are the mean and standard deviation of the column sums. COR measures the degree to which the rows (or columns) of the GLCM resemble each other [Weszka, *et al.*, 1976].

$$\text{COR} = \frac{\sum_{i=0}^{G-1} \sum_{j=0}^{G-1} ij C_{ij} - \mu_x \mu_y}{\sigma_x \sigma_y} \quad (3.2.4)$$

3.2.2 3D Gray Level Co-occurrence Matrices

There are two versions of how we take into account the third dimension in utilizing GLCM notions. The first version is where we take the concepts of the 2D GLCM and directly extend them to 3D to create a 3D GLCM. Although we use the term 3D GLCM, the 3D GLCM is still a 2D matrix. The basic ideas in the computation of the 2D GLCM are the same, but now the search for pixel pairs includes the third dimension. In other words, now each element C_{ij} of the GLCM, $\mathbf{C}(\mathbf{h})$, is the normalized number of times a point having gray level j occurs in position \mathbf{h} relative to a point having gray level i where \mathbf{h} is a displacement in the x , y , and z directions (if the space of gray level values is viewed in a Cartesian coordinate system with x , y , and z axes).

In section 3.2.1, the pooled GLCM $\mathbf{C}(1)$ was discussed in which, given a gray level i at a specific location, gray levels of value j were searched for at displacements \mathbf{h} of

length equivalents of one and angles in the horizontal, vertical, and diagonal directions. In 3D, we compute an analogous 3D GLCM using a displacement \mathbf{h} with length one and multiple angles in 3D space as shown in figure 3.3. More specifically, figure 3.3 shows a reference voxel of gray level i (shown as a black sphere at the center of the volume) from which gray levels of value j (e.g., at the locations indicated with white spheres) are searched for in a $3 \times 3 \times 3$ volume. Note that we do not consider the locations at the four corners of the $3 \times 3 \times 3$ cubic volume since these actual $\sqrt{3}$ distances are regarded as distances of length 2. The number of counts after processing the entire volume data is placed in row i , column j of the 3D GLCM, and each component of the 3D GLCM may be divided by the total number of pairs, $N_{\mathbf{h}}$, for normalization. The CON, ASM, ENT, and COR feature computations are the same as mentioned before.

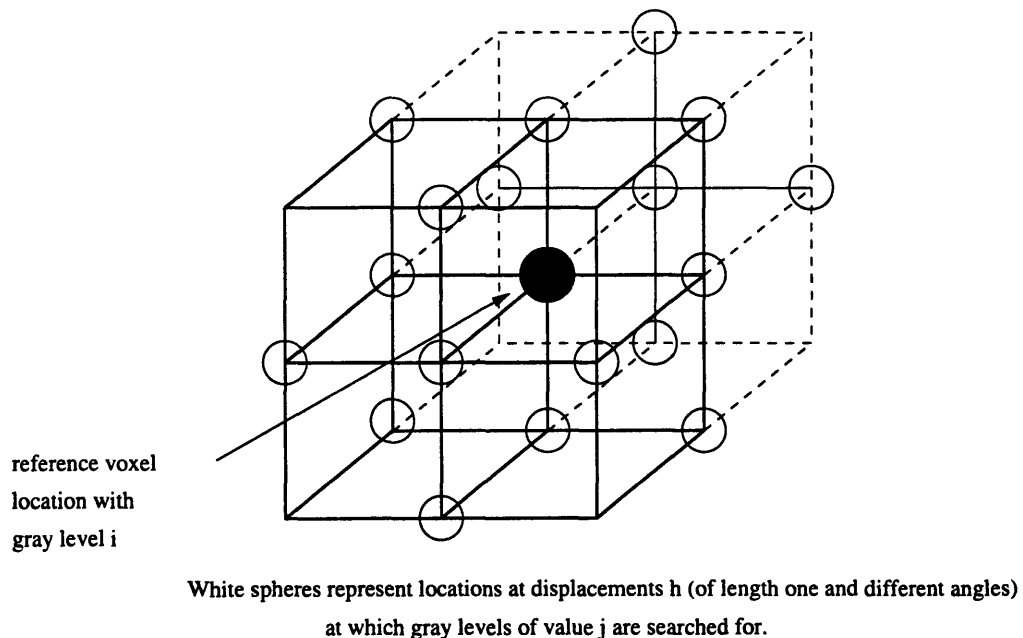


Figure 3.3: Visual description of specific displacements used for computation of direct version of 3D GLCM.

The second version entails viewing the 3D space of volume data in terms of three 2D planes (namely, the xy, zx, and yz planes). We take each 2D plane one at a time and compute the corresponding 3D GLCM for each 2D plane. Here, the term 3D GLCM refers to one matrix of a set of three gray level co-occurrence matrices, each one associated with a 2D plane (C_{xy} , C_{zx} , and C_{yz}). Figure 3.4 shows an example of computing a GLCM, $C_{xy}(1)$, that corresponds to the xy plane. We refer to this as the xy plane 3D GLCM. Figure 3.4 shows the specific example where the volume data of interest is $4 \times 4 \times 4$ in size, but there is no loss of generality. Clearly, there are actually four separate xy planes. As shown, a pooled GLCM, $C_i(1)$, is computed for each xy plane and then the elements of the four resulting $C_i(1)$ are averaged to give the elements of $C_{xy}(1)$ (the pooled GLCM $C(1)$ is described in section 3.2.1).

The same procedure may be done for the two other 2D planes, namely the zx and yz planes. The overall result is the set of 3D GLCM, i.e., a set of three pooled co-occurrence matrices $C_{xy}(1)$, $C_{zx}(1)$, and $C_{yz}(1)$. We compute four co-occurrence texture features (CON, ASM, ENT, and COR) from each matrix, giving us a total of twelve features. It should be noted that since we only compute pooled GLCM $C_i(1)$ (like the one in figure 3.3) for each xy plane in the set of parallel 2D planes, we do not take into account texture directionality as described in section 3.2.1, but the number of texture features that must be dealt with are reduced.

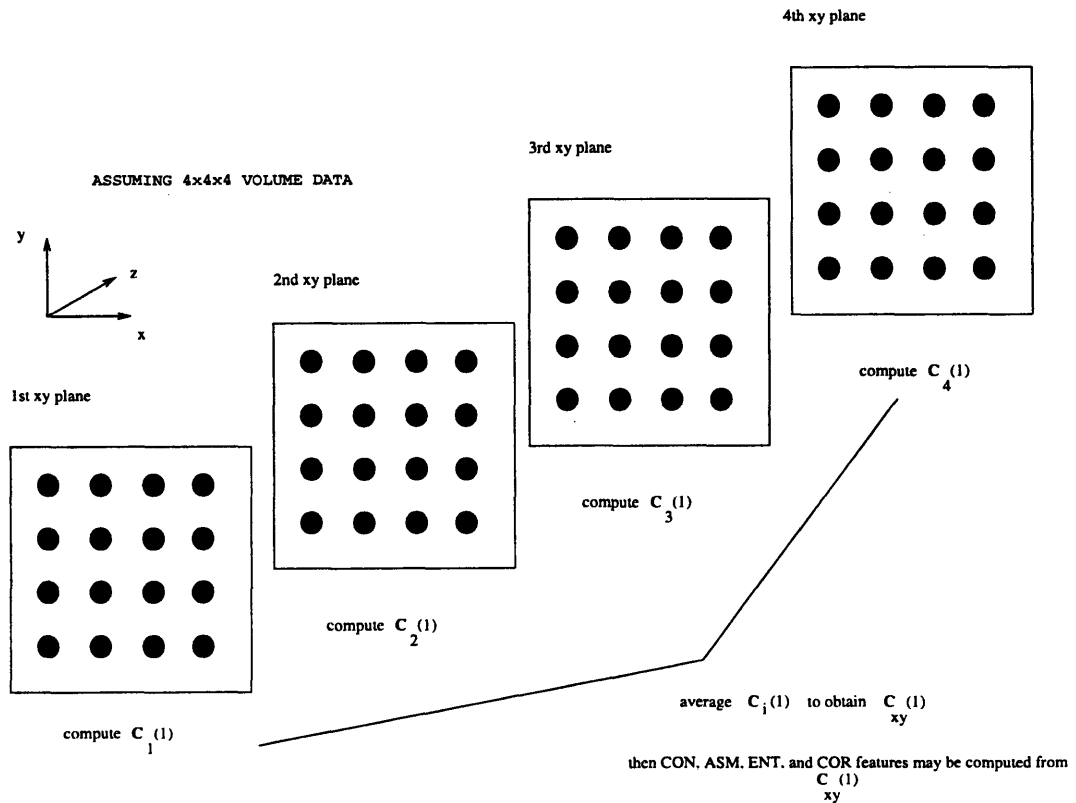


Figure 3.4: Computing $C_{xy}(1)$, the xy plane 3D GLCM, for 4x4x4 volume data. We may then compute four texture features from the resulting $C_{xy}(1)$.

3.3 Fourier Power Spectrum

Specific components in the spatial frequency domain representation of an image contain explicit information about the spatial distribution of intensity variations [D' Astous and Jernigan, 1984]. The Fourier transform has commonly been used as the tool to obtain the frequency domain representation of an image. The 2D discrete Fourier transform on a $N_1 \times N_2$ sized 2D image is defined by

$$F(u, v) = \sum_{k_2=0}^{N_2-1} \sum_{k_1=0}^{N_1-1} e^{j\left(\frac{2\pi k_2 n_2}{N_2}\right)} e^{j\left(\frac{2\pi k_1 n_1}{N_1}\right)} \quad (3.3.1)$$

The 3D discrete Fourier transform which may be used on $N_1 \times N_2 \times N_3$ 3D images is defined analogously by

$$F(u, v, w) = \sum_{k_3=0}^{N_3-1} \sum_{k_2=0}^{N_2-1} \sum_{k_1=0}^{N_1-1} e^{j\left(\frac{2\pi k_3 n_3}{N_3}\right)} e^{j\left(\frac{2\pi k_2 n_2}{N_2}\right)} e^{j\left(\frac{2\pi k_1 n_1}{N_1}\right)} \quad (3.3.2)$$

The Fourier power spectrum is the magnitude of $F(u, v)$ (or $F(u, v, w)$) squared, i.e., $|F(u, v)|^2$ (or $|F(u, v, w)|^2$). Features characterizing textures in an image may be computed from the respective power spectra.

The FFT algorithm is used to compute the 3D discrete Fourier transform, and the local volume data input into the algorithm is always $N \times N \times N$ in size, where N is a power of two. The primary interest here is in the 3D Fourier power spectrum and its associated features. As a result, the features below will be described assuming all 2D images are $N \times N$ in size and all 3D images are $N \times N \times N$ in size, N of course being a power of two. Subsection 3.3.1 presents the definitions of the 2D features and subsection 3.3.2 defines the 3D versions of these features.

3.3.1 2D Fourier Power Spectrum Features

Features from the Fourier power spectrum have long been used in the analysis of textures, but the effectiveness of such features has been debated [Weszka, *et al.*, 1976; D'Astous and Jernigan, 1984; Jernigan and D'Astous, 1984]. The criticisms stem from

the limited success of using spatial frequency information in texture analysis, but most come from work that has only used limited features derived from the frequency domain [Weszka, *et al.*, 1976]. These features are basically sums of spectral energies within regions (wedges and bands) in the frequency plane. Later work showed that the frequency domain is indeed rich in useful information. Specifically, D'Astous and Jernigan (1984) extended the set of features that can be computed from the Fourier power spectrum and found encouraging results [Liu and Jernigan, 1990; Jernigan and D'Astous, 1984].

The power spectrum in a local block or region R can be normalized according to the following definition if we let $|F(u,v)|^2 = P(u,v)$:

$$p(u, v) = \frac{P(u, v)}{\sum_{u, v \neq 0} P(u, v)} \quad (3.3.3)$$

The set of $\{ p(u,v) \mid u,v \in R \}$ may be thought of as a set of probabilities that can be used for certain feature computations. We exclude the DC component in the computation of $p(u,v)$, and this is denoted “ $u,v \neq 0$ ” in the summation sign of the denominator. Several of the features used by Weszka *et al.* (1976) and Jernigan and D'Astous (1984) will be listed here with $p(u,v)$ incorporated into some of the feature definitions.

It is known that the radial distribution of values in $|F(u,v)|^2$ is sensitive to coarseness in an image. Weszka *et al.* (1976) assert that a coarse texture will have high values of $|F(u,v)|^2$ concentrated near the origin of the power spectrum while a fine texture will have values of $|F(u,v)|^2$ that are more spread out. In the equations below, r represents the radius of a circle and θ represents an angle in radians with respect to the u -axis in the uv frequency plane. Note that for the wedge power feature, the “DC value” $(u,v) = (0,0)$ is omitted since it is common to all the wedges.

Average Power:

$$AP = \sum_{\substack{0 \leq u^2 + v^2 \leq r^2 \\ 0 \leq u, v \leq N-1}} |F(u, v)|^2 \quad (3.3.4)$$

Ring Power:

$$RP_{r_1 r_2} = \sum_{\substack{r_1^2 \leq u^2 + v^2 \leq r_2^2 \\ 0 \leq u, v \leq N-1}} |F(u, v)|^2 \quad (3.3.5)$$

Wedge Power:

$$WP_{\theta_1, \theta_2} = \sum_{\substack{\theta_1 \leq \text{atan}\left(\frac{v}{u}\right) < \theta_2 \\ 0 < u, v \leq N-1}} |F(u, v)|^2 \quad (3.3.6)$$

Entropy (EPY) is a measure of the spread of spatial frequency components within regions. EPY satisfies $0 \leq EPY \leq \log K$ where K is the number of discrete frequencies (u, v) in R . When some $p(u, v)$ is 1, then EPY takes on its minimum value. So EPY takes on a maximum value when all the $p(u, v)$ are equal (or $p(u, v) = 1/K$). Specifically for the EPY feature, $p(u, v)$ is computed without excluding the “DC component” where $(u, v) = (0, 0)$.

Entropy:

$$EPY = - \sum_{u, v \in R} p(u, v) \log(p(u, v)) \quad (3.3.7)$$

D’Astous and Jernigan (1984) mention that peaks in the power spectrum correspond to the degree of regularity in texture. That is, specific characteristics of the peaks

can be used as texture features. Using u_1, v_1 to indicate the frequency coordinates of the maximum peak of the power spectrum, the important peak characteristics are then as follows:

Energy in the major peak:

$$EMPeak = p(u_1, v_1) \times 100 \quad (3.3.8)$$

Another feature is the Laplacian which takes on a small value for a flat peak and a large value for a pointed peak. In addition, fine textures have peaks farther away from the origin of the power spectrum than coarse textures. This may be measured using the squared major peak frequency.

Laplacian of major peak:

$$LAPPeak = \nabla^2 P(u_1, v_1) = \begin{aligned} &P(u_1 + 1, v_1) + P(u_1 - 1, v_1) + P(u_1, v_1 + 1) \\ &+ P(u_1, v_1 - 1) - 4P(u_1, v_1) \end{aligned} \quad (3.3.9)$$

Squared major peak frequency:

$$SQPeakFreq = u_1^2 + v_1^2 \quad (3.3.10)$$

As D'Astous and Jernigan (1984) mention, the peak features do not provide information on the shape of the spatial frequency component distributions. They list a few features that may be used to characterize the shape of frequency components of the Fourier power spectrum, noting that a highly directional texture corresponds to an elongated elliptical shape distribution and that an isotropic texture (texture without directionality) corresponds to a more circular shape.

The isotropy of the power spectrum measures the elongation of the distribution of spatial frequencies and is minimum for isotropic textures and maximum for parallel line textures.

Isotropy of the power spectrum:

$$\text{ISO} = \frac{|\sigma_u - \sigma_v|}{\sqrt{(\sigma_u + \sigma_v)^2 - 4\sigma_{uv}^2}} \quad (3.3.11)$$

where

$$\sigma_u = \sum_u \sum_v u^2 p(u, v)$$

$$\sigma_v = \sum_u \sum_v v^2 p(u, v)$$

$$\sigma_{uv} = \sum_u \sum_v uv p(u, v)$$

The circularity feature compares the area of the distribution to the area of a circle having a radius equal to the length λ of the major axis of the distribution. This length λ is the maximum eigenvalue of the covariance matrix of $p(u, v)$. The area of the circle may be labeled A_C , and this represents the number of discrete frequencies enclosed within a circle of radius $\sqrt{\lambda}$. The area of the distribution, A_D , is the number of non-zero frequency components within a circle of radius $\sqrt{\lambda}$.

Circularity of the power spectrum:

$$\text{CIR} = \frac{A_D}{A_C} \quad (3.3.12)$$

3.3.2 3D Fourier Power Spectrum Features

Due to the nature of the computation of the 3D Fourier power spectrum, the 3D Fourier power spectrum features are straightforward extensions of these 2D features.

Regions now mean volumes, peaks of the power spectrum now are maximum values of the

power spectrum, and circles become spheres. Although all of the previously mentioned features have some 3D version, we list explicitly only some of their extensions. In the equations below, r represents the radius of a sphere, and $p(u,v,w)$ represents the 3D version of $p(u,v)$ described earlier:

$$p(u, v, w) = \frac{P(u, v, w)}{\sum_{u, v, w \neq 0} P(u, v, w)} \quad (3.3.13)$$

A local volume of size $N \times N \times N$ is represented by R . The coordinates u_1, v_1, w_1 represent the location of the maximum value of the 3D power spectrum. Also, $p(u,v,w)$ in equation 3.3.15 includes the “DC component” value located at $(u,v,w) = (0,0,0)$.

3D Average power:

$$AP = \sum_{\substack{0 \leq u^2 + v^2 + w^2 \leq r^2 \\ 0 \leq u, v, w \leq N-1}} |F(u, v, w)|^2 \quad (3.3.14)$$

3D Entropy:

$$EPY = - \sum_{u, v, w \in R} p(u, v, w) \log(p(u, v, w)) \quad (3.3.15)$$

3D Maximum energy:

$$ME = p(u_1, v_1, w_1) \times 100 \quad (3.3.16)$$

Laplacian of the maximum energy:

$$\begin{aligned} \text{LAPME} = \nabla^2 P(u_1, v_1, w_1) = & P(u_1 + 1, v_1, w_1) + P(u_1 - 1, v_1, w_1) \\ & + P(u_1, v_1 + 1, w_1) + P(u_1, v_1 - 1, w_1) \\ & + P(u_1, v_1, w_1 + 1) + P(u_1, v_1, w_1 - 1) \\ & - 6P(u_1, v_1, w_1) \end{aligned} \quad (3.3.17)$$

Squared maximum energy frequency:

$$\text{SQMEfreq} = u_1^2 + v_1^2 + w_1^2 \quad (3.3.18)$$

3.4 Gaussian Markov Random Fields

Markov random field (MRF) models have been used by many researchers for image analysis, classification, and segmentation [Carstensen, 1992; Mui, 1995; Geman and Geman, 1984; Hassner and Sklansky, 1980]. There are several Markov random field models, but only the Gaussian Markov random field (GMRF) model is specifically used in this thesis. It should be noted also that the GMRF model has been used by researchers to synthesize textures [Chellappa and Chatterjee, 1985; Mui, 1995].

3.4.1 2D GMRF Model

The conditional density for the GMRF model is given by the following equation [Mui, 1995; Carstensen, 1992; Besag, 1974]:

$$f(y_i | y_j, j \in N) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \left(y_i - \mu - \sum_{j \in N_i} b_j (y_j - \mu)\right)^2\right) \quad (3.4.1)$$

in which y_i represents the gray level of pixel i , and μ and σ represent the local gray level mean and variance. N represents all of the local neighbors of pixel i .

The structure of the neighborhood system determines the order of the MRF. A first order MRF means that the neighborhood of a pixel includes four of its nearest neighbors. For a second order MRF, the neighborhood includes eight of a pixel's nearest neighbors. The neighborhood structure of MRF models is shown in figure 3.5. An often used method

of indicating the set of neighbors is to denote the neighbor set as $N = \{(0,1),(0,-1),(-1,0),(1,0)\}$ for the first order MRF; and $N = \{(0,1),(0,-1),(1,0),(-1,0),(1,-1),(-1,-1),(1,1),(-1,1)\}$ for a second order MRF; etc. These pairs of numbers are to be added to the (i,j) Cartesian coordinate location of the pixel of interest to find the locations of the neighbors.

5	4	3	4	5
4	2	1	2	4
3	1	y_i	1	3
4	2	1	2	4
5	4	3	4	5

Figure 3.5: Order coding of neighborhood structure. The n th order neighborhood of the center pixel contains pixels with numbers less than or equal to n [Carstensen, 1992].

The GMRF model may be equivalently characterized by a linear interpolative equation containing a set of unknown parameters [Woods, 1972; Besag, 1974]. Let $\Omega = \{s = (i,j), 0 \leq i,j \leq M-1\}$ denote the set of grid points in the $M \times M$ image. We may then partition Ω into Ω_I , the interior set, and Ω_B , the boundary set, where $\Omega_B = \{s = (i,j): s \in \Omega \text{ and } (s+r) \notin \Omega \text{ for at least one } r \in N\}$ and $\Omega_I = \Omega - \Omega_B$ (where N is defined below). Furthermore, let $y(s)$ denote the zero mean observations from a given texture where $\{y(s), s \in \Omega_I\}$ are also Gaussian. The difference equation is then given by equation 3.4.2 where N is a symmetric neighbor set (where we let $N = \{s:s \in N_s\} \cup \{-s:s \in N_s\}$ and where symmetry

means $b_r = b_{-r}$) [Chellappa and Chatterjee, 1985; Kashyap and Chellappa, 1983]:

$$y(s) = \sum_{r \in N_s} b_r (y(s+r) + y(s-r)) + e(s) \quad (3.4.2)$$

where $e(s)$ is a zero mean stationary Gaussian noise sequence with the following properties:

$$\begin{aligned} E(e(s)e(r)) &= -b_{s-r}\sigma^2, \quad ((s-r) \in N) \\ &= \sigma^2, \quad s=r \\ &= 0, \quad \text{otherwise} \end{aligned} \quad (3.4.3)$$

From equations 3.4.2 and 3.4.3, it can be shown that

$$\begin{aligned} E(e(s)y(r)) &= 0 \quad (r \neq s) \\ &= \sigma^2 \quad (r = s) \end{aligned} \quad (3.4.4)$$

In turn, equation 3.4.4 implies that

$$E(e(s) | \text{all } y(r), r \neq s) = 0 \quad (3.4.5)$$

which further implies

$$p(y(s) | \text{all } y(r), r \neq s) = p(y(s) | \text{all } y(s+r), r \in N) \quad (3.4.6)$$

Thus we obtain equation 3.4.6, which is the conditional density that characterizes the GMRF model.

From this GMRF model representation, Chellappa and Chatterjee (1985) estimated the unknown parameters $\mathbf{b} = (b_r, r \in N_s)$ and σ^2 by using the least squares method. Asterisks signify values that are estimates and $\mathbf{q}(s)$ is the column vector $[y(s+r) + y(s-r), r \in N_s]$

where:

$$\mathbf{b}^* = \left[\sum_{\Omega_i} \mathbf{q}(s)(\mathbf{q}(s))^T \right]^{-1} \left[\sum_{\Omega_i} \mathbf{q}(s)y(s) \right] \quad (3.4.7)$$

and

$$\sigma^{2*} = \frac{1}{M^2} \sum_{\Omega_i} [y(s) - \mathbf{b}^{*T} \mathbf{q}(s)]^2 \quad (3.4.8)$$

For the specific case of the second order GMRF, there are only four distinct parameter values (for ease of notation, the symbols N, S, E, W represent the north, south, east, west directions at which the neighbors are located with respect to the pixel i of interest; see figure 3.6). The parameters \mathbf{b}^* and variance σ^{2*} of the model may be used as features to distinguish between different textures.

$$\mathbf{b}^* = \begin{bmatrix} b_{1,-1}^* \\ b_{1,0}^* \\ b_{1,1}^* \\ b_{0,1}^* \end{bmatrix} = \begin{bmatrix} b_0^* \\ b_1^* \\ b_2^* \\ b_3^* \end{bmatrix} \quad \mathbf{q}(s) = \begin{bmatrix} y_N + y_S \\ y_{SW} + y_{NE} \\ y_{SE} + y_{NW} \\ y_W + y_E \end{bmatrix} \quad (3.4.9)$$

These closed form solutions may be obtained by using the pseudo-likelihood function (PL) [Carstensen, 1992]:

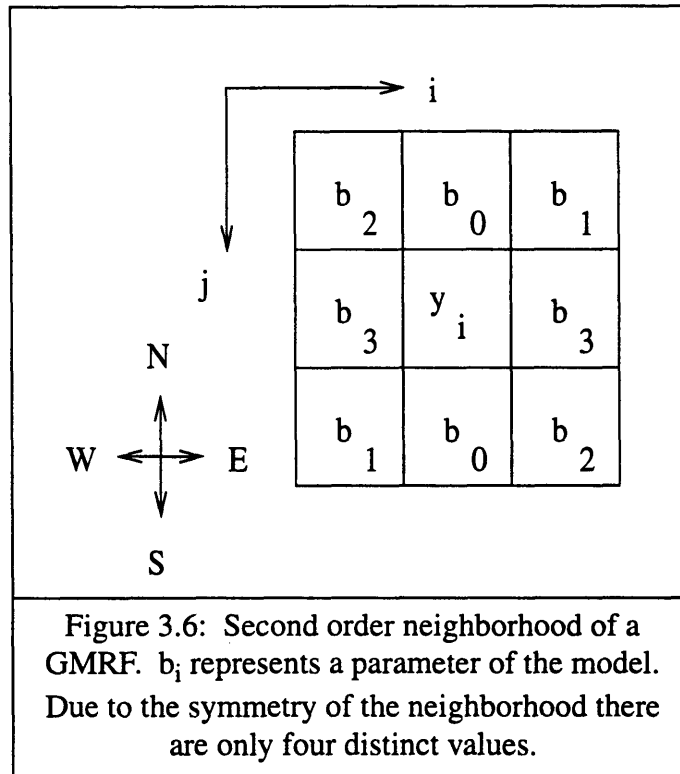
$$PL = \prod_i \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left\{ -\frac{1}{2\sigma^2} \left(y_i - \mu - \sum_{j \in N} b_j (y_j - \mu) \right)^2 \right\} \quad (3.4.10)$$

By setting the partial derivatives of the log-likelihood equal to zero, we obtain the closed form solutions of equations 3.4.7 and 3.4.8.

3.4.2 3D GMRF Model

There are two methods by which the 2D GMRF model is extended to 3D. The first method may be thought of as a direct extension to 3D. In forming a direct 3D extension of

the 2D model, we attempt to maintain the validity of the equations presented in section 3.4.1. The only change needed is to extend the neighborhood region around a pixel of interest to a neighborhood volume around a voxel of interest. In other words, we now let $\Omega = \{s = (i,j,k), 0 \leq i,j,k \leq M-1\}$ denote the set of grid points in the $M \times M \times M$ image. We again partition Ω into Ω_I , the interior set, and Ω_B , the boundary set, where $\Omega_B = \{s = (i,j): s \in \Omega \text{ and } (s+r) \notin \Omega \text{ for at least one } r \in N\}$ and $\Omega_I = \Omega - \Omega_B$ (N represents once again the symmetric neighborhood). All the equations of the previous section should then be maintained given this modification and given that an appropriate 3D neighborhood system is defined.



We propose a symmetric neighborhood N in 3D. Since only a second order GMRF model is dealt with in this thesis, all discussion will center around the second order model.

Extensions to higher order models should follow straightforwardly from the second order case. Having said that, figure 3.7 shows the symmetric neighbor set and the corresponding GMRF model parameter values for the second order GMRF. For a symmetric neighborhood in 3D, it was found that there are only three distinct model parameters.

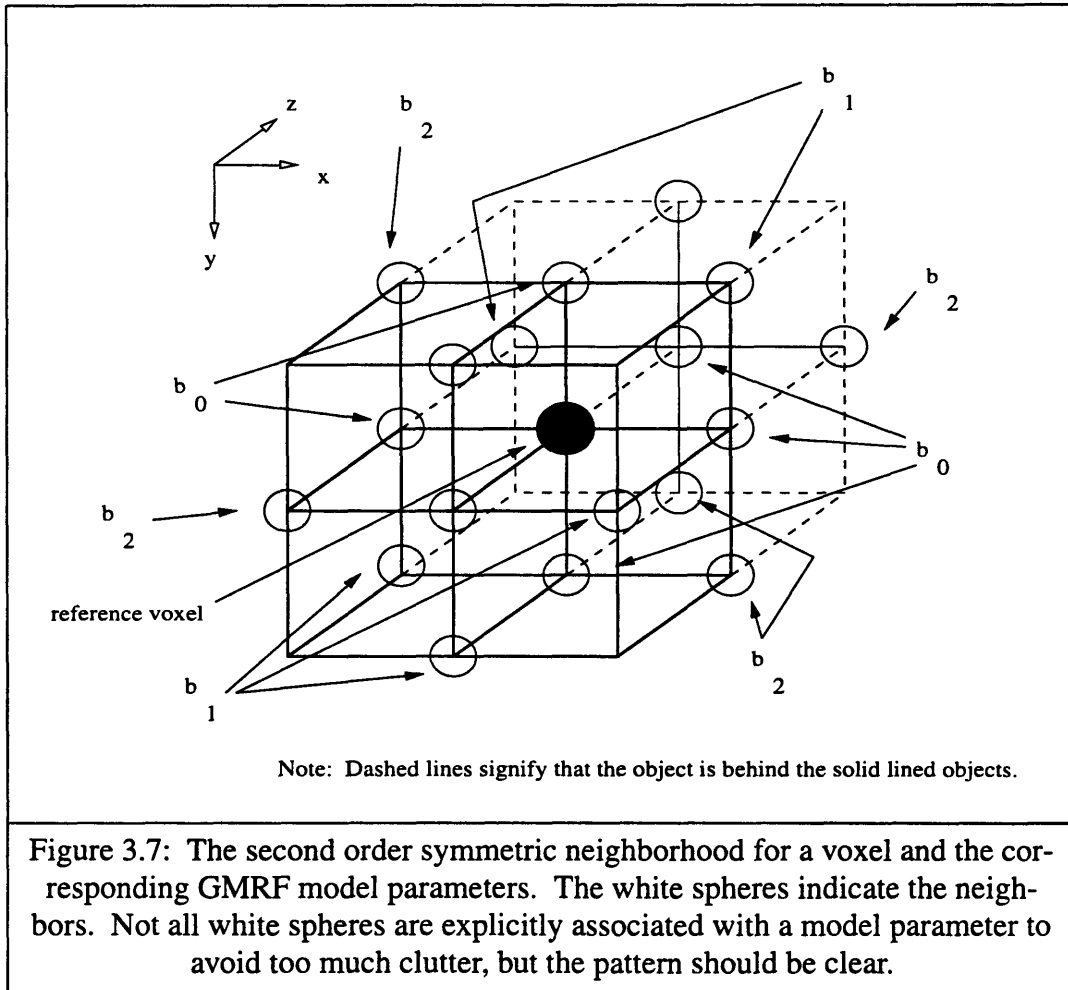
Equations 3.4.7 and 3.4.8 are still valid estimators of the model parameters and variance where for the specific case of a second order 3D GMRF we have the following modified version of equation 3.4.9:

$$\mathbf{b}^* = \begin{bmatrix} b_0^* \\ b_1^* \\ b_2^* \end{bmatrix} \quad \mathbf{q}(s) = \begin{bmatrix} y_N^{xy} + y_S^{xy} + y_W^{xy} + y_E^{xy} + y_{OP} + y_{IP} \\ \sum_{\text{for } i = xy, zx, yz \text{ plane}} (y_{SW}^i + y_{NE}^i) \\ \sum_{\text{for } i = xy, zx, yz \text{ plane}} (y_{SE}^i + y_{NW}^i) \end{bmatrix} \quad (3.4.11)$$

In the 3D second order neighborhood (as seen in figure 3.7), there are a total of eighteen neighboring voxels, and so each element of the column vector $\mathbf{q}(s)$ is the sum of a total of six different voxel gray level values. The symbols N, S, E, W represent the directions north, south, east, and west with respect to the center voxel of interest as explained in section 3.4.1. The symbol OP means out of the plane of the page, and IP means into the plane of the page (see figure 3.7). In terms of the yz plane in which the y_{OP} and y_{IP} voxels lie, y_{OP} is the same thing as y_N^{yz} and y_{IP} can be equivalently written as y_S^{yz} . This first method results in an estimate of \mathbf{b}^* and σ^{2*} for a total of four features.

In the second method, the 3D space is thought of in terms of three mutually perpendicular 2D planes (the xy, zx, and yz planes). Essentially, we estimate the second order 2D GMRF model parameters for the xy, zx, and yz planes. This results in a total of three feature sets. Thinking of a specific case with no loss of generality, we notice that if

we are computing xy plane features of the GMRF model where the volume data being used is $4 \times 4 \times 4$ in size, then we actually have four xy planes. GMRF model features are computed for each of the four planes and then averaged to give the features that represent the 2D xy plane. There is no true direct extension here of the 2D GMRF model to 3D. By estimating parameters for each group of parallel 2D planes, the procedure is akin to estimating the GMRF parameters for three different 2D images except that here it must be remembered that the three sets of mutually orthogonal 2D planes are not completely independent of each other and that averaging is done for the computed GMRF model parameters among the parallel 2D planes of the local volumes.



This means that each of the three sets of features contains three model parameters and one variance estimate. Note that we say three model parameters instead of four (which is to be expected for a 2D image; see figure 3.6) because we want to maintain a 3D symmetric neighborhood which means that the N, S, E, and W neighbors of a voxel in a specific 2D plane must have the same model parameter associated with them. To make this discussion clearer, observe the xy plane in figure 3.7. Note that the neighborhood voxels with which the parameter b_0 is associated with in this plane are the N, S, E, and W neighbors of the reference voxel. By using this method, all the equations and assumptions of section 3.4.1 should be valid. This second method of extending the GMRF model to 3D results in a total of twelve features.

3.5 Our Approach Using Texture Features

In this section, some ideas presented earlier are briefly restated, followed by a description of the proposed approach to segmentation of volume data. Having described all the texture features that may be computed in our implementation in the previous sections, we discuss how to incorporate these features into this approach. In addition, the similarity measure used by which volumes of texture are compared to one another are discussed.

3.5.1 The Method Using a Feature Vector and a Similarity Measure

Segmentation often entails finding distinct regions in an image by assigning cate-

gorical labels to pixels or voxels. In this approach however, we are segmenting the data by attempting to create surfaces between distinct 3D texture regions using texture features to determine where boundaries lie. This method may be thought of as essentially an edge-based, or boundary-based, approach to segmentation.

The marching cubes algorithm constructs surfaces between constant gray level regions quite well. It will be recalled that the basic idea is that the 3D space of voxel values is partitioned into a regularly spaced lattice of points which form cubical elements when considering eight neighboring points of the lattice at a time (see section 2.2 in chapter 2). At the heart of this method is a simple threshold test that is performed to determine whether a surface intersects a particular cube edge. In any image with texture, significant regions of the image may not have constant intensity. The marching cubes method cannot successfully construct adequate surfaces between homogeneous regions of texture.

As described before, our approach comprises a marching cubes framework. We process the data in scan-line order, meaning left to right and top to bottom. As we march along the volume data, we partition the data into cubes and compute texture features for each vertex of the current cube of interest. Each vertex has associated with it a local volume from which the texture features may be computed as detailed in section 2.4 of chapter 2.

These texture features are now used to compare local volumes of texture associated with the cube vertices located on opposite ends of a cube edge. It should be noted that the local volumes described in chapter 2 are symmetrical in nature. That is, although each cube vertex has three edge neighbors on that same cube, the same local volume is used for a cube vertex regardless of which edge is currently being investigated for a sur-

face intersection. Numerous texture features may be computed and placed into an array, or a feature vector, and these feature vectors which represent the textures of their corresponding local volumes may then be compared across a cube edge to determine whether a surface-edge intersection occurs. Essentially, particular features are extracted from local volumes and the 3D image is segmented based on the disparity between the feature vectors. This idea of comparing sub-regions of an image using features computed from the respective sub-regions has been used by numerous researchers involved with image classification and segmentation [Manjunath and Chellappa, 1991; Ehrlicke, 1990; Hu and Dennis, 1994; Graffigne, 1987]. In our case, we think of each cube vertex as being assigned a vector of texture features. Each point that represents a single voxel value in the 3D lattice of points is now transformed into a vector with multiple values.

We may obtain different feature vectors by using different categories of features to place in them. That is, we have already described four categories of features, namely first-order statistical features, co-occurrence matrix based features, Fourier power spectrum features, and GMRF model features. Groups of features from each category may be selected to form a feature vector so that differences between the performance of distinct categories of features may be investigated. Experimental tests using different feature vectors are described in chapter 4.

Texture feature vectors may be compared using a normalized Euclidean distance measure. The normalized Euclidean distance measure is defined as

$$d(F^i, F^j) = \sum_k \frac{(f_k^i - f_k^j)^2}{(f_k^i)^2 + (f_k^j)^2} \quad (3.5.1)$$

where F^i and F^j represent feature vectors associated with their respective cube vertices

along an edge, and f_k^i and f_k^j represent the k th feature in their respective feature vectors [Manjunath and Chellappa, 1991]. The normalization takes into account that some feature values are inherently larger than other feature values and thus offsets the weight of these larger valued features.

A surface-edge intersection is now determined to exist if

$$d(F^i, F^j) > K$$

where K is some pre-determined constant. That is, if the normalized Euclidean distance measure for the two feature vectors along a cube edge is greater than some threshold, a surface is said to intersect that cube edge. Again, note that we are essentially comparing volumes of texture and that we use triangular polygons to represent the boundary between different textures. Unfortunately, many segmentation methods are *ad hoc* in nature due to the complexity of images and due to the absence of any underlying mathematical theories for segmentation. This method is no exception. Optimal threshold values must be found through a tedious trial and error process. The thresholds used for testing in specific cases will be given in chapter 4.

Experiments: Tests and Results

Texture features lay the foundation of our approach to segmenting volume data as discussed in chapter 3. This chapter presents qualitative experimental results in which different texture features in the segmentation algorithm were employed to see how well surfaces could be created between distinct textured volumes. The segmentation algorithm was applied to a group of synthetic images and also to 3D regions of interest from MRI data of a marine mammal. Only qualitative results are discussed since most of the computed polygonal surfaces were not closed and therefore were not amenable to quantitative assessment. Section 4.1 describes the specific synthetic and real image test sets used. Section 4.2 serves to introduce the reader to how the standard marching cubes algorithm performs by presenting some examples. Section 4.3 presents the results of testing synthetic images. Then the results of testing real volume data are given in section 4.4. In the previous chapter, we categorized the texture features into four classes. In this chapter, this categorization forms the framework for an ordered structure for the presentation of results. Sections 4.3 and 4.4 demonstrate the results of using texture features obtained from first-order statistics, 3D co-occurrence matrices, 3D Fourier power spectra, and 3D GMRF models. Section 4.5 discusses some implementation details. Finally, section 4.6 gives a summary of the results in this chapter.

4.1 Synthetic and Real Volume Data Used

Chapter 2 presented two of the synthetic textured images that were used. Both were $46 \times 46 \times 46$ images of textured spheres within a different texture. However, one 3D image was created using the 3D GMRF model, and the other was generated by assuming a Gaussian distribution of intensities. Here we complete the presentation of the suite of synthetic test data. Figure 4.1 presents a $46 \times 46 \times 46$ image of a textured sphere using the GMRF model parameters shown in the figure's caption. Figure 4.1 also shows the histogram of gray level variations in the image of the textured sphere. The single peak of the histogram demonstrates that simple gray level thresholding cannot be used to segment the two distinct 3D regions in the image. Including the $46 \times 46 \times 46$ textured sphere images shown in chapter 2, we have a total of three different $46 \times 46 \times 46$ textured sphere images.

We similarly create three $64 \times 64 \times 64$ t-shaped cube (or tcube for simplicity) textured images using the same GMRF model parameters and the same Gaussian density means and variances that were used to generate the textured sphere images. Since the textures appear the same as the corresponding textures for the sphere volume data already discussed, we only show two of the textured tcube images in figure 4.2. The models and the model parameters by which the images are created are described in the corresponding figure captions.

In addition, three larger $84 \times 84 \times 84$ textured tcube images are generated. Each of these three images are created using the same models and model parameters as each of the three $64 \times 64 \times 64$ tcube images. Basically, size is the only difference between the two sets

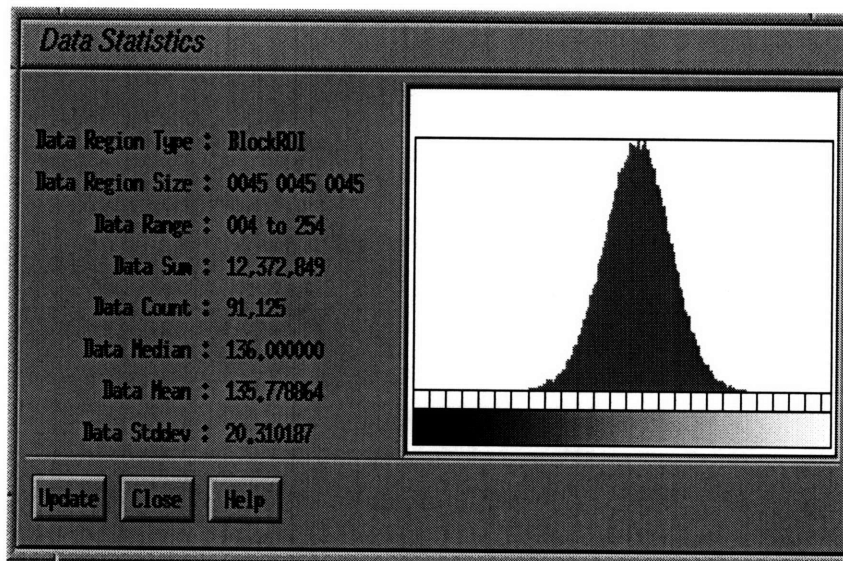
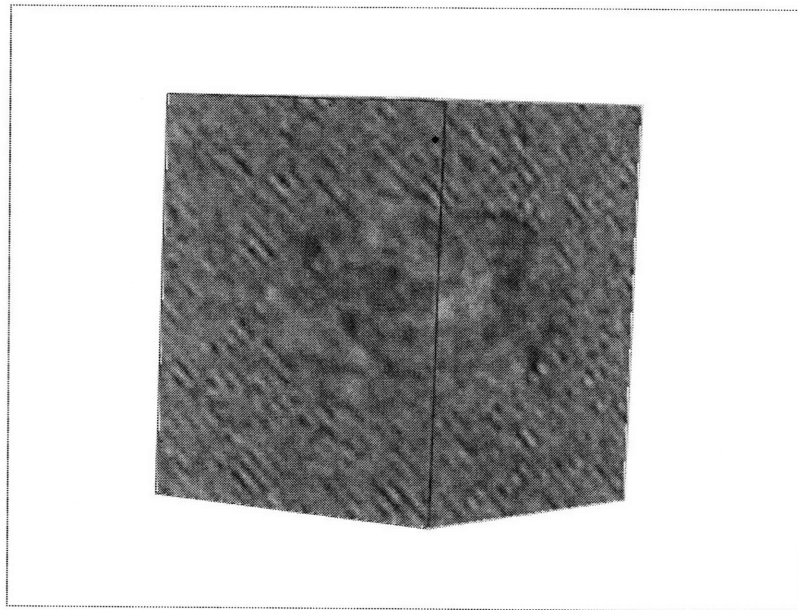


Figure 4.1: A 46x46x46 textured image of a sphere generated using the 3D GMRF model. Inside model parameters are $\{b_0, b_1, b_2\} = \{0.3, 0.0, 0.0\}$; outside model parameters are $\{b_0, b_1, b_2\} = \{-0.3, 0.0, 0.0\}$. The associated histogram shows that the two textures cannot be distinguished by using raw gray level alone.

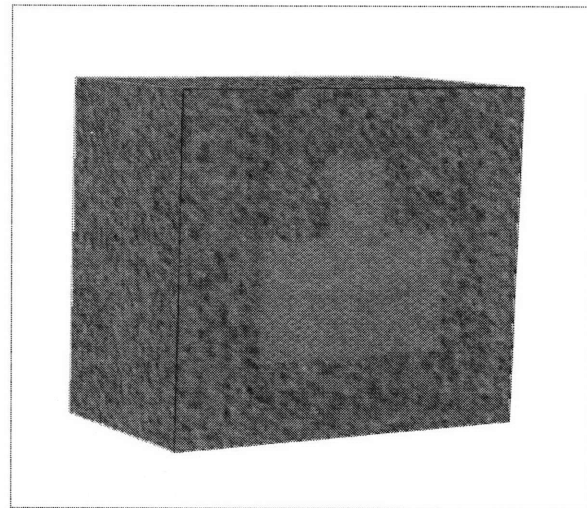
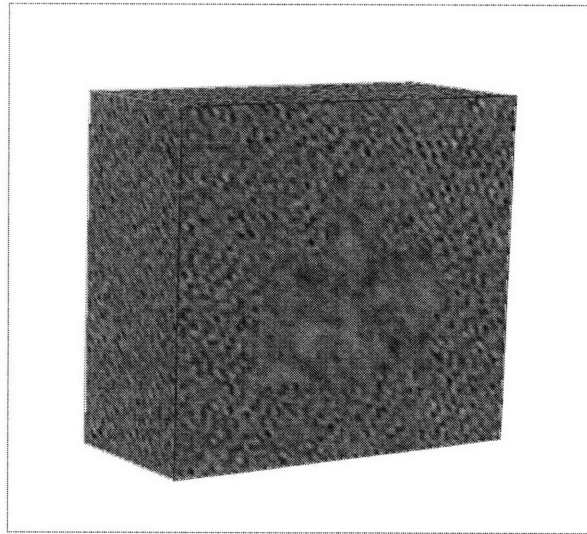


Figure 4.2: 64x64x64 textured images of tcubes. The top textured image was generated using a 3D GMRF model with inside parameters $\{b_0, b_1, b_2\} = \{0.2, 0.1, 0.0\}$ and outside parameters $\{b_0, b_1, b_2\} = \{-0.4, -0.4, 0.1\}$. The bottom textured image was generated assuming a Gaussian intensity distribution with inside mean = 180.0, inside variance = 400.0 and outside mean = 120.0, outside variance = 2,500.0.

of tcube data. We thus have a total of nine different synthetic test images.

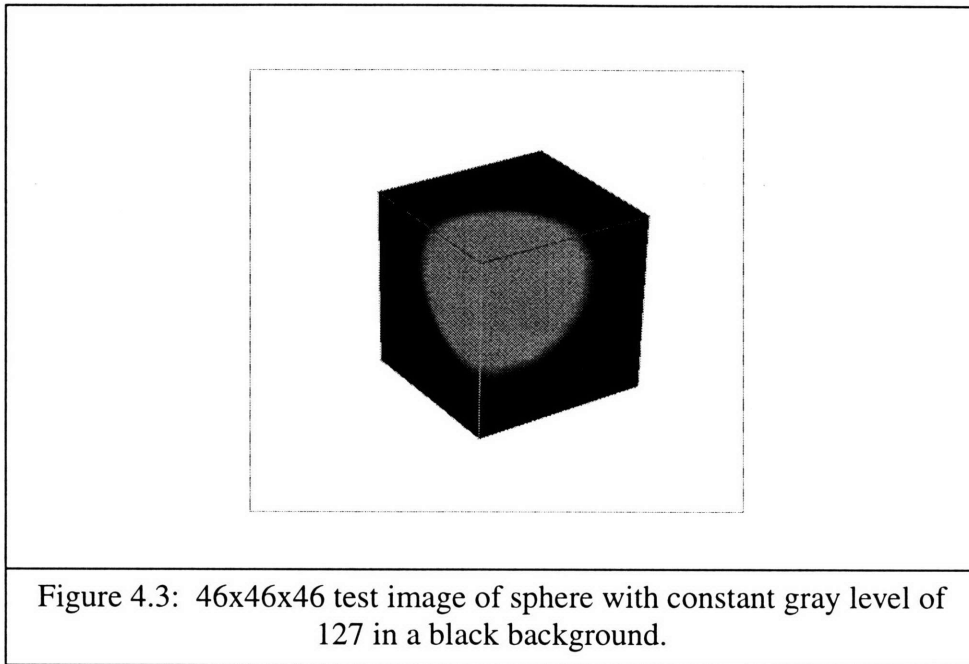
26x82x22 and 64x77x29 volumes of data were selected from a larger 256x256x36 MRI image of a porpoise as described in section 2.5.3 in chapter 2. Examples of the selected regions of data will be presented with the results in section 4.4. It should be noted that the 26x82x22 region is meant to be a portion of the MRI data that is easier to segment than the 64x77x29 region. As shown in chapter 2, the 64x77x29 image contains fairly complex boundaries of the brain (see figure 2.8 and figure 2.9). In order to simplify the discussion throughout this chapter, each of the synthetic test images is given a name which will be used to refer to the textured image in subsequent sections. Table 4.1 presents the list of names for each of the nine synthetic images.

4.2 Standard Marching Cubes

This section serves to introduce the reader to how the standard marching cubes algorithm performs on images of constant gray levels and on images with texture. We run the marching cubes algorithm on a 46x46x46 test image of a sphere with gray level 127 in a background of gray level zero (black). The marching cubes algorithm does well for this image which is shown in figure 4.3. An optimal threshold may be selected that distinguishes one volume of constant gray level from another volume of constant gray level. The result of the algorithm is shown in the top of figure 4.4.

Next the marching cubes algorithm is applied to a synthetic textured test image,

Table 4.1: Listing of the names used to refer to the synthetic test images.	
46x46x46 textured sphere generated by 3D GMRF model: inside parameters $\{b_0, b_1, b_2\} = \{0.2, 0.1, 0.0\}$, outside parameters $\{b_0, b_1, b_2\} = \{-0.4, -0.4, 0.1\}$	texsphere1
64x64x64 textured tcube generated by 3D GMRF model: (same parameters as above)	tcube64_1
84x84x84 textured tcube generated by 3D GMRF model: (same parameters as above)	tcube84_1
46x46x46 textured sphere generated by 3D GMRF model: inside parameters $\{b_0, b_1, b_2\} = \{0.3, 0.0, 0.0\}$, outside parameters $\{b_0, b_1, b_2\} = \{-0.3, 0.0, 0.0\}$	texsphere2
64x64x64 textured tcube generated by 3D GMRF model: (same parameters as above)	tcube64_2
84x84x84 textured tcube generated by 3D GMRF model: (same parameters as above)	tcube84_2
46x46x46 textured sphere generated by Gaussian distribution: inside mean = 180, var = 400; outside mean = 120, var = 2,500	gauss_sphere
64x64x64 textured tcube generated by Gaussian distribution: (same parameters as above)	gauss_tcube64
84x84x84 textured tcube generated by Gaussian distribution: (same parameters as above)	gauss_tcube84



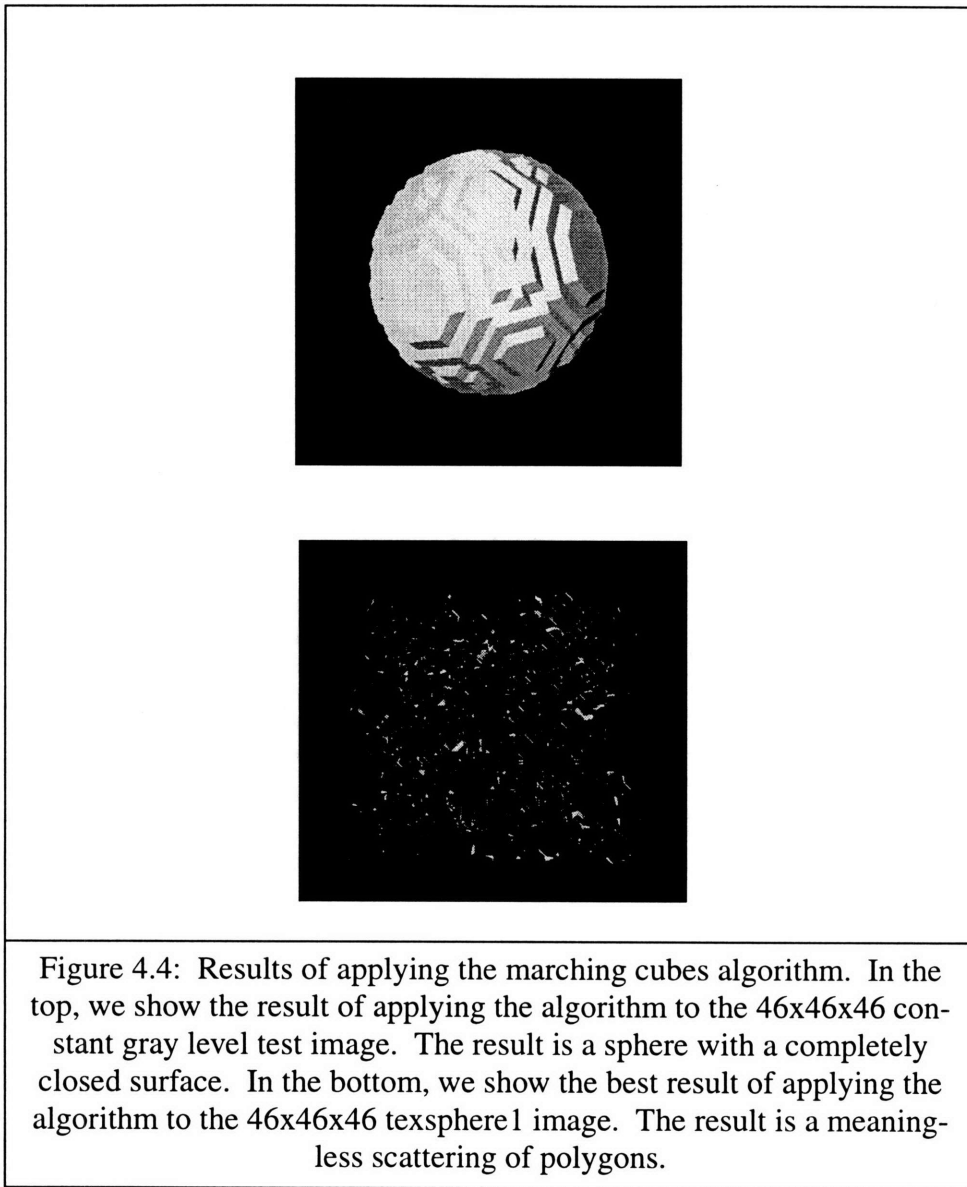
texsphere1. The result is shown in the bottom of figure 4.4. As expected, the marching cubes algorithm cannot construct surfaces between volumes of different textures. The result is a meaningless scattering of polygons. No surface result is obtained for all thresholds tried. It should be noted that we do not give any examples of applying the surface-based segmentation method using texture features to the constant gray level image in figure 4.3. The reason is that our method inherently requires that the input be a textured image. This is not surprising when it is realized that the texture features (which are incorporated into our method) have no meaning when computed from local volumes of constant gray level.

4.3 Tests With Synthetic Images

Many supervised image segmentation methods enable researchers to select a set of optimal features in order to classify or segment images with a low degree of error [Muzzolini, *et al.*, 1994; Liu and Jernigan, 1990; Ohanian and Dubes, 1992]. Most of these methods of selecting a set of optimal features either assume some *a priori* knowledge or access to a set of training images.

Here no emphasis is placed on finding optimal features primarily because we do not assume *a priori* knowledge nor do we use training images in this approach. Furthermore, the viewpoint of this thesis agrees with that expressed by Geman *et al.* (1990); that is, rather than search for the set of optimal features that varies depending on the image analyzed, it is better to find a way to integrate multiple and even redundant features. We speculate that this can be simplistically accomplished by using a feature vector, and therefore we group features together in an arbitrary manner.

In this section, we first present some preliminary results using selected single features and subsequently proceed to describe the experimental results of using first-order features, 3D co-occurrence matrix based features, power spectrum features, and 3D GMRF model based features in our algorithm. Finally, all the features implemented are combined into one feature vector and tested. As stated at the end of section 3.5.1 in chapter 3, threshold values for the normalized Euclidean distance measure were determined through a tedious trial and error process. We give the threshold values used in each specific case.



4.3.1 Single Feature Tests

Some tests were performed using individual features. When using single features, our algorithm compares the computed texture feature value for a vertex with a user speci-

fied texture threshold to determine surface-edge intersections in a cube. Note that this is different than the general algorithm's method of using the normalized Euclidean distance measure. Here threshold values are not normalized, and the difference between the magnitude of feature values will be clear. Thus it will also be clear why a normalized Euclidean distance measure must be used to take into account that some feature values may be inherently larger than other feature values. We briefly show the results of individual features to give the reader some sense of the performance of these features. Results of using single features varied from poor to fairly good, and here we show some of the better results.

For example, we apply the segmentation algorithm using the gray level gradient feature computed from 3x3x3 local volumes. When this method was tested on *texsphere2*, a threshold of 180 gave the constructed surface shown in figure 4.5. The surface of the sphere has been visibly constructed with 12,332 polygons, but it is by no means a smooth, complete surface. We also present the segmented surface obtained by applying the residual feature (average deviation of residual values; see section 3.1 of chapter 3) to *texsphere1*. Using a threshold of 8 gives the 23,924 polygon surface result shown in figure 4.6. The sphere in figure 4.6 has more of a complete surface than that shown in figure 4.5. But there are random polygons located away from the sphere's surface in figure 4.6 that are not found in figure 4.5.

We next present two results that are computed using features from the Fourier power spectrum. The first result involves the 3D average power (AP) feature and is shown in figure 4.7. Here a surface of 8,292 polygons is generated to form what seems to be a "partial sphere." That is, the overall shape is spherical although the structure resembles

something more like the semicircular canals in the inner ear. The AP feature generally returned fairly good results when applied to the synthetic textures created. The sphere in figure 4.8 has the appearance of a fairly smooth, coherent surface. This sphere's surface is composed of 12,092 polygons. The 3D entropy (EPY) feature generally allows for reliable construction of surfaces between the textures in our synthetic data sets. When comparing the segmentations of figures 4.5 - 4.8, the segmentation of figure 4.8 is qualitatively the best.

It is impossible to show here, but when rendering the polygons for the surface in figure 4.8, it was observed that there are actually multi-layers of surfaces that are constructed inside of the outermost surface that is visible in the figure. This problem will occur again. In fact, there is a similar scenario for the surface of figure 4.6. Surfaces inside of the outermost surface (which is visible in the figure) are observed when the polygons are rendered in real time. The reason for this involves the way that the volume data is processed as well as the method of computing local volumes for estimation of texture features in the segmentation method.

Consider that we are processing the volume data, marching from one cube to the next. If a surface-edge intersection is determined to exist for edge 1 of a particular cube, the next cube to be processed will most likely result in a determination of a surface cut along its edge 1 also. This is because the local volumes used to determine texture features for the current cube contain most of the same gray values of the local volumes of the previous cube that has already been processed. The problem is most evident when the resulting segmentations consist of well connected polygons, giving the appearance of a smooth surface. When the surfaces are not so "smooth," the multi-layer problem is harder to

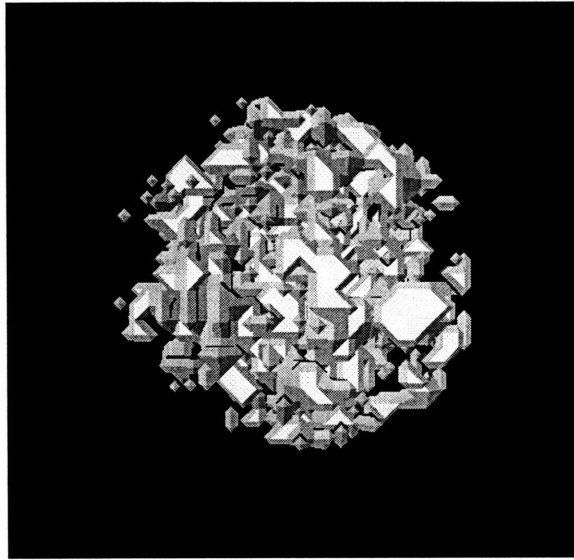


Figure 4.5: Surface constructed using the gradient feature for a threshold of 180 on texsphere2.

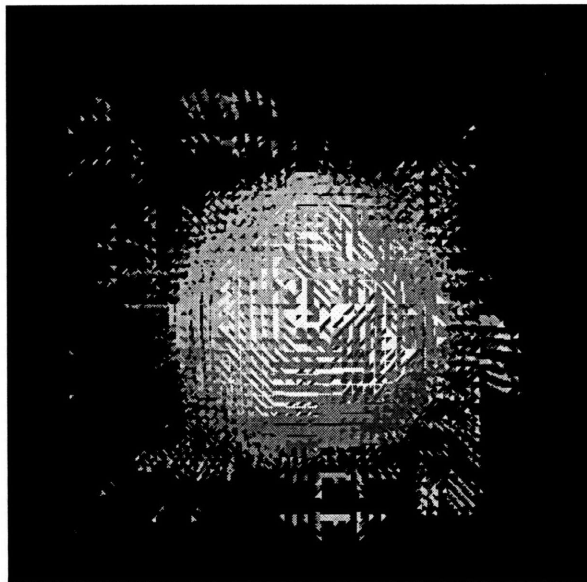


Figure 4.6: Surface constructed using residual features for a threshold of 8 on texsphere1.

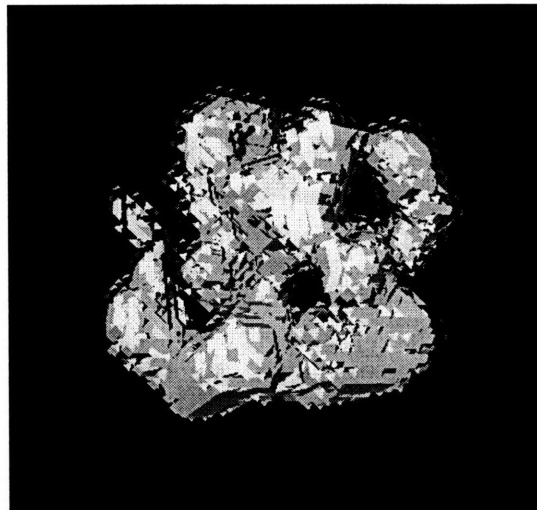


Figure 4.7: Surface constructed using 3D average power (AP) feature for a threshold of 85,000,000 on texsphere2.

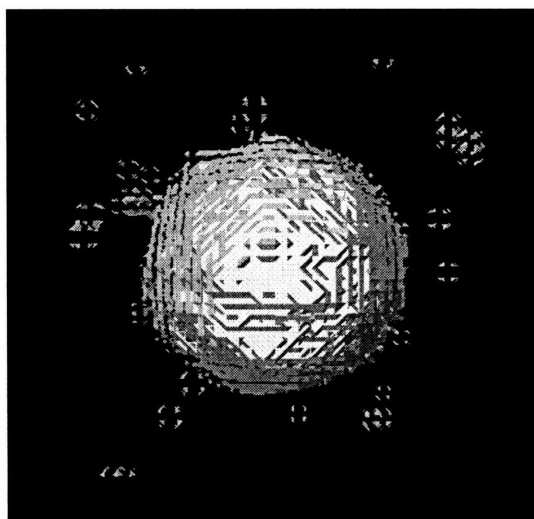


Figure 4.8: Surface constructed using 3D entropy (EPY) feature for a threshold of 0.03 on gauss_sphere.

recognize. This is one weakness of the segmentation algorithm used. The problem is recognized and solutions to the problem will be left up to future investigations.

4.3.2 Using the Feature Vector with First-Order Statistical Features

All first-order statistical measures were placed into the feature vector to test the algorithm. Using the three types of synthetic textured sphere images, it was not possible to get any meaningful results. The output of the algorithm consisted mostly of randomly scattered polygons for the synthetic images; no sphere surfaces were found. This may have been expected because some of these features are more suited to non-textured type images. The reason for the failure here is not too clear.

4.3.3 Using the Feature Vector with 3D Co-occurrence Matrix Features

Recall from section 3.2.1 that an important parameter associated with the implementation of 3D co-occurrence matrix based features is the size of the pooled GLCM $C(1)$, namely $G \times G$ where G is the number of gray levels in the image being analyzed. We bin the number of gray level values (which is usually larger than G) of the original image in order to obtain an image with G gray level values from which to compute the GLCM. This is done in order to reduce the amount of data that needs to be dealt with which corresponds to a reduction in computation time. As described in section 3.2.1, a reasonable value of G needs to be picked to keep enough resolution in the image to be able to still dis-

criminate textures. Here experimental values of $G = 8$, $G = 16$, and $G = 32$ were used. Results varied little among these choices, and $G = 16$ was kept as the standard for all subsequent experimental tests.

Both the first version, or the direct 3D GLCM method, and the second version, or the 3D GLCM separate planes method, give basically similar results. Figure 4.9 contains the results of running the algorithm with just a single contrast (CON) feature for both the direct method and the separate planes method. Further details are given in the figure caption. Clearly, the two results are fairly similar. The computation time needed to compute texture features associated with the separate planes method is significantly greater than that required by the direct 3D version. Therefore, for the experiments detailed here, the direct 3D method of computing the 3D GLCM was always used because the performance of the separate planes method did not warrant the computational effort. The specific 3D GLCM features that were used here are the CON, ASM, ENT, and COR features as described in section 3.2.1.

This section presents two results of using the feature vector of 3D GLCM features. Figure 4.10 shows the result of using a feature vector of 3D GLCM features with a threshold of 0.4 for `texsphere1`. Ideally, we should obtain a distinct surface of a sphere, but here we see a broken surface outline within a scattering of polygons. This is one of the better results from experiments with thresholds. In figure 4.11, we see that pieces of the surface of a sphere for `gauss_sphere` were segmented. The threshold required is 0.5 which produces 8,181 polygons. The results in figure 4.10 and figure 4.11 represent the best segmentations for using only 3D GLCM features in the feature vector.

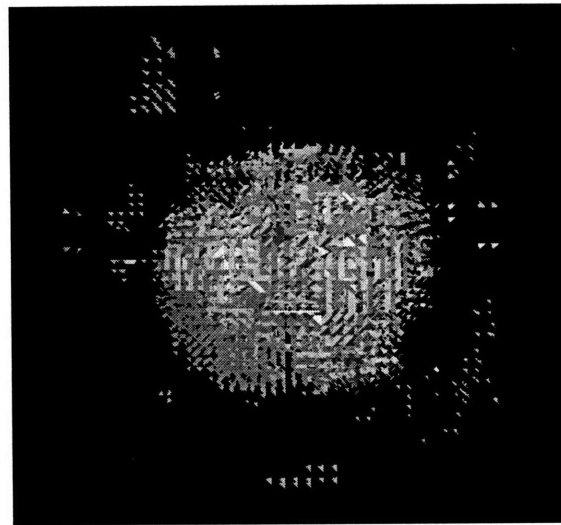
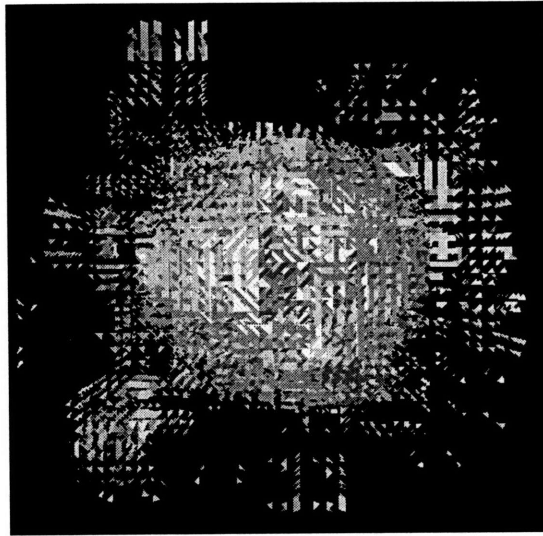


Figure 4.9: The top sphere is the result of using the CON feature of the direct 3D GLCM method with a threshold of 10 on texsphere1. The bottom sphere is the result of using the CON feature of the 2D separate planes method with thresholds of 7, 7, and 7 (one threshold for each of the xy, zx, yz plane computed CON values) on texsphere1 also.

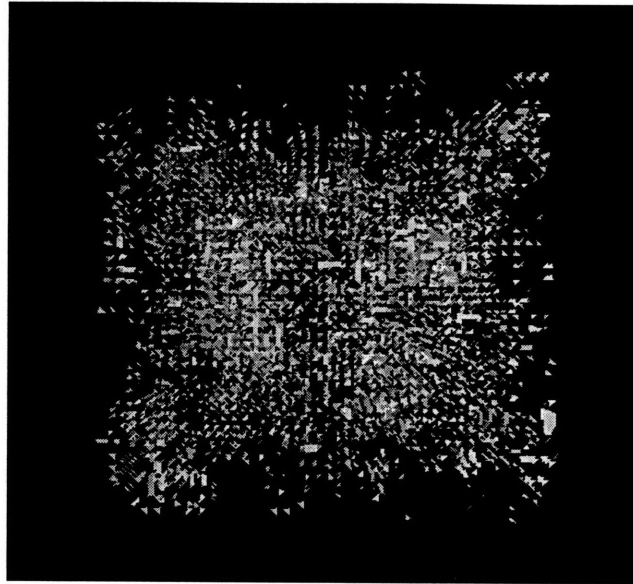


Figure 4.10: Surface outline is visible here using feature vector of 3D GLCM features with a threshold of 0.4 for texsphere1 synthetic data. 6x6x6 local volumes were used to compute the features. 7,519 polygons make up the whole picture.

A weakness of the algorithm is apparent here in that many undesired triangular polygons are being constructed. This is in addition to the connectivity problem. That is, the polygons that should make up the segmented surface are not well connected which results in fragmented surfaces. In section 4.3.2, the multi-layer problem was recognized. Although more difficult to recognize here, inspection shows that surface polygons are constructed in a manner that is indicative of the problem of having identical surfaces constructed one just inside the other.

Both these results were obtained by calculating the 3D GLCM features over 6x6x6 local volumes. Experimentally we found that using 6x6x6 local volumes to compute tex-

ture features gave better results than using local 4x4x4 volumes. The more the data available from which to calculate features, the more accurate the feature estimate. The 6x6x6 local volumes provide for computations of more accurate and effective GLCM features.

We attempted to repeat the results for `texsphere1` and `gauss_sphere` on their corresponding larger `tcube` images, `tcube64_1` and `gauss_tcube64`. Unfortunately, similar results were not able to be repeated despite the various thresholds tried. The output of the algorithm using `tcube64_1` and `gauss_tcube64` were basically scattered polygons. Since the result for `texsphere1` was a faint outline of a surface to begin with, the result for `tcube64_1` is not too surprising, but a better result for `gauss_tcube64` was expected.

4.3.4 Using the Feature Vector with Fourier Power Spectrum Features

The only local volumes we implemented that may be used with the power spectrum features are the 2x2x2 and 4x4x4 local volumes (remember the volumes must have dimensions that are powers of 2). Trial and error showed that we can neglect the 2x2x2 local volumes, and so all experiments presented here involve computing power spectrum features on local 4x4x4 volumes. We implement only four 3D power spectrum features which are the 3D average power (AP), the 3D entropy (EPY), the 3D maximum energy (ME), and the squared maximum energy frequency (SQMEfreq) described in section 3.3.2.

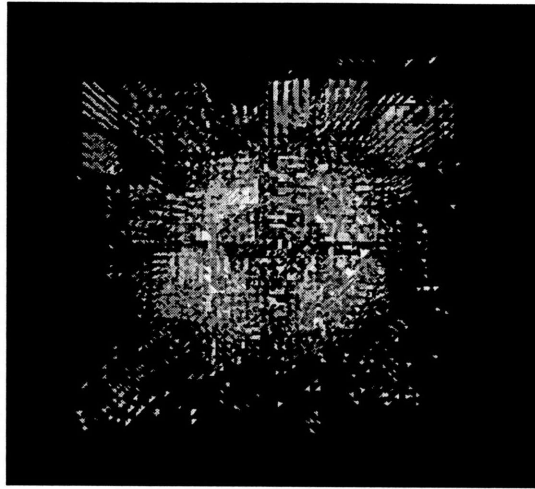


Figure 4.11: Pieces of the sphere surface are constructed for `gauss_sphere` using a feature vector containing 3D GLCM features with a threshold of 0.5. 8,181 polygons make up the polygonated data. We used 6x6x6 local volumes to compute the features.

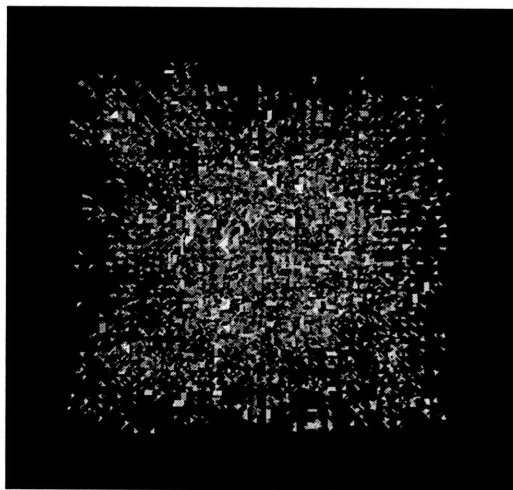


Figure 4.12: Surface of textured sphere in `texsphere1` constructed using a feature vector of 3D Fourier power spectrum features with a threshold of 0.3.

We now proceed to describe the results of applying the algorithm to each of the 46x46x46 textured sphere images, namely `texsphere1`, `texsphere2`, and `gauss_sphere`. In figure 4.12 we present the result of using a threshold of 0.3 on `texsphere1` to give a 6,776 polygon output. Although there are many unwanted polygons scattered about, the sphere's fragmented surface is still visible. In figure 4.13, the result of applying our algorithm to `texsphere2` is shown. The best threshold found was 0.3 which still gave a scattered polygon data output. This figure primarily shows the failure of our method of using a feature vector containing power spectrum features on `texsphere2`.

Lastly, in figure 4.14 we show the result for `gauss_sphere`. We use a threshold of 0.33 to produce the 11,643 polygon result. It can be observed that the surface of the sphere is only partially constructed. Although the results of using the textured cube images are not presented here, note that similar results are obtained if the same thresholds were used on the larger synthetic cube images. The problems of surface connectivity and multi-layers reveal themselves in all these results.

4.3.5 Using the Feature Vector with 3D GMRF Model Features

As noted in section 4.3.3 for 3D GLCM features, using 6x6x6 local volumes for the estimation of GMRF model parameters produces better results than the local 4x4x4 volumes. Thus for the presentation of results in this section, all experimental tests were conducted using 6x6x6 local volumes. It was also found that the results for the first version of the 3D GMRF model (or the direct 3D extension method) and the second version of the 3D GMRF model (or the separate 2D plane method) gave fairly similar results,

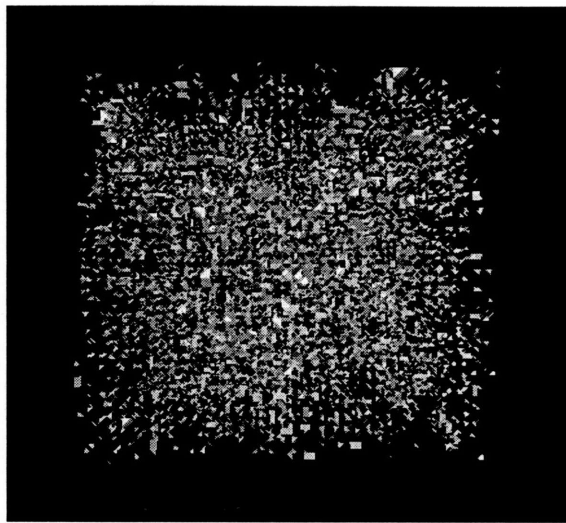


Figure 4.13: 9,931 polygon result of applying algorithm to texsphere2 with a threshold of 0.3 using a feature vector of 3D power spectrum features.

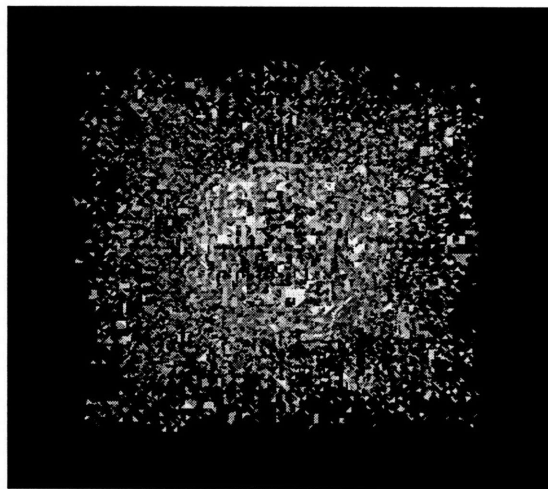


Figure 4.14: Surface created from gauss_sphere with a feature vector of 3D power spectrum features using a threshold of 0.33. Data output contains 11,643 polygons.

analogous to the case for the two versions of the 3D GLCM. Since the separate 2D plane version of the 3D GMRF model requires far more computational time for feature estimation, only the direct 3D version of the 3D GMRF model was used in the experimental tests presented here. These two versions of the GMRF model are described in section 3.4.2.

Before presenting specific results, consider some estimated 3D GMRF model parameters in tables 4.2 and 4.3. These estimates demonstrate that using 6x6x6 local volumes to compute the model parameters is indeed better than using 4x4x4 local volumes. In the tables, {GMRF coeff1,GMRF coeff2,GMRF coeff3} represent the model parameters $\{b_0, b_1, b_2\}$. Table 4.2 lists the first and second outside texture parameters of *texsphere2* found by using the direct 3D version of the 3D GMRF model. Recall from table 4.1 that the outside parameters for *texsphere2* are GMRF coeff1 = -0.3 and GMRF coeff2 = 0.0. It is clear that estimates using the 6x6x6 local volume are better.

Table 4.3 shows the same effect when the 2D separate planes version of the 3D GMRF model is used to estimate model parameters. Parameters for each of the xy, zx, and yz planes are shown. Comparing the parameter estimates in tables 4.2 and 4.3 serves to demonstrate that there is little difference between the respective estimates. This further supports the observation that the two versions of the 3D GMRF model give fairly similar results.

Furthermore, the numbers in tables 4.2 and 4.3 validate the idea of extending the original 2D GMRF model to the 3D GMRF model since the estimates are close to the actual values used to create the synthetic image *texsphere2*. Similar results were obtained for other synthetic images generated from using the 3D GMRF model such as *texsphere1*. Recall that in section 3.4.2 of chapter 3 it was stated that the validity of the 2D GMRF

equations given in section 3.4.1 should be maintained after our extension to 3D. Here, we see that experimental data support that assertion.

Table 4.2: Computed estimates of outside texture parameters for texsphere2. Direct 3D version of 3D GMRF model.			
4x4x4 local vol GMRF coeff1	6x6x6 local vol GMRF coeff1	4x4x4 local vol GMRF coeff2	6x6x6 local vol GMRF coeff2
-0.727774	-0.273329	0.268405	-0.060686
-0.089488	-0.339727	-0.192841	-0.007875
-0.514029	-0.221725	0.273959	-0.030902
-0.450977	-0.250016	-0.189747	0.041631
-0.391720	-0.239818	-0.017821	-0.068062
-0.185505	-0.335651	0.156393	-0.074564
-0.453078	-0.181049	-0.019042	-0.025621
-0.173736	-0.247778	-0.126426	-0.033819
-0.123691	-0.260219	0.159453	-0.063481
-0.190796	-0.261243	0.570558	0.010917
-0.727774	-0.243300	0.097824	-0.089353
-0.450977	-0.228161	-0.192388	0.051165
-0.201023	-0.248807	-0.059496	-0.039052
0.072886	-0.297159	0.033579	-0.057055

We now present a few experimental results of the segmentation algorithm which uses a feature vector containing 3D GMRF model features (model parameters and model variance). It is noted that the GMRF variance estimate σ^{2*} is made robust against illumination changes (as suggested by Chellappa and Chatterjee) by estimating the variance v of the gray values in the local 6x6x6 volume in which σ^{2*} is computed and then taking the actual variance estimate used as σ^{2*} / v [Chellappa and Chatterjee, 1985]. Figure 4.15 shows the segmentation of texsphere1 using a threshold of 2.5. The surface shown

Table 4.3: Computed estimates of outside texture parameters for texsphere2. The 2D separate planes version of the 3D GMRF model	
4x4x4 local volumes	6x6x6 local volumes
GMRFcoeff1 XY	GMRFcoeff1 XY
-1.120273	-0.248799
-0.007710	-0.319316
0.293797	-0.240776
0.022238	-0.251277
-0.272150	-0.304666
0.190108	-0.281753
0.698612	-0.309436
-0.088603	-0.291968
-0.084484	-0.210788
-1.010190	-0.333319
0.469692	-0.221759
GMRFcoeff1 ZX	GMRFcoeff1 ZX
0.045779	-0.252385
0.180122	-0.442120
-0.580335	-0.250285
2.682306	-0.304165
2.682306	-0.288385
1.070641	-0.328052
0.898850	-0.177385
0.935060	-0.248733
1.257115	-0.336672
-1.107089	-0.371331
-0.783175	-0.212147
-0.895612	-0.248080
GMRFcoeff1 YZ	GMRFcoeff1 YZ
-0.431926	-0.269405
-0.388591	-0.344647
-0.406919	-0.385738
-0.357345	-0.313670
-0.384852	-0.236435
-0.419296	-0.241507
0.309672	-0.315439
0.740886	-0.262399
-0.477694	-0.263383
-0.442350	-0.303366
-0.166437	-0.376672

consists of 14,514 polygons. Here again we notice the multi-layer and surface polygon connectivity problems mentioned in previous sections.

This result is the best we have obtained to date when using the texture feature vector in our segmentation algorithm. This suggests two things. One is that because the synthetic image `texsphere1` is generated using a 3D GMRF model, it is not surprising that using the same 3D GMRF model to characterize the textures in the image gives good results. The second is that it demonstrates our segmentation algorithm is valid since GMRF generated textures can be segmented with GMRF model features using this approach.

Figure 4.17 shows that the threshold of 2.5 also gives a fairly good segmentation of `tcube84_1` consisting of 41,941 polygons. In figure 4.16 and 4.18, the surface-based segmentations are poor. The polygons form an approximate outline of the texture boundary, but the algorithm does not perform well on these synthetic images. The synthetic images generated with a Gaussian intensity distribution do not produce good results; only scattered polygons were obtained. This suggests that Gaussian generated textures do not produce good results because we are estimating GMRF model features when the image is generated with a different texture model.

4.3.6 Using the Feature Vector With All Features

The ultimate goal for this thesis was to combine all the computed features into one feature vector that contains information on the texture within the $N \times N \times N$ volume in which the texture features are computed. Since no previous work has been found on computing

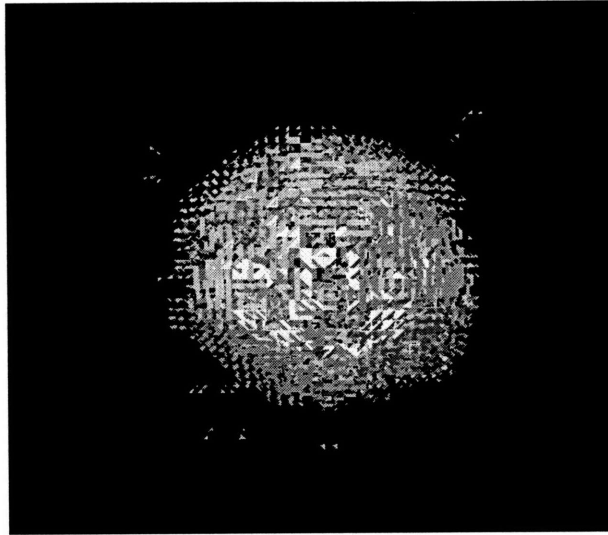


Figure 4.15: Segmentation of texpshere1 using feature vector of 3D GMRF model parameters and variance with a threshold of 2.5.

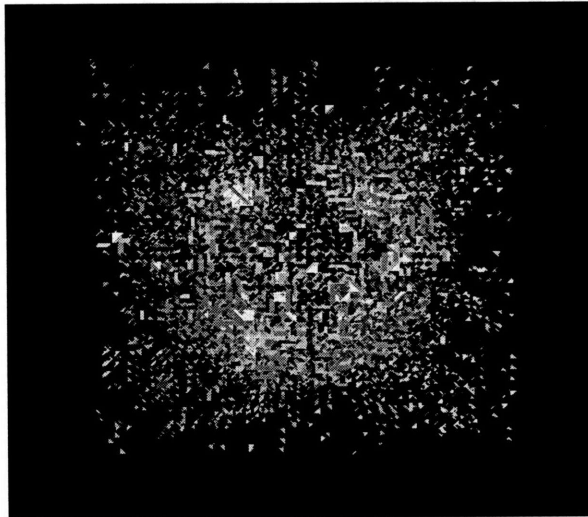


Figure 4.16: Segmentation of texpshere2 using feature vector of 3D GMRF model parameters and variance with a threshold of 3.0. 8,396 polygons in output.

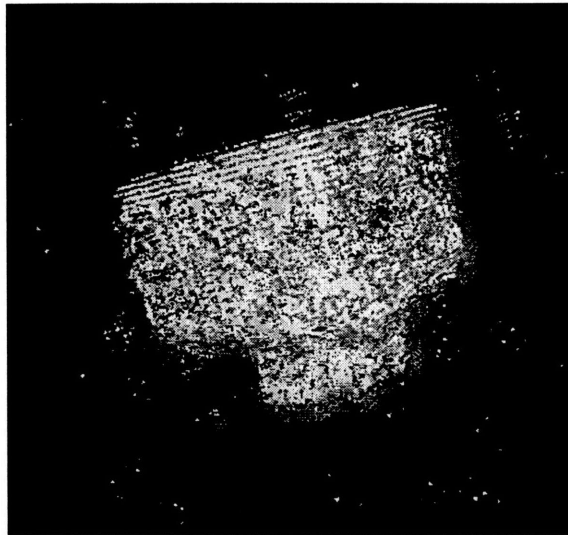


Figure 4.17: Segmentation of tcube84_1 using feature vector of 3D GMRF model parameters and variance with a threshold of 2.5.

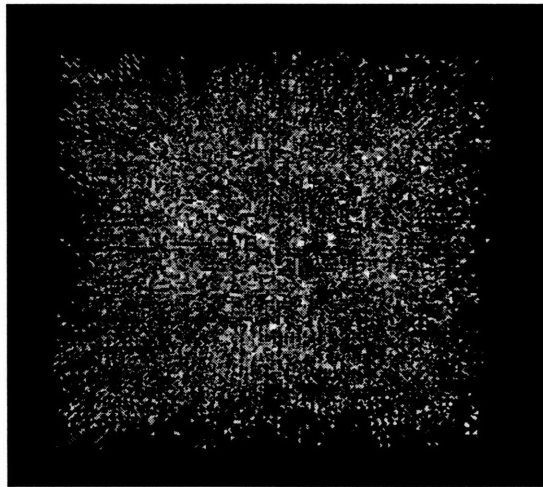


Figure 4.18: Segmentation of tcube64_2 using feature vector of 3D GMRF model parameters and variance with a threshold of 3.0. 17,123 polygons.

and utilizing 3D texture features, the previous sections were necessary to demonstrate the performance of the several different types of 3D features. The results in subsections 4.2.1 to 4.2.5 therefore set the stage for this section by presenting the tests in a systematic manner.

As mentioned before, better texture features are obtained if we compute them from $6 \times 6 \times 6$ local volumes versus $4 \times 4 \times 4$ local volumes. Here only $6 \times 6 \times 6$ local volumes were used for all feature computations other than the first-order statistical measures and the power spectrum features. First-order statistical measures have their own version of local volumes and the power spectrum features are computed from $4 \times 4 \times 4$ volumes.

Figure 4.19 shows the surface-based segmentation of `texsphere1` using a threshold of 4.0. There are 7,420 polygons in this output. Note that the surface of the sphere is not complete or smooth, but it is a fairly good result. In fact, this result is qualitatively better than the surface-based segmentations in figures 4.20 and 4.21. The surfaces of the spheres in figures 4.20 and 4.21 appear less complete. In figure 4.22, the surface of the textured `tcube` in `tcube64_1` is constructed fairly well using a threshold of 4.0. On the other hand, in figure 4.23, a faint outline of the surface of the textured `tcube` in the image `gauss_tcube64` may exist, but it is difficult to detect.

Overall, these results are encouraging. They show that our approach to segmentation performs fairly well in extracting the surface boundaries between different synthetic textures despite the multi-layer and the surface connectivity problems. The hope was that if enough texture features are employed, then two distinct textured volumes will differ in at least one of the features. Placing the features in a vector and comparing feature

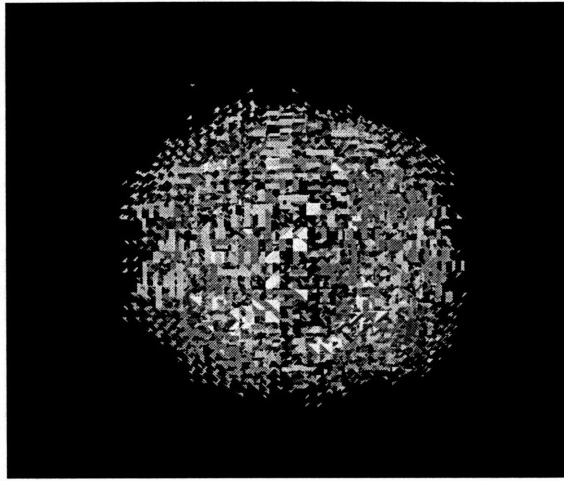


Figure 4.19: Surface segmentation of texsphere1 using all texture features in the feature vector. Threshold is 4.0.

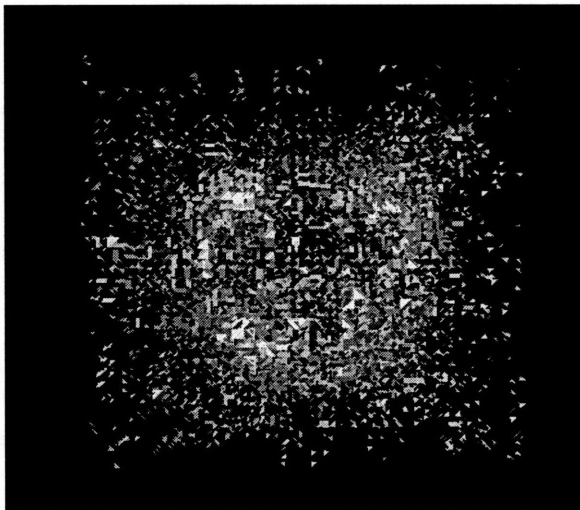


Figure 4.20: Surface segmentation of texsphere2 using all texture features in the feature vector with a threshold of 4.0. 8,648 polygons.

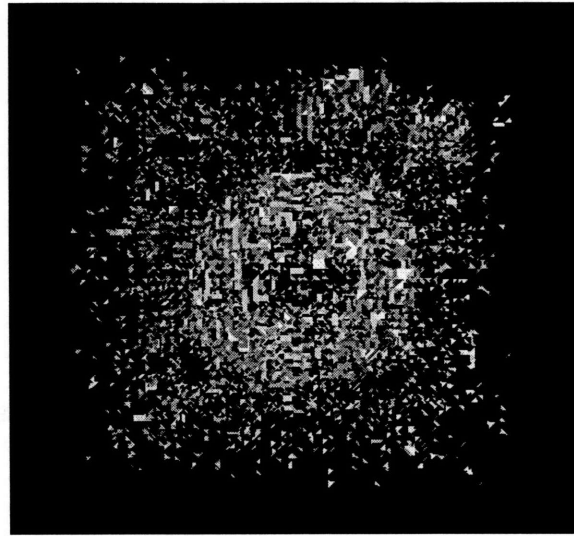


Figure 4.21: Surface of sphere in gauss_sphere using all texture features in the feature vector with a threshold of 5.0. 6,695 polygons.

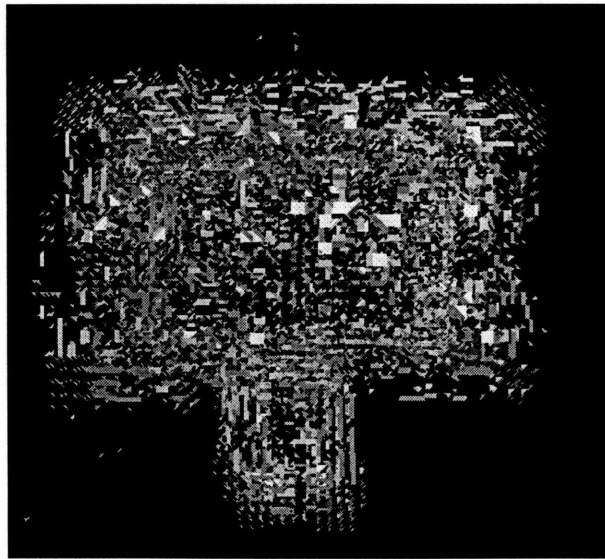
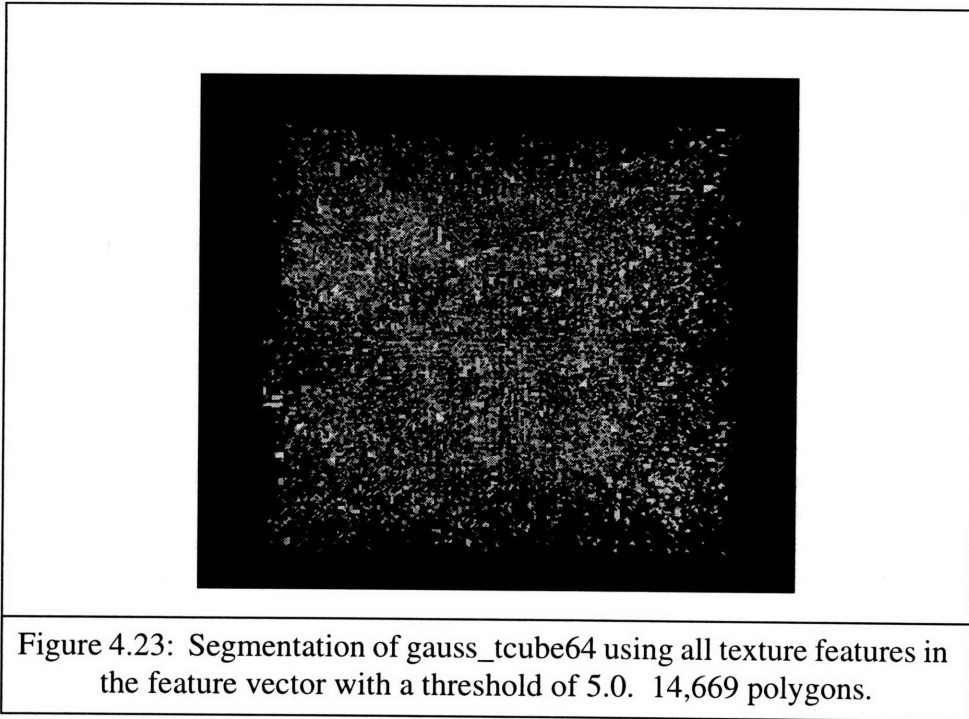


Figure 4.22: Segmentation of tcube64_1 using all texture features in the feature vector with a threshold of 4.0. 10,684 polygons.



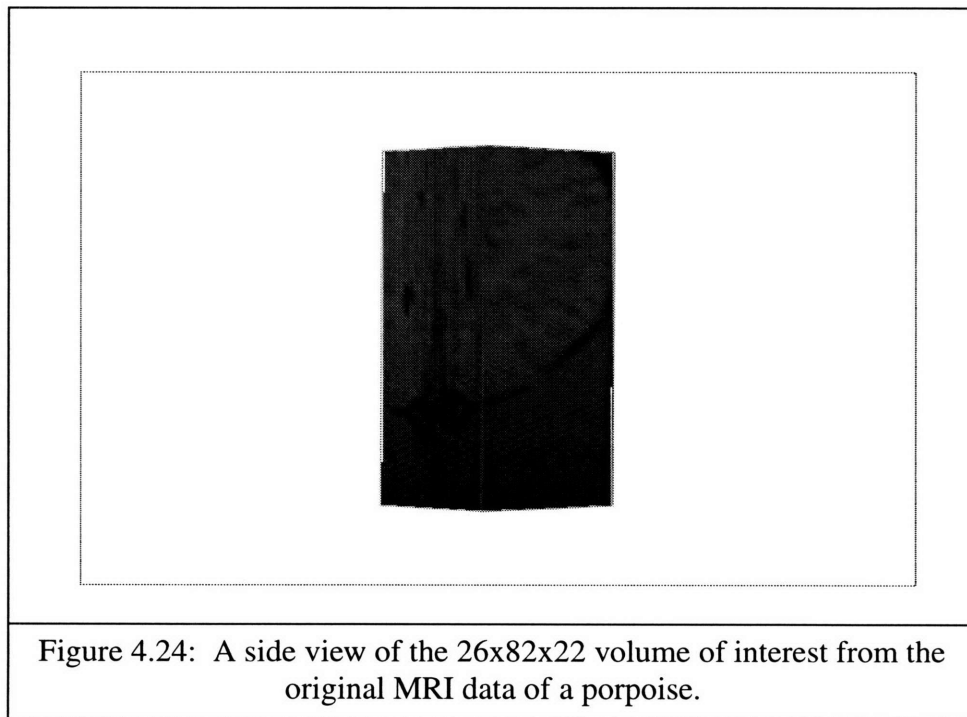
vectors using the normalized Euclidean distance measure appears to provide for an effective method of integrating multiple values while using only a single threshold.

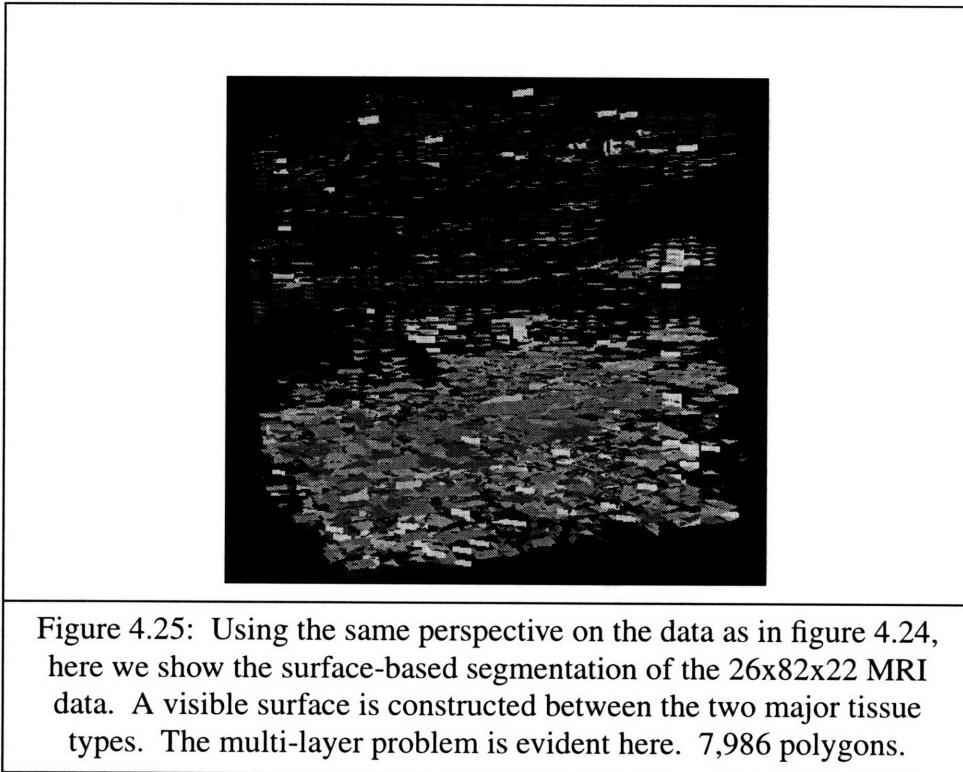
4.4 Tests with a Real Image

Section 4.3 showed the results of applying the segmentation algorithm to synthetic textured images. In this section, we test the algorithm on a real data set for a porpoise. Section 2.5.3 of chapter 2 and section 4.1 have briefly discussed the MRI data that were used. Specifically, we designated two different volumes of interest of varying size from the porpoise MRI data. The 3D regions of interest were obtained from the portion of the image that contains a boundary between brain tissue and other tissue located in the por-

poise head. In particular, we chose $26 \times 82 \times 22$ and $64 \times 77 \times 29$ sized regions for testing. The $26 \times 82 \times 22$ region was chosen to represent a simple boundary between two tissue types, and the $64 \times 77 \times 29$ region was chosen to represent a more complex boundary between soft and bony tissue in the head.

Although we have found that our segmentation method does not produce ideal results on the synthetic test images, the algorithm was tested on MRI data to obtain a feel for the algorithm's performance on real images. As mentioned earlier regarding the tests with synthetic images, the threshold values for the normalized Euclidean distance measure were determined through a tedious trial and error process. Here we also give the threshold values used in each specific case. Section 4.4.1 contains a discussion of the results of using the feature vector in the segmentation algorithm.

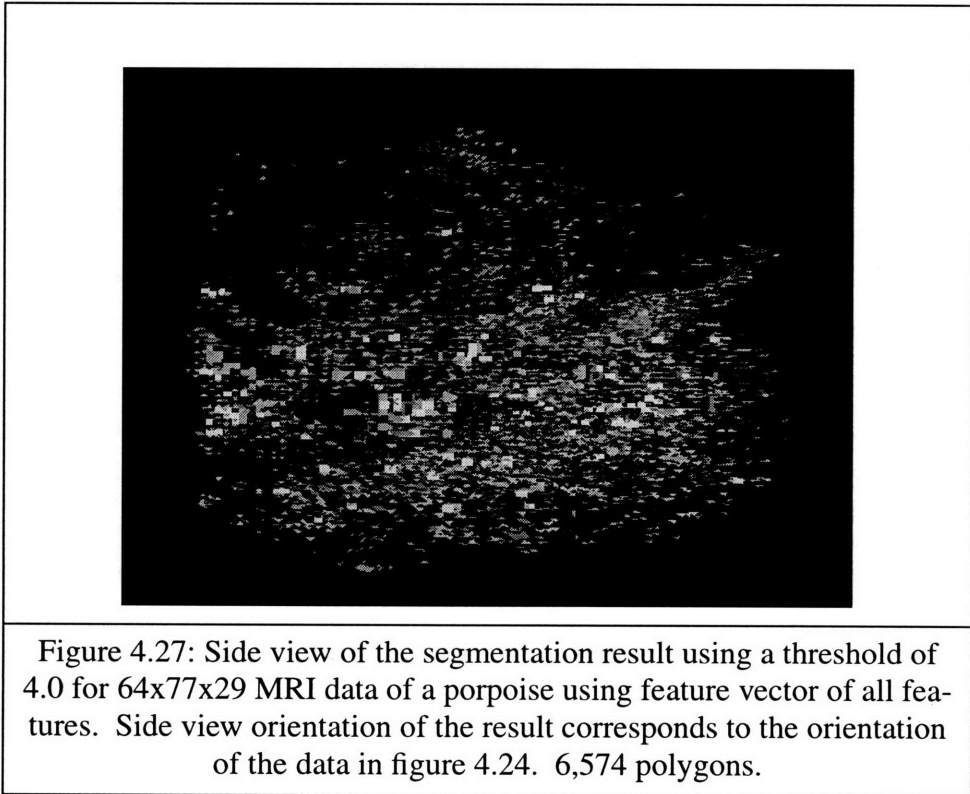
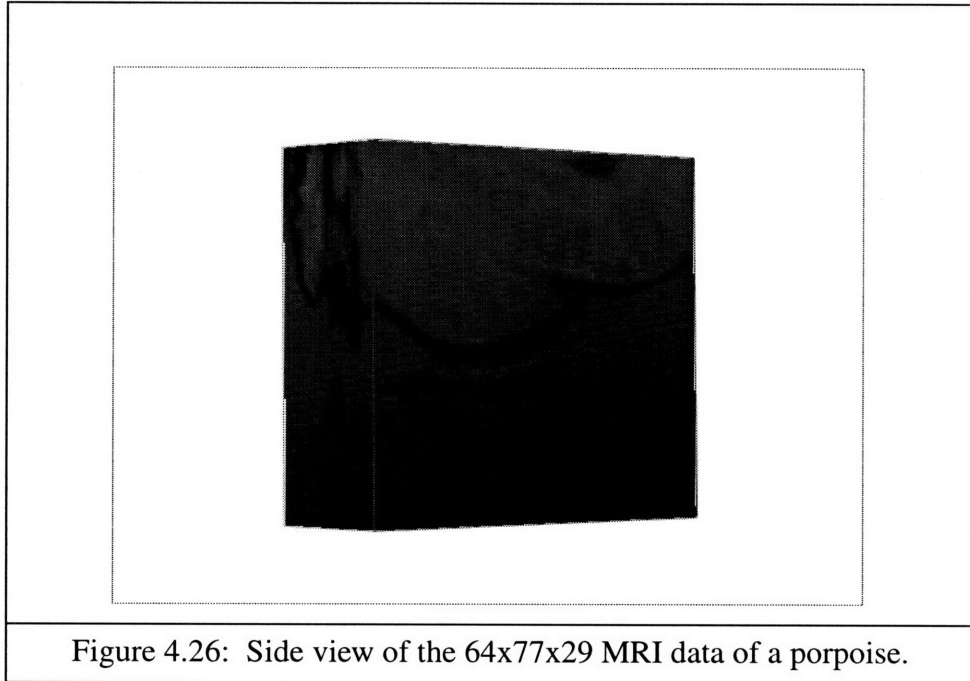




4.4.1 Using the Feature Vector

We applied the segmentation algorithm to the 26x82x22 and 64x77x29 real images using a vector containing features from the four categories of first-order statistics, 3D co-occurrence matrices, Fourier power spectra, and 3D GMRF models. The algorithm was also tested on the real data using a feature vector containing all the texture features combined, as was done for the synthetic test images in section 4.3.6.

Figure 4.24 shows the 26x82x22 data that were used. It should be noted in advance that the orientation of the 26x82x22 data is the same as the orientation of the result in figure 4.25. This is so that the two may be compared. As mentioned in section 4.1, we selected this as the simplest volume (containing a boundary between textured



regions) that we could find. Notice that the boundary of interest is the lower brain border in the figure. Ideally, we should get a curved surface representing this boundary as the output of the segmentation algorithm. Nevertheless, it should be noted that we are idealizing the image when it is said that the image contains just a boundary between textured regions, and so we should expect the results to diverge from the ideal.

The results from applying the algorithm to the 26x82x22 data using the feature vector containing features from each of the four categories gave similar results to that of using the feature vector containing all the texture features that were implemented. Thus only the result of the segmentation algorithm using the feature vector with all the texture features is presented. Figure 4.25 shows the 7,986 polygon result of applying the algorithm with a threshold of 2.8.

It is encouraging to note that, from the side perspective on the data and the result shown, there is a visible surface oriented diagonally from the bottom-left to the top-right corresponding to that in the raw input data. What is called a visible surface here is actually the boundary between the lower portion of the result, which contains many polygons, and the upper portion of the result, which contains visibly fewer polygons. The source of the many polygons below the boundary may be attributed to the multi-layer problem. The multi-layer problem is more evident here than in the tests for the synthetic images since the surface of interest is not the surface of a closed geometrical object. Overlapping local volumes within which texture features are computed result in the redundant placement of polygons which should ideally represent one single surface. Instead of obtaining a single surface, we obtain a lower region of many polygons and an upper region of few polygons. It is clear that the result is far from ideal but nonetheless encouraging since a pseudo-sur-

face is returned that seems to represent the brain boundary in the input image.

Next we applied the segmentation algorithm to the more complex $64 \times 77 \times 29$ real volume data using vectors containing features from the four categories of first-order statistics, 3D co-occurrence matrices, Fourier power spectra, and 3D GMRF models. Figure 4.26 shows the $64 \times 77 \times 29$ image data. The results were not encouraging. The algorithm using various feature vectors which contained measures from each of the four categories of the texture features did not perform well. The output was always a scattered group of sparsely located polygons. Thus we do not give examples of these results.

We also applied the segmentation algorithm using a feature vector containing all of the texture features that we have introduced. This too gave discouraging results which was to be expected. This illustrates that the complexity of the real image requires a more robust and complex approach to surface-based segmentation. Another consideration is that the real image contains multiple textures whereas the synthetic images we tested were designed to contain only two different textures. In any case, these are matters for future investigation.

We show the best result found by applying the segmentation algorithm to the $64 \times 77 \times 29$ MRI data, where we use a feature vector containing all of the texture features. It is difficult to make sense of the result shown in figure 4.27. Polygons are scattered in a random manner although there does seem to be clusters of polygons approximating possible boundaries. The result is oriented in the same manner as the image data shown in figure 4.26 so that the two may be compared visually.

4.5 Implementation Details

All tests were performed on either a SiliconGraphics Indy or a SiliconGraphics Indigo2. We have mentioned throughout this chapter that computational demands are a concern of our surface-based segmentation algorithm. The majority of the processing time is used in computing the texture features. Table 4.4 lists sample algorithm running times for different sized input data. It also lists sample running times when using different sets of texture features in the feature vector of the algorithm. The intent is to give the reader a rough idea of how long it takes to run the algorithm.

Note that in table 4.4, the last four entries are the running times of the algorithm using a single feature which is different from using the feature vector. The reader may compare each running time of the two pairs with the other running time in the pair to obtain an idea of the difference between running times for computing the direct 3D versions of the corresponding texture features and for computing the separate planes versions of those same texture features.

4.6 Summary

This thesis details the experimental results of using a surface-based segmentation algorithm with texture features. Testing was performed on a group of synthetic images and on MRI data of a porpoise. Single features were experimented with on the synthetic images. When using single features, this algorithm compares the computed texture fea-

Table 4.4: Sample running times for the segmentation algorithm on a SGI Indy.	
Volume Data	Time (minutes)
64x77x29 porpoise data (using feature vector with all texture features)	36
26x82x22 porpoise data (using feature vector with all texture features)	10
texsphere1 (using feature vector with all texture features)	25
tcube64_1 (using feature vector with all texture features)	80
texsphere1 (using feature vector with first-order statistical features)	3
texsphere1 (using feature vector with 3D co-occurrence features)	10
texsphere1 (using feature vectors with power spectrum features)	7
texsphere1 (using feature vectors with 3D GMRF features)	6
texsphere2 (using single CON feature; direct 3D co-occurrence method)	6
texsphere2 (using single CON feature; separate planes co-occurrence method)	10
texsphere2 (using single GMRF b_0 parameter; direct 3D method)	5
texsphere2 (using single GMRF b_0 parameter; separate planes method)	20

ture value for a vertex with a user specified texture threshold to determine surface-edge intersections in a cube.

The single feature trials gave results in which the output polygons were well connected, thereby forming smoother and more coherent surfaces than those obtained by using the algorithm with the feature vector. In particular, the single 3D average power (AP) feature was selected for use on the porpoise data and showed encouraging results.

Experiments with feature vectors containing first-order statistical features, 3D GLCM features, 3D Fourier power spectrum features, and 3D GMRF model features were also attempted. Results for the synthetic images varied from poor to encouraging. The result for the 26x82x22 porpoise data was a visible pseudo-surface, and the result for the 64x77x29 porpoise data consisted mainly of scattered polygons. The segmentation algorithm using feature vectors of first-order statistical measures failed to segment the surface between the two textures in the synthetic images. The algorithm, which used feature vectors containing the 3D GMRF model features, gave the best output for the synthetic test images generated using the 3D GMRF model and the worst for the test images generated using the Gaussian distribution assumption.

Above all, the segmentation algorithm using feature vectors containing all the texture features we implemented gave encouraging results for the synthetic images. Results for the synthetic image tests are summarized in table 4.5. In almost all cases, visible surfaces were generated. The results obtained by applying this segmentation method to the porpoise data was encouraging for the 26x82x22 data. The algorithm produces a visible surface although the multi-layer problem prevents the construction of a single surface. The 64x77x29 data serves to illustrate the failure of the algorithm when the input image

Table 4.5: Summary of Results Using the Algorithm with the Normalized Euclidean Distance Measure on Synthetic Sphere Images	
Volume Data	Qualitative Results
Vector with first-order statistical features	
texsphere1	scattered polygons
texsphere2	scattered polygons
gauss_sphere	scattered polygons
Vector with co-occurrence matrix features (6x6x6 local volumes; direct 3D version)	
texsphere1	sphere surface outline
texsphere2	scattered polygons
gauss_sphere	isolated surface patches give sphere surface outline
Vector with Fourier power spectrum features (4x4x4 local volumes)	
texsphere1	visible sphere surface but very fragmented
texsphere2	scattered polygons
gauss_sphere	partial sphere surface
Vector with GMRF model features (6x6x6 local volumes; direct 3D version)	
texsphere1	good sphere surface
texsphere2	partial, fragmented sphere surface
gauss_sphere	scattered polygons
Vector with all texture features	
texsphere1	good sphere surface
texsphere2	partial, fragmented sphere surface
gauss_sphere	partial, fragmented sphere surface

contains complex boundaries between varying regions of multiple textures.

These experimental tests demonstrated two main problems with the current algorithm. The first is the surface connectivity problem. When using feature vectors, the surface output of the algorithm is mostly fragmented. That is, the polygons that make up the surface between the two textures in the images are disconnected. The second weakness of the segmentation algorithm lies in the multi-layer problem. This arises because of the way local volumes overlap as the volume data is processed, resulting in the determination of redundant surface-edge intersections for multiple cubes. The overall result is that we get the creation of repetitive surfaces or polygon placements. These are important issues for future investigation.

Chapter 5

Conclusions

This thesis presents a method for partitioning textured volume data into homogeneous regions by locating the boundaries between different textures utilizing a marching cubes framework. Triangular polygons form the surfaces which represent such textural boundaries. The uniqueness of this approach lies in two areas. First, we use 3D texture features that are extensions of common 2D features used by researchers to analyze texture. These texture features are extracted from defined local $N \times N \times N$ volumes, and they allow for the characterization and differentiation of various texture types. We do not assume, in the overall approach, that the textures follow any one specific model.

Second, we attempt to directly segment volume data in a computationally rapid and practical way rather than segment the images on a 2D section by section basis. This allows for direct utilization of all of the information contained in the 3D data. Furthermore, the processing of volume data may be computationally very demanding. Also, any unsupervised segmentation method can require extraordinary computation time especially when estimation and segmentation are attempted simultaneously, which is essentially our procedure. The marching cubes framework provides a potential option for a practical and relatively rapid approach to segmentation. The normalized Euclidean distance similarity measure between feature vectors (which represent the textures in overlapping local volumes) drives the segmentation.

Experimental results have been at least encouraging. They suggest that this approach to surface-based segmentation of volume data may lead to better and more effective methods. The segmentation algorithm was able to construct approximate surfaces representing boundaries between volumes of different texture. Results varied from poor to good in a qualitative sense.

There are several possible avenues for future investigation:

1. Different methods and measures of comparing volumes of texture may need further exploration. The normalized Euclidean distance measure was used because the current algorithm processes one cube at a time and stores only the feature vectors of the eight vertices of the current cube. If the feature vectors at neighboring locations of the current cube's vertices in the 3D lattice are stored, perhaps some measure such as a Mahalanobis distance may be used (see Mui, 1995). However, the computation of texture features for a feature vector requires time. Storing multiple feature vectors will mean computing more texture features which will increase the computational load significantly.

2. Different local volumes may be defined and used. The local volumes implemented here are convenient in that the local volume for a specific cube vertex is the same regardless with which neighboring cube vertex it is compared in determining surface-edge intersections. However, there may be better ways of defining local volumes.

3. Some method of automatic threshold selection may be implemented. Currently, texture thresholds are determined for the similarity measure in a trial and error fashion which is tedious. One idea is to use some modified approach to the adaptive region growing method developed by Chang and Li (1994). Their algorithm uses both position- and time-varied

thresholds that are dynamically and automatically computed in a region growing process. They compare regions constructed of subregions with other regions by computing features in the subregions and specifying the allowance for which feature values within the regions may be allowed to deviate. For two regions to be considered homogeneous, they required that each region's feature mean fall within what they define as the other region's adaptive range. These techniques may be adapted to compare volumes. It should be noted that here we have primarily dealt with images with only two different volumes of texture. MRI and CT images will contain multiple types of texture. Some adaptive thresholding process like the idea just described may be incorporated in order to process images with more than two types of texture.

4. Solutions to the polygon connectivity problem or the multi-layer problem may be devised. Since the two problems are inter-related, the ideas presented here might help either one. Ideally, if a 3D image with two different textures is input into the algorithm, one smooth, well-connected surface between the textures is the desired output. This is the area of greatest interest for future investigations.

One possible solution is to assign texture gradients to the created triangles and use a measure of the texture gradients to filter out only the polygonal triangles of the surface of interest in the image. A texture gradient may be computed for a triangle by computing the texture gradient for a cube vertex and then interpolating the gradient to the location of one of the triangle vertices. If we assume that each computed polygonal triangle is flat, the texture gradient for one triangle's vertex should be sufficient to represent the texture gradient for the triangle as a whole.

Another idea is to modify the way the volume data is processed. The current algorithm divides the data space into cubes and determines surface cuts through each cube while

processing the data in scan-line order. Each point of the 3D lattice represents a cube vertex, but it is possible to decrease the resolution and allow for $N \times N \times N$ sized basic cube elements where N represents a cube edge length and where the edge length between two points in the 3D lattice of the volume data is defined as one. We then determine surface cuts through these $N \times N \times N$ cubes instead of the $1 \times 1 \times 1$ cubes currently used.

Doing this may help resolve the multi-layer problem but at the expense of not being able to segment objects of smaller resolution than the cube size used. We present a specific case that will help to explain the idea. Assume that we are processing $5 \times 5 \times 5$ cubes. First, we can determine surface-edge intersections of each $5 \times 5 \times 5$ cube by using local $4 \times 4 \times 4$ volumes that are located entirely within each cube (remember that the local volume dimensions $4 \times 4 \times 4$ refer to voxel numbers). Second, this means that there will be less overlap between local volumes associated with each cube used to compute texture features. This in turn may help reduce the redundant placement of polygons which is the root of the multi-layer problem.

One last suggestion is that contextual information, or information on spatially adjacent objects or regions, may help improve the determination of accurate surface-edge intersections and may help maintain connectivity between surface polygons. Label-based segmentation methods on 2D images that use contextual information have given better results than those methods that neglect context, especially for those segmentation methods interested in finding accurate boundaries between regions of texture. Contextual information has been introduced into label-based segmentation algorithms by *ad hoc* methods combined with clustering (see Hu and Dennis, 1994), stochastic relaxation methods that determine the MAP estimate of an image model which is a joint probability distribution of pixel gray levels and pixel labels (see Geman *et al.*, 1990 and Graffigne, 1987), and probabilistic relaxation methods which enforce

spatial constraints by iteratively updating pixel labels to ensure local consistency (see Hsiao and Sawchuk, 1989).

It may be possible to garner ideas from such work and modify them for an improved surface-based segmentation method. More specifically, it seems that incorporating contextual information (in some manner in our segmentation algorithm) holds great potential for ensuring surface polygon connectivity. Contextual information could help in accurately deciding where surface-edge intersections are located, thereby forming the foundations for a set of well-connected polygons. This might mean a combined boundary-based and region-based segmentation method. The more difficult segmentation problems usually necessitate a well designed combination of boundary-based and region-based approaches in one segmentation system [Grinaker, 1980]. As hinted at earlier, the polygon connectivity problem should form the core of all future investigations.

The results in this thesis should provide a basis for such future work and may motivate the development of algorithms that will some day be able to construct accurate and well-connected polygonal surfaces between different volumes of texture in 3D images.

Bibliography

T. Aach and H. Dawid. Region Oriented 3D-Segmentation of NMR-Datasets: A Statistical Model-Based Approach. *SPIE Vol. 1360 Visual Communications and Image Processing*, (1990), pp. 690-701.

J. Besag. Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society, Series B*, Vol. B-36, (1974), pp. 192-236.

J. Besag. On the Statistical Analysis of Dirty Pictures. *J.R. Statist. Soc. B*, Vol. 48, No. 3, (1986), pp. 259-302.

J.C. Bezdek, L.O. Hall, and L.P. Clarke. Review of MR Image Segmentation Techniques Using Pattern Recognition. *Med. Phys.*, 20(4), (July/Aug 1993), pp. 1033-1048.

M. Bomans, K. Hohne, U. Tiede, and M. Riemer. 3-D Segmentation of MR Images of the Head for 3-D Display. *IEEE Transactions on Medical Imaging*, Vol. 9, No. 2, (June 1990), pp. 177-183.

J.D. Cappelletti and A. Rosenfeld. Three-Dimensional Boundary Following. *Computer Vision, Graphics, and Image Processing*, 48, (1989), pp. 80-92.

J.M. Carstensen. Description and Simulation of Visual Texture. Ph.D. thesis, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, (April 1992).

A. Chakraborty, L. H. Staib, and J. S. Duncan. An Integrated Approach to Boundary Finding in Medical Images. *Proceedings of the IEEE Workshop on Biomedical Image Analysis*, Seattle, Washington, (June 24-25, 1994), pp. 13-22.

T. Chang and C.C. Jay Kuo. Texture Analysis and Classification with Tree-Structured Wavelet Transform. *IEEE Transactions on Image Processing*, Vol. 2, No. 4, (1993), pp. 429-441.

Y.-L. Chang and X. Li. Adaptive Image Region-Growing. *IEEE Transactions on Image Processing*, Vol. 3, No. 6, (November 1994), pp. 868-872.

R. Chellappa and S. Chatterjee. Classification of Textures Using Gaussian Markov Random Fields. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-33, No. 4, (August 1985), pp. 959-963.

R. Chellappa and R.L. Kashyap. Synthetic Generation and Estimation in Random Field Models of Images. *Proc. IEEE Comput. Soc. Conf. Pattern Recog. Image Proc.*, Dallas, TX, (August 1981), pp. 577-582.

H.E. Cline, W.E. Lorensen, R. Kikinis, and F. Jolesz. Three-Dimensional Segmentation of MR Images of the Head Using Probability and Connectivity. *Journal of Computer Assisted Tomography*, Vol. 14, No. 6, (November/December 1990), pp. 1037-1045.

F.S. Cohen and Z. Fan. Maximum Likelihood Unsupervised Textured Image Segmentation. *CVGIP: Graphical Models and Image Processing*, Vol. 54, No. 3, (May 1992), pp. 239-251.

I. Cohen, L.D. Cohen, and N. Ayache. Using Deformable Surfaces to Segment 3-D Images and Infer Differential Structures. *Computer Vision - ECCV '92, Second European Conference on Computer Vision Proceedings*, pp. 242-263.

G.R. Cross and A.K. Jain. Markov Random Field Texture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-5, No. 1, (January 1983), pp. 25-39.

F. D'Astous and M.E. Jernigan. Texture Discrimination Based on Detailed Measures of the Power Spectrum. *7th International Conference on Pattern Recognition*, Montreal, Canada, Vol. 1, (July 30-August 2, 1984), pp. 83-86.

R.A. Drebin, L. Carpenter, P. Hanrahan. Volume Rendering. *Computer Graphics*, Vol. 22, No. 4, (1988), pp. 65-74.

H.H. Ehrlicke. Problems and Approaches for Tissue Segmentation in 3D-MR Imaging. *SPIE Medical Imaging IV: Image Processing*, Vol. 1233, (1990), pp. 128-137.

D. Geman and S. Geman. Relaxation and Annealing with Constraints. *Division Appl. Math., Brown Univ., Complex Systems Tech. Rep. 35*, (1987).

S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 6, (Nov. 1984), pp. 721-741.

D. Geman, S. Geman, C. Graffigne, and P. Dong. Boundary Detection by Constrained Optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 7, (July 1990), pp. 609-628.

G. Gerig, J. Martin, R. Kikinis, O. Kubler, M. Shenton, and F. A. Jolesz. Unsupervised Tissue Type Segmentation of 3D Dual-Echo MR Head Data. *Image and Visual Computing*, Vol. 10, No. 6, (July/August 1992), pp. 349-360.

R.C. Gonzalez and P. Wintz. *Digital Image Processing*. Addison-Wesley Publishing Co., (1987).

C. Graffigne. Experiments in Texture Analysis and Segmentation. Ph.D. dissertation, Division Appli. Math., Brown Univ., (1987).

W.E.L. Grimson, G.J. Ettinger, T. Kapur, M.E. Leventon, W.M. Wells III, and R. Kikinis. Utilizing Segmented MRI Data in Image-Guided Surgery. (submitted for publication) Available at web-site: http://www.ai.mit.edu/projects/vision-surgery/surgery_home_page.html. (July 1996).

W.E.L. Grimson and T. Pavlidis. Discontinuity Detection For Visual Surface Reconstruction. *Comput. Vision, Graphics, Image Processing*, Vol. 30, (1985), pp. 316-330.

S. Grinaker. Edge Based Segmentation and Texture Separation. *IEEE Proc. 5th International Conf. Pattern Recog.*, Miami, Florida, (Dec. 1980), pp. 554-557.

R.M. Haralick. Statistical and Structural Approaches to Texture. *Proceedings of the IEEE*, Vol. 67, No. 5, (May 1979), pp. 786-803.

R.M. Haralick, K. Shanmugam, and I. Dinstein. Textural Features for Image Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-3, No. 6, (November 1973), pp. 610-621.

M. Hassner and J. Sklansky. The Use of Markov Random Fields as Models of Texture. *Computer Graphics and Image Processing*, 12, (1980), pp. 357-370.

J.Y. Hsiao and A.A. Sawchuk. Unsupervised Textured Image Segmentation Using Feature Smoothing and Probabilistic Relaxation Techniques. *Computer Vision, Graphics, and Image Processing*, 48, (1989), pp. 1-21.

Y. Hu and T.J. Dennis. Textured Image Segmentation by Context Enhanced Clustering. *IEEE Proc., Vision, Image, and Signal Processing*, 141(6), (December 1994), pp. 413-421.

IEEE Standard 610.4-1990, *IEEE Standard Glossary of Image Processing and Pattern Recognition Terminology*, IEEE Press, New York, 1990.

M.E. Jernigan and F. D'Astous. Entropy-Based Texture Analysis in the Spatial Frequency Domain. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-6, No. 2, (March 1984), pp. 237-243.

M. Joliot and B. M. Mazoyer. Three-Dimensional Segmentation and Interpolation of Magnetic Resonance Brain Images. *IEEE Transactions on Medical Imaging*, Vol. 12, No. 2, (June 1993), pp. 269-277.

T. Kapur. Segmentation of Brain Tissue from Magnetic Resonance Images. *MIT Artificial Intelligence Laboratory Technical Report 1566*, (January 1995).

R.L. Kashyap and R. Chellappa. Estimation and Choice of Neighbors in Spatial-Interaction Models of Images. *IEEE Transactions on Information Theory*, Vol. IT-29, No. 1, (January 1983), pp. 60-72.

R.L. Kashyap, R. Chellappa, and N. Ahuja. Decision Rules for Choice of Neighbors in Random Field Models of Images. *Computer Graphics and Image Processing*, 15, (1981), pp. 301-318.

R.L. Kashyap, R. Chellappa, and A. Khotanzad. Texture Classification Using Features Derived From Random Field Models. *Pattern Recognition Letters*, Vol. 1, No. 1, (October 1982), pp. 43-50.

R.L. Kashyap and A. Khotanzad. A Model-Based Method for Rotation Invariant Texture Classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-8, No. 4, (July 1986), pp. 472-481.

D.R. Ketten. The Marine Mammal Ear: Specializations for Aquatic Audition and Echolocation in: *The Evolutionary Biology of Hearing*, D. Webster, R. Fay, and A. Popper, eds. New York: Springer-Verlag, (1992), pp. 717-750.

D.R. Ketten. Functional Analyses of Whale Ears: Adaptations for Underwater Hearing. *IEEE Proceedings in Underwater Acoustics*, Vol. 1, (1994), pp. 264-270.

D. Ketten and D. Wartzok. Three-Dimensional Reconstructions of the Dolphin Ear in: *Sensory Abilities of Cetaceans*, J. Thomas and R. Kastelein, eds. New York: Plenum Press, (1990), pp. 81-105.

J. Kittler and J. Illingworth. Relaxation Labelling Algorithms-- A Review. *Image and Vision Computing*, Vol. 3, No. 4, (Nov 1985), pp. 206-216.

H.A. Koenig and G. Laub. Tissue Discrimination in Magnetic Resonance 3D Data Sets. *SPIE Vol. 914 Medical Imaging II*, (1988), pp. 669-672.

J. Krumm and S.A. Shafer. Local Spatial Frequency Analysis of Image Texture. *IEEE Third International Conf. Computer Vision*, (December 1990), pp. 354-358.

J. Krumm and S.A. Shafer. Segmenting Textured 3D Surfaces Using the Space/Frequency Representation. *Spatial Vision*, Vol. 8, No. 2, (1994), pp. 281-308.

Z. Liang, J.R. MacFall, and D.P. Harrington. Parameter Estimation and Tissue Segmentation from Multispectral MR Images. *IEEE Transactions on Medical Imaging*, Vol. 13, No. 3, (Sept. 1994), pp. 441-449.

S. Liou and R.C. Jain. An Approach to Three-Dimensional Image Segmentation. *CVGIP: Image Understanding*, Vol. 53, No. 3, (May 1991), pp. 237-252.

S. Liu and M.E. Jernigan. Texture Analysis and Discrimination in Additive Noise. *Computer Vision, Graphics, and Image Processing*, 49, (January 1990), pp. 52-67.

G. Lohmann. Analysis and Synthesis of Textures: A Co-Occurrence-Based Approach. *Comput. and Graphics*, Vol. 19, No. 1, (1995), pp. 29-36.

W.E. Lorensen and H.E. Cline. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. *Computer Graphics*, Volume 21, Number 4, (July 1987), pp. 163-169.

B.S. Manjunath and R. Chellappa. Unsupervised Texture Segmentation Using Markov Random Field Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 13, No. 5, (May 1991), pp. 478-482.

J. Mao and A.K. Jain. Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models. *Pattern Recognition*, Vol. 25, No. 2, (1992), pp. 173-188.

O. Monga, R. Deriche, G. Malandain, and J.P. Cocquerez. Recursive Filtering and Edge Tracking: Two Primary Tools for 3D Edge Detection. *Image and Vision Computing*, Vol. 9, No. 4, (August 1991), pp. 203-214.

L. Mui. A Statistical Multi-Experts Approach to Image Classification and Segmentation. M.Eng. thesis, Massachusetts Institute of Technology, (August 1995).

H. Muller and M. Stark. Adaptive Generation of Surfaces in Volume Data. *The Visual Computer*, 9, (1993), pp. 182-199.

R.E. Muzzolini, Y. Yang, and R. Pierson. Three Dimensional Segmentation of Volume Data. *Proceedings ICIP-94*, Vol. 3, Austin, Texas, (Nov. 13-16, 1994), pp. 488-492.

R. Muzzolini, Y. Yang, and R. Pierson. Texture Characterization Using Robust Statistics. *Pattern Recognition*, Vol. 27, No. 1, (1994), pp. 119-134.

K.M. Oh and K.H. Park. A Vertex Merging Algorithm for Extracting a Variable-Resolution Isosurface from Volume Data. *IEEE International Conference on Systems, Man, and Cybernetics*, Vancouver, British Columbia, Canada, Vol. 4, (Oct. 22-25, 1995), pp. 3543-3548.

P.P. Ohanian and R.C. Dubes. Performance Evaluation for Four Classes of Textural Features. *Pattern Recognition*, Vol. 25, No. 8, (1992), pp. 819-833.

T. Peters, B. Davey, P. Munger, R. Comeau, A. Evans, and A. Olivier. Three-Dimensional Multimodal Image-Guidance for Neurosurgery. *IEEE Transactions on Medical Imaging*, Vol. 15, No. 2, (April 1996), pp. 121-128.

R. Porter and N. Canagarajah. A Robust Automatic Clustering Scheme For Image Segmentation Using Wavelets. *IEEE Transactions on Image Processing*, Vol. 5, No. 4, (April 1996), pp. 662-665.

W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. Numerical Recipes in C: The Art of Scientific Computing. 2nd edition, New York: Cambridge University Press, (1992).

T.R. Reed and J.M. Hans Du Buf. A Review of Recent Texture Segmentation and Feature Extraction Techniques. *CVGIP: Image Understanding*, Vol. 57, No. 3, (May 1993), pp. 359-372.

A. Rosenfeld, R.A. Hummel, and S.W. Zucker. Scene Labeling by Relaxation Operations. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6, No. 6, (June 1976), pp. 420-433.

W.J. Schroeder, J.A. Zarge, and W.E. Lorensen. Decimation of Triangle Meshes. *Computer Graphics*, 26, 2, (July 1992), pp. 65-70.

SegmentVIEW Version 2.1 User's Guide and Reference Manual. TechnoData Software, Inc. (1996).

R. Shu, C. Zhou, and M.S. Kankanhalli. Adaptive Marching Cubes. *The Visual Computer*, Vol. 11, (1995), pp. 202-217.

K.C. Strasters and J.J. Gerbrands. Three-dimensional Image Segmentation Using a Split, Merge, and Group Approach. *Pattern Recognition Letters 12*, (May 1991), pp. 307-325.

J.K. Udupa, S.N. Srihari, and G.T. Herman. Boundary Detection in Multidimensions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-4, No. 1, (January 1982), pp. 41-50.

M. Unser. Texture Classification and Segmentation Using Wavelet Frames. *IEEE Transactions on Image Processing*, Vol. 4, No. 11, (1995), pp. 1549-1560.

H. Wechsler. Taxonomy and Segmentation of Textured Images. *Proceedings: 5th International Conference on Pattern Recognition*, (1980), pp. 532-534.

D. Wermser. Unsupervised Segmentation by Use of a Texture Gradient. *7th International Conference on Pattern Recognition*, Montreal, Canada, Vol. 2, (July 30-August 2, 1984), pp. 1114-1116.

J.S. Weszka, C.R. Dyer, and A. Rosenfeld. A Comparative Study of Texture Measures for Terrain Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-6, No. 4, (April 1976), pp. 269-285.

A.S. Willsky, G.W. Wornell, and J.H. Shapiro. Stochastic Processes, Detection and Estimation. 6.432 Supplementary Course Notes, MIT, 1995.

J.W. Woods. Two-Dimensional Discrete Markovian Fields. *IEEE Transactions on Information Theory*, Vol. IT-18, No. 2, (March 1972), pp. 232-240.

S.W. Zucker and R.A. Hummel. A Three-Dimensional Edge Operator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-3, No. 3, (May 1981), pp. 324-331.