Material Classification of Magnetic Resonance Volume Data

Thesis by

David H. Laidlaw

In Partial Fulfillment of the Requirements
for the Degree of
Master of Science

California Institute of Technology
Pasadena, California

1992

(Submitted May 29, 1992)

Abstract

A major unsolved problem in computer graphics is that of making high-quality models. Traditionally, models have consisted of interactively or algorithmically described collections of graphics primitives such as polygons. The process of constructing these models is painstaking and often misses features and behavior that we wish to model. Models extracted from volume data collected from real, physical objects have the potential to show features and behavior that are difficult to capture using these traditional modeling methods.

We use vector-valued magnetic resonance volume data in this thesis. The process of extracting models from such data involves four main steps: collecting the sampled volume data; preprocessing it to reduce artifacts from the collection process; classifying materials within the data; and creating either a rigid geometric model that is static, or a flexible, dynamic model that can be simulated. In this thesis we focus on the the first three steps.

We present guidelines and techniques for collecting and processing magnetic resonance data to meet the needs of the later steps. Our material classification and model extraction techniques work better when the data values for a given material are constant throughout the dataset, when data values for different materials are different, and when the dataset is free of aliasing artifacts and noise.

We present a new material-classification method that operates on vector-valued volume data. The method produces a continuous probability function for each material over the volume of the dataset, and requires no expert interaction to teach it different material classes. It operates by fitting peaks in the histogram of a collected dataset using parameterized gaussian bumps, and by using Bayes' law to calculate material probabilities, with each gaussian bump representing one material.

To illustrate the classification method, we apply it to real magnetic resonance data of a human head, a human hand, a banana, and a jade plant. From the classified data, we produce "computationally stained" slices that discriminate among materials better than do the original grey-scale versions. We also generate volume-rendered images of classified datasets clearly showing different anatomical features of various materials. Finally, we extract preliminary static and dynamic geometric models of different tissues.

Acknowledgments

I would like to thank Al Barr, my advisor, for the many new directions and fresh ideas that he suggested, for teaching me applied math, among other things, and for providing the resources and creating the environment that made this work possible. I would also like to thank my fellow graduate students, Ronen Barzel, Bena Currin, Kurt Fleischer, Jeff Goldsmith, Devendra Kalra, Dave Kirk, John Snyder, and Adam Woodbury, for lots of productive suggestions and help with this work, and for the sense of camaraderie that made working in the lab a lot more fun and stimulating. In particular, John and Adam helped extensively with model extraction, Dave and Devendra started the process of collecting and working with MR data, and Ronen and Kurt, among others, provided me with lots of insight during many long, involved conversations. I am also grateful to Matthew Avalos for implementing much of the volume-rendering software and for producing many of the volume-rendered images in Chapter 5.

I would also like to thank Donald Marks, Bruce Bell, Mike Meckler, Mark Montague, Pete Wenzel and Allen Corcoran for helping to keep the lab software and hardware running, and Dian De Sha and Sandra Reyna for keeping the lab functioning otherwise. Mark and Dian also made helpful comments on earlier versions of this work.

The data used in this work was collected at the Huntington Magnetic Resonance Center in Pasadena. I am grateful to Dr. Brian Ross for his help in organizing the process, to Bassem for many instructive conversations, and to Jose Jiminez for the late-night collection sessions.

Thanks also to John Allman and Russ Jacobs for help in learning about magnetic resonance, biology, and anatomy.

This work was supported by grants from the Beckman Institute Foundation and the National Science Foundation Science and Technology Center for Computer Graphics and Visualization.

Finally, I would like to thank my parents for their long-distance support and encouragement, and my wife, Barbara Meier, not only for her technical and moral support, but for her patience and understanding.

Contents

Chapter 1

Introduction

1.1 Goals and Motivation

The primary motivation for the work in this thesis is to develop methods and techniques for creating high-quality computer graphics models of real-world objects. Our goal is to create flexible models of plants and animals whose multiple parts can interact and whose behavior can be simulated. We would like the models to have flexible parts that slide across one another without penetrating, that can exert a pull on one another through some region of connection, and that can push and deform one another as they are moved and simulated. Examples of such parts include bones, muscles, subcutaneous fat, and skin. We present several steps towards achieving that goal, although the models we have created for this thesis are preliminary.

Traditionally, computer graphics models have been created using programs or lists of graphics commands. The models have consisted of collections of connected rigid objects or of kinematically-deformed surfaces [Foley et al. 90]. More recently research has modeled fairly simple physically-based flexible objects [Terzopoulos et al. 87] [Platt and Barr 88] [Terzopoulos and Fleischer 88]. In addition to having fairly simple geometries, the flexible objects modeled generally consist of a single "part" made of a single, homogeneous "material." Examples of a single-part object include a trampoline, a cube of jello, and an eggplant. The material of such a part is modeled as a flexible object, and can interact with rigid models, such as a table top or ball, but cannot interact with other flexible models.

While multiple parts and some of the desired interactions have been successfully modeled kinematically, as can be seen in many computer graphics animations [Siggraph Films 87] [Siggraph Films 88] [Siggraph Films 89], the resulting models are often ad-hoc and lack important features of the real objects they are modeling. Using the kinematic techniques for this purpose is also very painstaking and time consuming.

We propose creating models from measurements of real-world objects. Such models have the potential to capture features and behavior that otherwise might be inaccurate or even absent. Measurements that provide information about the inside of an object, not just about the surface, are useful for modeling the parts of an object that are not visible from the the outside, such as the bones and muscles in a hand.

There are several different measurement methodologies that produce data from which we could extract models. Two-dimensional techniques include monocular camera images, stereo images, and range data. True three-dimensional techniques include computed tomography (CT), magnetic resonance (MR), optical sectioning, and physical sectioning.

From this collection of possibilities, we have chosen to use magnetic resonance data. MR

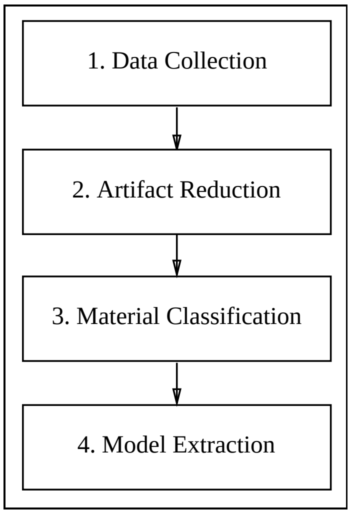

Figure 1.1: Four steps in creating computer graphics models from volume data. In this thesis we focus on the first three steps of this process. □

data provides volume measurements; these measurements give information not only about external surfaces of an object, but also about materials and properties inside the object, unlike camera images and range data. Compared to physical sectioning, MR is non-invasive. MR is more acceptable for human experimentation than CT, because MR does not significantly damage living tissue. MR can image objects opaque to light, unlike optical sectioning. MR distinguishes soft tissues well, measures more than one physical characteristic of an object, provides reasonably high-resolution volume images, and is readily available.

Chapter 2 describes the MR data-collection process, and Chapters 2 and 3 address some of its limitations and how to minimize them.

1.2 Overview

Our goal is to create models that have the behavior of flexible, physical objects, with parts that interact with each other and with their environment. In Fig. 1.1, we identify four basic steps involved in extracting models from volume data. In this thesis, we focus on the first three steps of the process, which are prerequisites of the final model-extraction step.

1. **Data Collection** involves choosing objects to model and scanning them to obtain vector-valued, three-dimensional volume data. Choosing the objects can be surprisingly difficult because of the constraints imposed by the imaging technology and because of the level of detail we desire in our models. The object must be small enough to fit in the machine, and yet must not have detail too small to identify in the resulting volume data. It must also contain materials that can be distinguished using MR. Selecting appropriate parameters to differentiate selected tissues and sub-parts, to minimize artifacts, and to work well with the subsequent steps of the model extraction procedure is also a highly-constrained problem. In Chapter 2 we will discuss these problems, and our experience with collecting data from which we can extract models.

2. **Artifact Reduction** focuses on reducing problems due to aliasing, mis-alignment of different volumes collected from the same object, non-uniform response of a single type of material

within a volume, and poor contrast between different materials. These processes are discussed in Chapter 3.

3. **Material Classification** is the third step of the process. In this step we define a continuous "classification function" over the imaged volume, rather than picking a discrete material for each sample. The value of this function is the probability density of the input point being a particular material. To calculate these probability densities, we create a histogram of the collected data and fit a sum of gaussian functions to the histogram. We then use each gaussian to represent the distribution of data values for one material and use Bayes' Law to infer material probabilities. The classification algorithm is described in Chapter 4.

4. **Model Extraction** converts the highly detailed but low-level data into more abstracted geometric or physically-based models that have enhanced behavioral and movement properties. Our preliminary work on extracting static and dynamic models from classified MR datasets is included in Chapter 5, "Applications to Visualization".

Chapter 5 also illustrates the techniques and their applications with *computationally stained* two-dimensional slices, and volume-rendered images of the classified data. As examples, we will show images created from processed data of human heads, a human hand, bananas, and a jade plant.

1.3 Related Work

This work extends and combines work from medical magnetic resonance imaging, pattern classification, image processing, and computer graphics.

The MR literature is extensive [Wehrli 88] and describes many techniques for collecting volume datasets of objects. We are primarily collecting data using a technique called spin-echo imaging, which we describe in Section 2.2 [Keller 88]. We present guidelines for selecting objects and collecting data that can be used effectively for creating computer graphics models.

The signal-processing and computer graphics literature discusses many techniques for processing data to avoid and reduce artifacts [Oppenheim 83] [Lim 90]. MR machines use these techniques to produce images appropriate to the medical community, which is their primary focus. Our goal of extracting models from the data, rather than examining slices of the data, places different requirements on the sampling process. We need to correct artifacts that impede our classification and model-extraction techniques. For example, the use of a Hamming filter, as described in Chapter 2, to low-pass filter the discretely sampled data to reduce aliasing is a known process. We believe that within the computer graphics community the process is not applied often enough to eliminate the artifacts evident in many static geometric models derived from volume data.

[Duda and Hart 73] describes matching parametric models of distributions to sets of samples by applying statistical techniques to calculate the parameters directly. Finding the mean and variance of a set of real numbers is a simple example of such a process. We, instead, fit similar distribution models to histograms of samples. We attempt to avoid local minima and to determine the number of different materials by identifying materials in the histogram one at a time, finding the most prevalent materials first, and continuing until there are no significant histogram peaks that correspond to unidentified materials.

Material-classification techniques in computer graphics and MR are becoming more common [Drebin et al. 88], [Cline 90], [Kikinis et al. 90]. The technique in [Drebin et al. 88] involves interactively classifying the histogram of a scalar-valued dataset. The other two references use interactive, supervised techniques: a person trained in anatomy determines characteristic material responses, and then the classification algorithm labels the rest of a volume dataset using that information. [Yoo et al. 92] and [Pizer 90] present another technique that uses unsupervised methods to divide a dataset into multiple regions, which are then interactively placed into different classes. Our technique differs from all of these supervised ones because it does not require a trained operator to distinguish the different materials and regions.

[Vannier et al. 85] and [Vannier et al. 88] use techniques originally designed and written to process satellite images to identify different materials in data. Some of these techniques are unsupervised, but they all identify each voxel as a single material. We produce a continuous function of the volume, where each sample can consist of a mixture several materials. This function can be evaluated anywhere within the volume. This continuous classification avoids some artifacts, like aliasing and discontinuities, that discrete classification can cause in extracted models.

[Levoy 88], [Drebin et al. 88], and [Upson and Keeler 88] describe volume rendering techniques. Some earlier work is described in [Kajiya and von Herzen 84]. Our contribution to these standard volume-rendering techniques is our method for classifying the volume data and mapping it to properties that the volume renderer can use to create images.

Related work has utilized 2-D image data in making computer graphics models [Muraki 91], [Terzopoulos and Fleischer 88]. Our work differs because we are working with 3-D volume data. [Lorensen and Cline 87], [Snyder 92], and [Miller et al. 91] all extract surface models from volume data. Our preliminary techniques are similar to those of [Lorensen and Cline 87] and [Snyder 92], but we extend them to create dynamic models by including the "insides" of the objects as well as the surfaces.

1.4 Thesis Organization

Chapter 2, "Data Collection," discusses collecting MR imaging data. Chapter 3, "Artifact Reduction," presents the different artifacts for which we can compensate, together with the details of the compensation techniques. Chapter 4, "Material Classification," describes the creation of multi-dimensional histograms of volume data, fitting each with a sum of parameterized gaussian functions, and using the results to classify the volume data. Chapter 5, "Applications to Visualization," illustrates our techniques with computationally-stained images from the collected and classified volume data, with volume-rendered images of the data, and with rudimentary geometric and dynamic models that are extracted from the data. Results and future work are summarized in Chapter 6.

Chapter 2

Data Collection

Our goal of automatically extracting geometric and dynamic models from three-dimensional MR data leads to a number of goals for the data collection process. As explained in Section 2.2, MR data is provided as discrete, vector-valued samples of a function of three-dimensional space. We need the following properties in these sampled MR datasets:

- sufficient resolution to distinguish the features that we are interested in,
- different values for regions of an image corresponding to different materials in an object,
- sufficiently small aliasing artifacts,
- constant values in an image for regions of constant material in an object, modulo noise, and
- geometric alignment of different scalar volume images of the same object, so that they can be combined to form a vector-valued image.

This chapter will describe some background on sampling and its terminology, present some basic information about how MR works, and briefly discuss parameters that control MR data collection. We will also present some guidelines we have developed for choosing objects and parameters, and describe some examples of the data that we have collected.

2.1 Sampling Terminology

In this section we review some terminology from digital signal processing and introduce related notation that we will use. Appendix A is a more detailed summary than this review for readers less familiar with the field. Sources such as [Oppenheim 83] or [Lim 90] provide an even more in-depth treatment, while [Blinn 89a] and [Blinn 89b] give a more intuitive description. Terms in this summary will be *italicized* at their first use.

For our purposes, digital signal processing consists of mapping from functions of continuous parameters to *sampled* functions and back again, as well as operating on those functions or the samples that represent them.

Functions are sampled at regular intervals using a particular *kernel function* to produce a set of samples called a *dataset*. Each sample is a *convolution* of the original function with a translated version of the kernel function. From that dataset we can *reconstruct* a continuous approximation to the original function using another kernel function.

The *fourier transform* of a function is its representation in *fourier-* or *frequency-space*. The fourier transform of a kernel function is its *frequency response*. *Filtering* a function involves convolving it with a kernel function (or *filter kernel*) at each point, and has the effect of scaling the frequency-space representation of the input function by the frequency response of the kernel function. Filtering a function with a filter kernel that attenuates high frequencies and passes low frequencies relatively unchanged is called *low-pass filtering*.

A dataset can only represent a function containing frequency components less than the *Nyquist limit* for the sampling rate of the dataset. If higher frequencies are not removed by the sampling kernel function, they show up as *aliasing* artifacts in the dataset. *Gibb's phenomenon* can also cause *ringing* artifacts in a dataset if the frequency cutoff of the sampling kernel function is too abrupt.

We discuss and use several specific kernel functions, including the Dirac δ-function, the *sinc* function, a *box* function, a *triangle* function, a *cubic b-spline* approximation to a gaussian, and a *Hamming* function for sampling, filtering, and reconstructing functions. See Appendix A for as equations and figures for these functions, as well as a discussion of their performance and frequency-response trade-offs.

Different MR collection techniques use different sampling kernels. We will discuss these and the collection techniques in Section 2.2 and in the aliasing subsection of Chapter 3.

2.2 Introduction to Magnetic Resonance

In order to create models, we need appropriate collected data. We describe the MR collection process in this section to make clear the origin of some of the artifacts we remove and to explain the basis for some of the collection guidelines we present at the end of this chapter. In this section, we describe the characteristic material values that MR measures, discuss the sampling that an MR machine does, and explain some of the parameters that control the data-collection process.

2.2.1 MR Terminology and Definitions

A physical material such as skin, muscle, or fat can be modeled as having three MR-measurable values related to the atomic nuclei of some specific element. The first is *net spin density*, and the other two are magnetic decay time constants characteristic for each material. These values are often referred to as $\rho(x), T_1(x)$ and $T_2(x)$, and are functions of 3-D space. We will describe these values briefly. (See [Keller 88] [Wehrli 88] and [Wehrli] for more details.)

An object in an MR machine is subjected to a strong, constant magnetic field. Because atomic nuclei are charged and are spinning, their axes tend to line up with the constant field of the machine. They can line up spinning in either direction, but tend to favor one direction over the other. Net spin density, sometimes referred to as $\rho(x)$, measures this difference between the density of nuclei spinning one direction and of those spinning the other direction.

Nuclei absorb radio-frequency (RF) energy at certain wavelengths related to the strength of the magnetic field surrounding them, and are thus perturbed from their aligned position. They then relax back into their aligned positions, and release that RF energy, with an exponential time constant of $T_1(x)$.

Although nuclei tend to line up with the constant field, the alignment is not exact. As a result, their rotational axes precess around an axis in the direction of the constant field. Because materials can affect the magnetic field locally, the precession can happen at slightly different rates, and spins that are initially all in phase get out of phase over time. The exponential time constant for this dephasing is $T_2(x)$.

Typically, we measure these properties for hydrogen nuclei, or protons, because hydrogen is so prevalent in biological tissue and because hydrogen nuclei produce a very strong MR signal relative to other nuclei. Other nuclei, however, can produce a stronger signal than hydrogen nuclei within materials that have little or no hydrogen.

The RF energy that is transmitted to and received from the nuclei is both produced and measured by *coils*. In many cases, a single coil does both jobs. If the coil can deliver the same amount of RF energy to every part of the volume being imaged, and can receive equally well from every part, then the resulting images will show uniform materials with uniform intensities. Transmission and reception efficiency are generally complicated functions of space, although there are often regions of relative uniformity. For cylindrical coils, in particular, the region near the center of the coil is relatively uniform, while near the ends of the cylinders the response drops off quite a bit. For some other coils, unfortunately, there is no uniform region.

2.2.2 Spin-echo Data Collection

MR data is produced as vector-valued samples of a function of 3-D space. We primarily use a collection protocol called *spin-echo* imaging. Spin-echo imaging it is less sensitive to local inhomogeneities in the constant magnetic field than other imaging techniques, and tends to avoid some geometric and intensity distortions that those inhomogeneities can cause. We have also collected one dataset using a gradient-echo technique, which provides thinner slices, but which is more susceptible to geometric distortions.

Spin-echo values, $v(x)$, of samples in datasets collected with the spin-echo technique can be modeled by the following equation:

$$v(x) = \rho(x)(1 - e^{-T_R/T_1(x)})e^{-T_E/T_2(x)}, \tag{2.1}$$

where T_R, the *relaxation time*, and T_E, the *echo time* are parameters that can be set for each collection to vary the function of ρ, T_1, and T_2 that is collected. The collection sequence excites the object with RF energy and measures the response from each excitation. The sequence pauses for the relaxation time between excitations, and measures the response after each excitation at the echo time. Varying these two parameters gives much control over the image values that different materials give.

There are many ways to set T_R and T_E to get different spin-echo values, $v(x)$, for the same material. By varying these parameters we can get a spin-echo value for one material that is large for one (T_R, T_E) pair and small for another, while spin-echo values for another material are large for both (T_R, T_E) pairs. We can look up approximate values for ρ, T_1, and T_2 of materials similar to those we are imaging to help choose (T_R, T_E) pairs as described in Fig. 2.1. These values are available in sources such as [Keller 88].

MR Term	Conditions	$v(x)$ Approximation
proton-weighted	$T_R \gg T_1$ and $T_E \ll T_2$	$v(x) \approx \rho(x)$
T_1-weighted	$T_R \approx T_1$ and $T_E \ll T_2$	$v(x) \approx \rho(x)(1 - e^{-T_R/T_1(x)})$
T_2-weighted	$T_R \gg T_1$ and $T_E \approx T_2$	$v(x) \approx \rho(x)e^{-T_E/T_2(x)}$

Figure 2.1: For specific materials with known values of ρ, T_1 and T_2, there are (T_R, T_E) combinations for which some terms of Eqn. 2.1 can be ignored, to first approximation. Images produced in these modes are called proton-, T_1-, or T_2-weighted. Note that in T_1-weighted images, materials with a larger T_1 produce smaller values, whereas in T_2-weighted images, materials with a larger T_2 produce larger values. T_1 and T_2 tend to be correlated – materials with a long T_1 tend to have a long T_2 as well. □

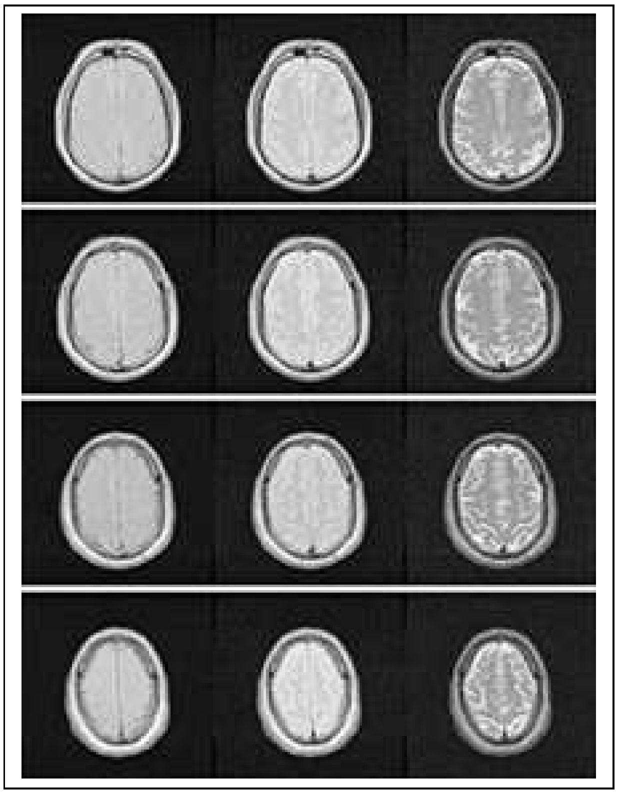

Figure 2.2: Four 2-D slices of a dataset of a human head. Samples are three-vectors. Each row contains three images of the same slice with each image representing one of the elements of the vector. The echo time, T_E, is 25ms, 50ms and 100ms from left to right. The relaxation time, T_R, is 2000ms for all images. □

Figure 2.2 shows four slices from a dataset with samples that are three-vectors. The dataset is of a human brain. Each row shows the same slice with each image showing one element of the vector. The three values making up the vector were collected with a different relaxation-, echo-time combination. Note that the contrast between tissues changes across each row. The echo time, T_E, is 25ms, 50ms, and 100ms, from left to right in each row, with the relaxation time, $T_R = 2000$ms.

2.2.3 Collection Parameters

MR datasets are influenced by several parameters which we divide into geometric and magnetic categories.

Geometric parameters determine the volume of space over which data is collected and the spatial resolution of the resulting data. They also determine the sampling kernel function $k(x)$.

Magnetic parameters include the relaxation time, T_R, and the echo time, T_E, which determine the function of ρ, T_1, and T_2 that is collected, as described in Eqn. 2.1. Figure 2.3 lists the parameters and shows typical values for them.

2.2.4 MR Sampling

Spin-echo data is collected from the MR imaging machine in the fourier domain (see [Keller 88]), and an inverse fourier transform is applied to it to produce images. The collection is often done as a set of slices, which are imaged one at a time, with an inverse two-dimensional transform performed

Geometric Parameters	Typical Values
x, y resolution	256x256, 512x512
field of view	10-30cm
x, y center	0, 0
number of slices	8-60
slice thickness	3-8mm
slice spacing	0-10mm
slice locations	varies
Magnetic Parameters	**Typical Values**
imaging protocol	spin-echo
relaxation time, T_R	100-7000ms
echo time, T_E	10-100ms
number of excitations (NEX)	1-4

Figure 2.3: Typical parameters controlling collection of medical resolution MR data. The geometric parameters determine the region of space to be imaged, and the magnetic parameters affect the sampled values for different materials, as shown in Eqn. 2.1. The number of excitations determines how many times the dataset will be collected and averaged together to improve the signal to noise ratio. □

on each slice, although it is possible to image the entire volume volume. When a set of slices is collected, the sampling kernel within each slice inherently low-pass filters the data because the data is collected in fourier space within the slice. In the direction perpendicular to the slices, the sampling kernel is often less well-behaved. It can be too narrow, or can otherwise have a frequency response that passes frequencies higher than the Nyquist limit. In either case, aliasing artifacts can be introduced into the reconstructed images.

When an entire volume is imaged, instead of one slice at a time, the data is collected in fourier space for the entire volume, and an inverse three-dimensional fourier transform is applied to the whole three-dimensional dataset. In this case, the 3-D sampling kernel low-pass filters the data in all directions, and thereby avoids severe aliasing artifacts.

2.3 Collection Guidelines

There are interdependencies and tradeoffs among the many parameters that affect MR collections. Some of the most notable goals that must be traded off are signal-to-noise ratio, spatial resolution, sampling rate, collection time, and number of values per spatial point. In general, any improvement in one of the measures above is almost always accompanied by a corresponding loss in one or more of the others. The following guidelines are contradictory in places. The advantages and disadvantages listed with the guidelines should help in making decisions about which to follow.

- Choose objects that will fit within the uniform region of the coil.
  - √ Avoids having to correct for non-uniform response of constant tissues.
  - √ Tends to reduce collection time because object is smaller.
  - × Reduces size of object that can be imaged.
  - × May push the resolution limit, since smaller objects may have smaller features that need to be distinguished.

- Choose objects with sufficiently large features.
  - $\surd$ Avoids need for data at finer resolution than is possible, either technologically or due to time constraints.
  - $\surd$ Tends to reduce collection time.
  - $\times$ Limits class of objects that can be imaged.
- Choose objects with high hydrogen density. The signal-to-noise ratio is directly related to the amount of hydrogen in the object.
  - $\surd$ Tends to reduce collection time.
  - $\surd$ Tends to increase data quality.
  - $\times$ Limits class of objects that can be imaged.
- Choose objects without magnetic materials and without very sharp discontinuities in the hydrogen density.
  - $\surd$ Avoids geometric and intensity distortions that are difficult to correct, such as ghosting and ringing.
  - $\times$ Limits class of objects that can be imaged.
- Use spin-echo collection protocol.
  - $\surd$ Reduces some geometric and intensity distortions that other protocols can cause. Is less sensitive to static magnetic-field inhomogeneities than some other protocols.
  - $\surd$ Provides good range of contrast for materials with varying T_1 and T_2 values.
  - $\times$ Can require more collection time than some protocols.
- Choose objects that do not change during the collection time, or limit collection time to the time an object can remain relatively static.
  - $\surd$ Avoids artifacts due to motion or changes during collection.
  - $\times$ Rules out most living things, and so is often not practical. Blood flow in almost any part of the body, respiration in the upper torso, and muscle movements in any part of the body can cause artifacts.
- Collect multiple different values for each point in space. Ideally, collect a proton-weighted dataset; one or more T_1-weighted datasets, with relaxation time, T_R, spanning the range of expected T_1 values; and one or more T_2-weighted datasets, with echo time, T_E, spanning the range of expected T_2 values.
  - $\surd$ Gives more opportunity for collecting data that will distinguish materials with similar characteristics.
  - $\times$ Increases collection time.
- Collect data with sufficient resolution to distinguish features of interest.
  - $\surd$ Can find smaller features.
  - $\times$ Increases collection time.
- Overlap slices if possible, or at least minimize slice spacing.
  - $\surd$ Avoids aliasing artifacts in the collected data in the slice direction.
  - $\surd$ Improves resolution.
  - $\times$ Can require interleaving two or more datasets, and can increase collection time for a fixed-size volume.
- Collect three-dimensional volume instead of slices.
  - $\surd$ Avoids aliasing problems in slice direction.
  - $\times$ Increases collection time because different slices cannot be collected simultaneously.
  - $\times$ Protocol not available on some machines.
- Reconstruct data using filter kernels that do not pass high frequencies.
  - $\surd$ Avoids aliasing problems in the reconstructed data.
  - $\times$ Can require more computational time.

2.4 Examples of Datasets

All of the data that we have used for this thesis have come from the Huntington Magnetic Resonance Center in Pasadena, California. They operate a clinical GE 1.5-tesla Signa machine, which can collect spin-echo slices down to 3mm thick, with resolution in each slice somewhat better than 1mm. The machine can image volumes within a cylinder up to approximately 750mm in diameter, and 1m long. The data produced by the machine satisfies many of the criteria mentioned at the beginning of this chapter, although it collects spin-echo data only as slices.

We have collected data of a number of objects using this machine, including human heads, a human hand, a jade plant, and several bananas. Figure 2.4 lists some of the datasets that we have collected, together with the collection parameters that we used.

We are just beginning to use a new MR facility here at Caltech. It currently collects data at a resolution of about 0.1mm, and can image volumes within a cylinder up to approximately 25mm in diameter and 25mm long.

2.5 Summary

In this chapter, we presented the goals for resolution, contrast, alignment and signal uniformity in the data-collection process. We also briefly discussed sampling and reconstructing continuous functions as these processes pertain to collecting and examining MR datasets. We then described spin-echo MR collections, which primarily samples nuclear spin density, $\rho(x)$, decayed by two characteristic time constants of a material, $T_1(x)$ and $T_2(x)$. Geometric and magnetic parameters control what volume of space and what function of $\rho(x)$, $T_1(x)$, and $T_2(x)$ will be collected. We suggested guidelines for collecting data, and listed the advantages and disadvantages of each guideline to help with deciding among them when they conflict. Finally, we enumerated the data we used for this work and the machines that we collected it on.

Object	XxY res	XY FOV	Z slices	slice thick	slice spacing	protocol	T_R	T_E	NEX
brain #1	256x256	27cm	24	3mm	1.5mm	SE	500ms	20ms	2
							500ms	40ms	2
							500ms	60ms	2
							500ms	80ms	2
brain #2	256x256	24cm	8	5mm	2.5mm	SE	2000ms	25ms	4
							2000ms	50ms	4
							2000ms	75ms	4
							2000ms	100ms	4
brain #3	192x192	22cm	21	5mm	1.5mm	SE	3400ms	25ms	2
							3400ms	50ms	2
							3400ms	75ms	2
							3400ms	100ms	2
banana #1	256x256	16cm	55	4mm	0mm	SE	3000ms	19ms	1
							3000ms	80ms	1
							650ms	19ms	1
banana #2	256x256	10cm	54	4mm	0mm	SE	5000ms	17ms	1
							5000ms	70ms	1
							600ms	18ms	2
							600ms	36ms	2
banana #3	256x256	16cm	18	1mm	0mm	SE	2500ms	20ms	2
							2500ms	40ms	2
							2500ms	60ms	2
							2500ms	80ms	2
							400ms	13ms	2
							200ms	13ms	2
jade plant	256x128	15cm	124	1mm	0mm	GRE	50ms	15ms	1.5
	256x192	15cm	41	3mm	0mm	SE	6000ms	23ms	1
							6000ms	70ms	1
							600ms	18ms	2
hand	256x256	18cm	78	3mm	0mm	SE	2000ms	23ms	1
							2000ms	50ms	1
							600ms	30ms	2

Figure 2.4: Datasets that we have collected and used as examples in this thesis. SE = spin-echo, GRE = gradient-echo, NEX = number of excitations. □

Chapter 3

Artifact Reduction

Chapter 2 suggests guidelines that can help avoid many artifacts in collected MR data. Almost all datasets, however, are collected with some artifacts. These can occur because conflicting goals prevent some guidelines from being followed, or because there is no practical way to avoid a particular problem. In this chapter we discuss some causes of artifacts and methods to reduce artifacts.

We have developed techniques for reducing several types of artifacts. First, we have been able to reduce some aliasing artifacts, at the expense of decreasing the resolution of a dataset. Second, we have compensated within datasets for non-constant response of constant materials. Third, we have aligned multiple mis-aligned scalar-valued datasets of the same object to create a new vector-valued dataset with each element a sample from one of the aligned scalar datasets. And fourth, from a dataset collected with several (T_R, T_E) pairs we have generated datasets with different (T_R, T_E) combinations so that the resulting datasets would better distinguish particular materials.

3.1 Aliasing

A sampled dataset will usually contain aliasing artifacts. This phenomenon occurs when frequencies higher than the Nyquist limit for the sampling rate are aliased to frequencies below the limit. Higher frequencies will then be included in the sampled data, even though they cannot be reconstructed from it. These higher frequencies can show up as bumps in extracted models, as seen in Fig. 3.1.

We reduce these aliasing artifacts by determining what frequencies could have aliased information in them, and band-limiting the function to the lowest such frequency. This band-limiting causes correct information above that frequency to be lost; this reduces the resolution, but also reduces the aliasing. As it is not possible to separate the correct data from the aliased data above that frequency, maintaining only the correct information is not possible.

3.2 Non-constant Response of Constant Material

A second artifact often found in MR data is non-uniform data values for a single material. Figure 3.5 shows an example of this problem where a slice from a wrist dataset gets quite "dim" toward the bottom of the image; the left image of Fig. 3.3 shows a slice of a bunch of bananas where the image gets dim at the right and left edges. This problem has several possible causes. First, the RF coil used as an antenna for exciting and measuring signal from the volume may not produce a uniform field (see Section 2.2.1). Second, the static magnetic field of an MR machine can also have inhomogeneities that cause similar distortions. And third, collection software can cause systematic distortions of the image values. Software may try to avoid "wraparound" artifacts, where portions

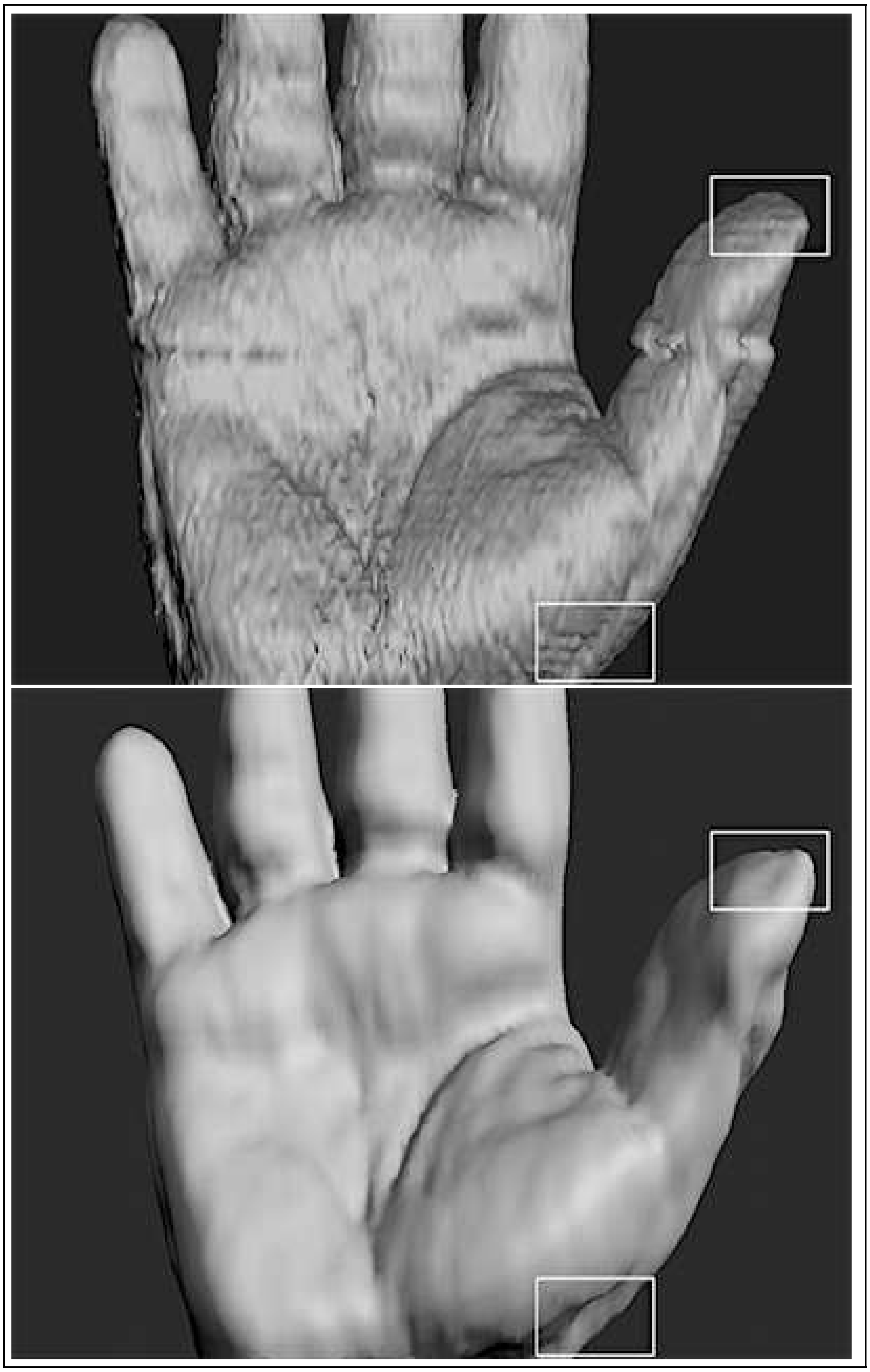

Figure 3.1: Geometric model of a human hand extracted from filtered MR data. The model on the top was created directly from the data collected on the machine. The model on the bottom was created from a dataset that had reduced aliasing artifacts. Note the bumps on the model in the bottom image, particularly the horizontal ridges within the boxes on the end of the thumb and near the bottom right of the palm. Figure 3.2 shows details of these areas. □

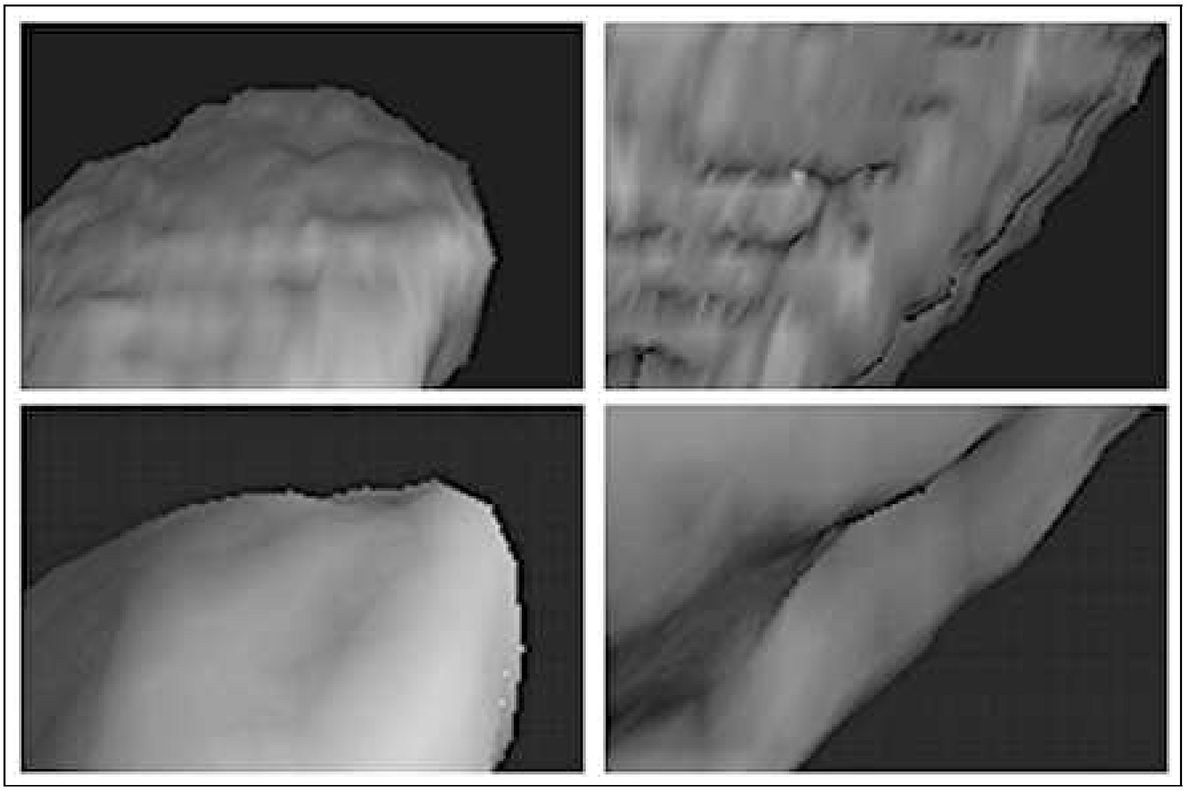

Figure 3.2: Details of two areas of the geometric models of the hand. The top images show the areas within the boxes in the top image of Fig. 3.1, which was created from data with aliasing artifacts. The bottom images show the areas within the boxes in the bottom image of Fig. 3.1, which was created from data with aliasing artifacts reduced. Note the horizontal ridges in each of the top images. The bottom images do not show this artifact. □

of an object outside of the volume being imaged appear within the volume on the opposite side, by attenuating the RF it delivers to slices near a face.

We have experimented with ways of correcting this problem, although more work needs to be done in this area. We have tried two methods: The first makes the assumption that the attenuation of the signal is a function of only one of the coordinate directions. This is a reasonable assumption for the latter case where a cylindrical coil becomes less effective near its ends, and for the case where the software has attenuated the response as a function of one direction. For this case, we can statistically analyze the data in each "slice" of data perpendicular to the axis along which we are assuming the variation occurs. By identifying a reference statistical value that should be the same in each slice, we can linearly transform the data to make the reference values constant. The mean or the median are reasonable candidates for this reference value. Figure 3.4 shows a few sample histograms of slices on the left and the reference values as a function of z on the right.

While this technique can correct for some artifacts, it tends to significantly increase the noise in slices that are scaled by a large amount. As a result, it is far preferable to collect data that does not suffer from this problem by imaging within the uniform region of the coil and magnet.

A second method for reducing these artifacts does not operate with the assumption that the attenuation is a function of one of the coordinate variables. This technique requires collecting data of a single material, e.g., water, within the same volume as the dataset of object, and ensuring that the two images are aligned properly. By dividing each value in the image of the object by

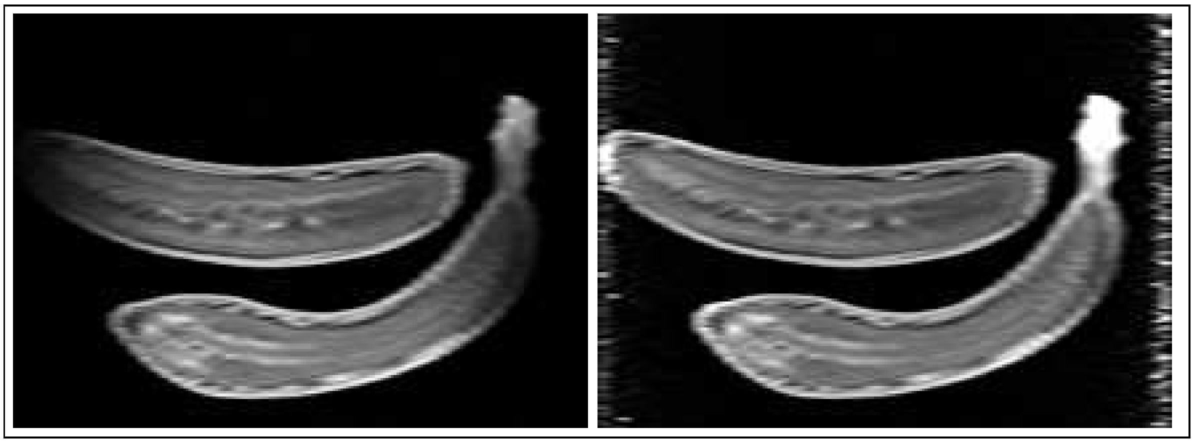

Figure 3.3: The image on the left shows non-uniform response artifacts. Note the dim areas near the left and right edge of this image. The brightness as a function of the horizontal, or z, axis is shown in Fig. 3.4. The image on the right shows the left image corrected for attenuation. Note the increased noise near the left and right edges that occurs as a result of the very low signal-to-noise ratio there. □

Figure 3.4: The left graph shows histograms of four constant z slices from a dataset whose values vary as a function of z. The value at which the peak occurs in each histogram increases as a function of z. The right graph shows the relative location of each peak as a function of the z value of the slice. The four marked locations on the right graph correspond to the locations of the peaks in the four sample histograms on the left. □

the corresponding value in the image of the water, we adjust for any inhomogeneity. As with the previous technique, this technique tends to increase noise where the dataset is scaled by a large amount, and so collecting data within the uniform region of the coil and magnet is preferable.

3.3 Alignment of Datasets

In many cases, MR data can be collected so that each of the scalar values making up the vector at each spatial point are aligned. When the datasets are not aligned, for example when a live subject moves between acquisitions, the data must then be computationally aligned.

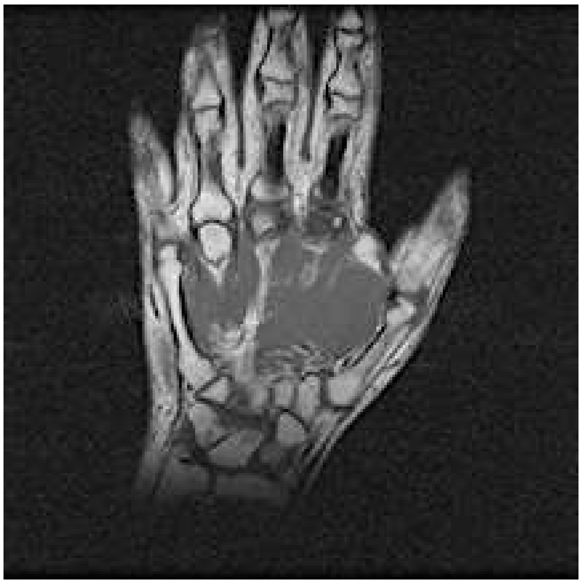

Figure 3.5: One slice from a high-resolution image of a human hand. Note that the intensity of the image falls off particularly near the bottom of the wrist. □

```
choose one dataset as a fixed, reference dataset
for each moving dataset
    align moving dataset with fixed datasets
    add to list of fixed datasets
end
```

Figure 3.6: The alignment algorithm. We align datasets one at a time relative to those datasets that have already been aligned. □

This problem is particularly difficult because even perfectly aligned images will look different. Features within the images will be in the same places, but their dataset values may be higher or lower, depending on their collection parameters. We somehow need to map from values in one image to expected values in another image in order to compare directly. We also must address the fact that the volume images we are aligning will not occupy the same space, and so we must correctly deal with areas that are only inside a single volume, as well as with areas of overlap.

Consider a vector-valued volume dataset as several scalar-valued volume datasets, one for each element of the vector. We align multiple such scalar datasets by holding one or more of them fixed in position and moving and deforming another until it is optimally aligned. Then we repeat the process, using the newly aligned dataset to help align the additional ones. The dataset that is chosen as the initial fixed reference should distinguish as many distinct materials as possible, since this will give the most information for aligning additional datasets.

To perform the alignment, we define a correlation function $E(p)$ that tells us how well a deformed dataset is registered with respect to a collection of fixed datasets. $E(p)$ is a function of the parameters, p, of a deformation function that moves and deforms the moving dataset. We minimize $E(p)$ to find the optimal parameters of the deformation so that the moving dataset is registered with respect to the fixed ones.

Deformation Function

There are a number of design criteria for the deformation function of the moving dataset. The deformation function should rotate, translate, and scale the moving dataset, since this would account for patient motion and the inaccuracy of positioning and orientation measurements of the MR machine. In order to compensate for geometric distortions due to distance from the detector coil, the deformation should also skew the moving dataset [Bradley 85]. While there can be more complex geometric distortions due to magnetic field inhomogeneities and other effects, we have chosen to ignore them and use a quasi-linear deformation. This choice achieves all of the goals above, is reasonably simple, and empirically seems to work adequately.

Correlation Function

We define the correlation function, $E(p)$, in terms of one or more fixed, or stationary, datasets and one moving dataset. $E(p)$ integrates the square of an error function over the volume of interest. This least-squares method is a generic approach to many types of optimization problems. Consider the basic form of the following equation:

$$\hat{E}(p) = \int_V e^2(x,p)dv, \tag{3.1}$$

where $\hat{E}$ is the correlation function; $e(\underline{x},p)$ is some sort of error function between the stationary datasets and the moving dataset; x is the point at which we are evaluating the error function; p represents the parameters of the deformation of the moving dataset; and V is the volume of overlap of the datasets.

This first choice has an undesirable solution for minimizing the function $E(p)$: The datasets can be moved so that they do not overlap at all. We can correct that bias by normalizing the correlation function $E(p)$ by the overlap volume of the datasets in question.

$$E(p) = \frac{\int_V \omega(x)e^2(x,p)dv}{\int_V \omega(x)dv}, \tag{3.2}$$

where ω is a weighting function that is zero where any dataset is undefined, and one where they all are defined. The denominator eliminates the bias of the overlapping volume. For this choice, $E(p)$ is thus an average error value, or a "specific error" (error per unit volume), for the set of points within the volume. The correlation function, $E(p)$, is defined as long as there is some overlap between the volumes. Typically we expect that the overlap will be significant, since the images usually represent the same geometry.

Now we refine the error measure, $e(x,p)$, of the equation:

$$e(x,p) = D_m(x,p) - C(D_f(x)), \tag{3.3}$$

where $D_m(x,p)$ and $D_f(x)$ are the data values within the moving and fixed datasets, respectively, and a conversion function, C, converts the values from the fixed dataset into a value that should appear in the corresponding location in the moving dataset. The moving dataset function, $D_m(x,p)$, deforms the moving dataset according to the parameters p, and then returns the value of the dataset at the specified point.

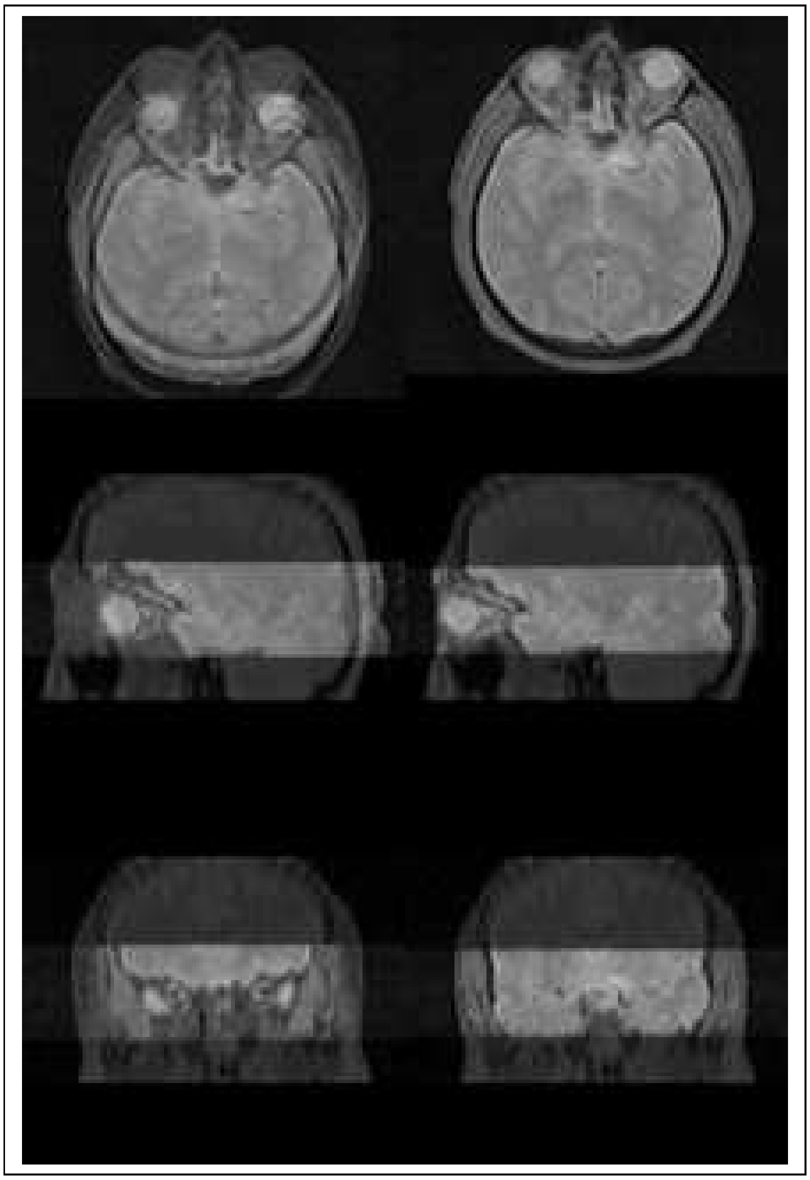

Figure 3.7: An overlay of two volume datasets before and after alignment. On the left we see three views of the datasets before alignment. Note in the top left that the lighter colored dataset is significantly lower than the darker dataset. On the right, they are aligned in all three views. □

Relating Values in Different Datasets

The conversion function, C, is of key importance in our algorithm because it is the mechanism by which we are able to relate data values from different datasets. Without the C function, we would not be able to compare and align values from datasets with differing MR acquisition parameters. For the purposes of alignment, we have used an easy-to-implement supervised classification technique to calculate C, although an unsupervised technique could be substituted. Within each dataset, we interactively select points representative of each material. Values from the fixed dataset are classified with a tentative material based on their relationship to these points, and the mean value for that material in the moving dataset is returned by C.

We approximate the integral $E(p)$ with a sum over a finite number of points using numerical techniques, such as Monte Carlo integration, and have minimized it using a quasi-Newton minimizer [NAG]. Figure 3.7 shows two volume datasets before alignment, on the left, and after alignment, on the right.

3.4 Calculating New Spin-echo Values

The fourth type of processing that we perform on the MR data can help to create data values that differ for different materials. As discussed in Chapter 2, T_E and T_R significantly affect the values collected within a dataset. By collecting datasets for several values of T_R and T_E, we can calculate ρ, T_1, and T_2 for individual points in the collected volume. From these values, we can calculate new data values for those points as if they were collected with different T_R and T_E values. We do not need to actually collect this new data. We can vary the T_R and T_E to find the best contrast.

To calculate new datasets, we use the following equation based on Eqn. 2.1:

$$v_i = \rho(1 - e^{-T_{Ri}/T_1})e^{-T_{Ei}/T_2}. \tag{3.4}$$

For the fixed values of ρ, T_1, and T_2 at a spatial point corresponding to a sample, Eqn. 3.4 defines the values, v_i, that will be measured by an MR machine as a function of T_R and T_E. Given a vector of collected values, v_i, and the corresponding vectors, T_{Ri} and T_{Ei}, we can find a least-squares solution of Eqn. 3.4 for ρ, T_1, and T_2.

With this solution, we can calculate a new data value for any given T_R and T_E, and so can choose (T_R, T_E) pairs that produce the best possible contrast. Note that Eqn. 3.4 has three parameters, so we need at least three collected values for each voxel in order to solve the equation. More collected values will improve the least-squares solution.

3.5 Summary

In this chapter, we have discussed techniques for reducing four types of artifacts in MR data. Reducing these artifacts gives us datasets that can be used to produce more accurate models. First, aliasing artifacts can be reduced by low-pass filtering, using a filter kernel with a smooth dropoff in frequency response at its boundaries. Second, inhomogeneities in the values for a simple material within a volume can be normalized with respect to a statistical value calculated from the data, or with respect to an additional collected reference value. Third, datasets can be aligned by deforming them with respect to one another, and minimizing and correlation function that determines how closely they are matched. And, fourth, contrast can be enhanced by calculating datasets for new values of T_R and T_E from collected datasets.

Chapter 4

Material Classification

In this chapter we present an algorithm for identifying different materials in a volume dataset and creating new datasets that represent the probability of each material in the object as a function of 3-D space. The main goal for our material-classification technique is to enhance models and images created from sampled volumetric data. As [Drebin et al. 88] points out, classification techniques that classify voxels as one of a fixed set of materials cause artifacts when used for computer graphics because the techniques tend to create incorrect hard edges between materials. To avoid this problem, we produce smoothly varying classified voxel data. The classified voxel values represent the probability that a voxel is a particular material. Of course, the probabilities at a given voxel for all materials must sum to one.

1. Histogram construction
2. Gaussian fitting
3. Data classification

Figure 4.1: The steps of the material-classification algorithm □

The steps of the algorithm are listed in Fig. 4.1. First, we create a multi-dimensional histogram of the values in the image. If we have a sufficiently large number of samples in a volumetric dataset, the law of large numbers [James 76] suggests that the distribution of MR values for a given material will be gaussian. Second, we model the histogram as a sum of parameterized gaussian functions, where each function represents the distribution of samples for a single material in the volume. We find parameters for the collection of gaussian functions that make the model agree with the histogram. Third, material probabilities are calculated by examining the sampled voxel data and using the model to lookup the likelihood of each material given that data value.

One problem with modeling the histogram with a sum of gaussians is that noise in the histogram can obscure a distribution for a material that only occupies a small portion of the volume. Since our work on extracting models is usually concerned with the main materials that make up an object, this problem is less limiting for us than it might be for other applications, where an abnormality covering only a few voxels could be difficult to find in the histogram.

4.1 Terms and Definitions

We first introduce some notation and definitions we will use to describe our algorithm.

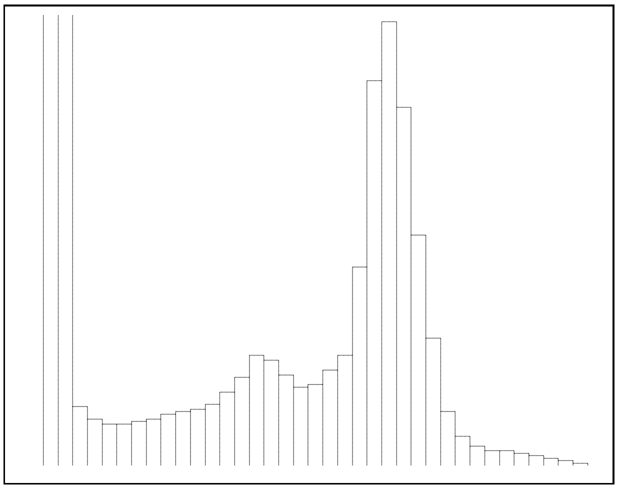

Figure 4.2: A one-dimensional histogram of a a human brain dataset. Each "bin" in the histogram is a constant value. Note the three peaks in this histogram and compare this figure to Fig. 4.8, which shows the set of gaussians fit to it. □

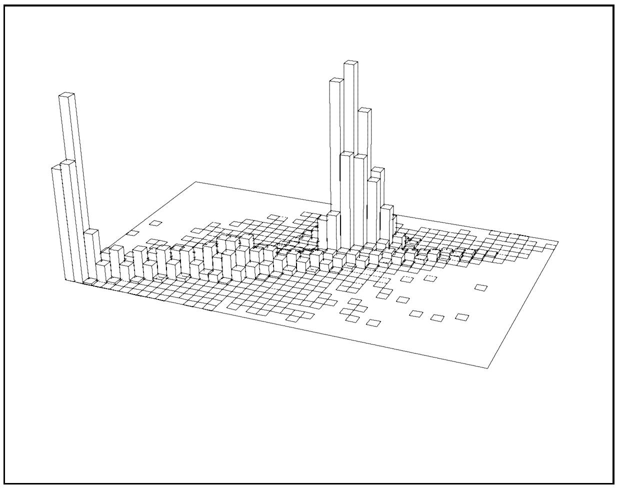

Figure 4.3: A two-dimensional histogram of a human brain dataset. Note the correlation between this histogram and the one-dimensional histogram in Fig. 4.2. Note also the correspondence in peaks in the histogram to the centers of colored regions in Fig. 5.4. □

Data

Collected data, $v = V(x)$, is an n-vector for each sampled point x in $\mathbf{R}^3$.

Histograms

We define a point in *feature space* as a point in $\mathbf{R}^n$ that corresponds to a particular vector of data values. A histogram discretizes feature space into *bins*, or rectangular regions. The histogram function

$$h = H(v)$$

is a piecewise constant function over these bins and maps $\mathbf{R}^n \to \mathbf{R}$. Within a bin, the histogram function, $H(v)$, is constant. $H(v)$ returns the number of voxels with data values in the bin containing v, divided by the size of the bin, or the number of voxels with a particular range of data value per unit volume. Figures 4.2 and 4.3 show a one-dimensional example and a two-dimensional example of a histogram.

We call the set of bins in a histogram $\mathcal{H}$. Given a bin, $b \in \mathcal{H}$, we define the bin location v_b as a feature space point within the bin, h_b as the value of the histogram at that point, and s_b is the volume of the bin in feature space. $s_b h_b$, then, is the cumulative frequency of all data values within a bin, b.

Gaussian Functions

A *gaussian* is a multi-dimensional gaussian bump function representing a distribution of data values [Larson 82]. The parameters $p_i = [h_{p_i}, c_{p_i}, w_{p_i}, r_{p_i}]$ controlling the gaussian bump are shown in Fig. 4.4.

	type	description
h_{p_i}	scalar	height
c_{p_i}	n-vector	center in feature space
w_{p_i}	n-vector	pre-rotation axial widths
r_{p_i}	$\frac{(n^2-n)}{2}$-vector	rotation

Figure 4.4: Parameters of a n-dimensional gaussian □

Figures 4.5 and 4.6 illustrate the effects of the parameters of one-dimensional and two-dimensional gaussians, respectively.

We define the gaussian function as:

$$g(v, p_i) = h_{p_i} e^{-|t(v, c_{p_i}, w_{p_i}, r_{p_i})|^2}$$

where t transforms the point v to appropriately shape the gaussian, and is given by:

$$t(v, c, w, r) = (R(r)W(w))^{-1}(v - c).$$

$W(w)$ is a diagonal matrix containing elements of w and $R(r)$ is a rotation matrix parameterized by r. For one dimension, r is a zero-vector and R is the identity. For two dimensions r is a scalar,

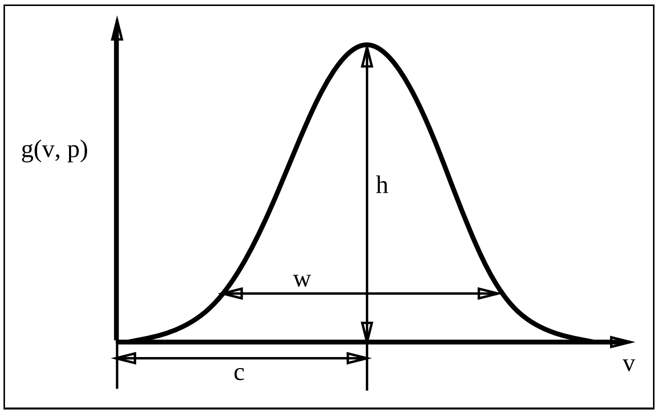

Figure 4.5: Parameters controlling a one-dimensional gaussian □

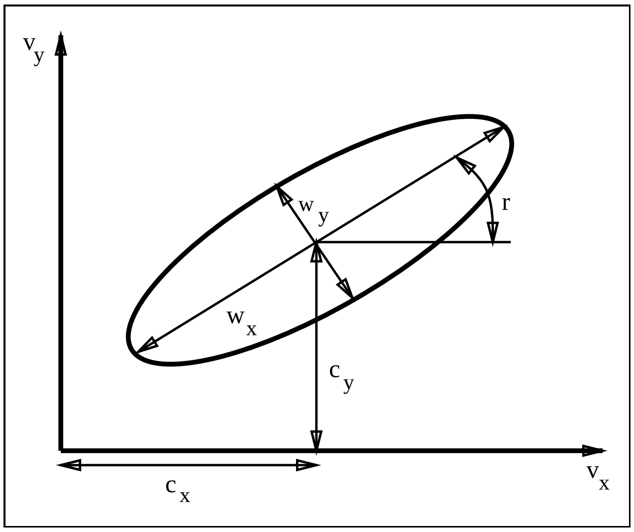

Figure 4.6: Parameters controlling a two-dimensional gaussian. The elliptical shape is a contour of the height function of the gaussian. □

and R is a rotation around the origin. For three dimensions, r is a three-vector and the rotation matrix is a rotation around the axis represented by $(-r_2, r_1, -r_0)$ with a magnitude equal to the length of r.

For the general d-dimensional case, we construct an anti-symmetric matrix ω with the elements of r above the diagonal and their negatives below. R, then, becomes e^{ω}.

We define a sum of gaussian functions, $G(v, \mathcal{P})$, as

$$G(v, \mathcal{P}) = \sum_{i=1}^{||\mathcal{P}||} g(v, p_i)$$

where $\mathcal{P} = \{p_i\}$ is the collection of all the parameters of all the gaussians and $||\mathcal{P}||$ is the number of gaussians.

4.2 Histogram Construction

Histograms for scalar volume data $H(v)$ are piecewise constant curves as shown in Fig. 4.2. Note that there are three clearly visible peaks in that figure, each of which will be fit by a gaussian in the model of the histogram. To improve running time, a histogram should have bins large enough to contain sufficient samples so that the resulting functions are relatively smooth, but small enough that multiple individual bumps in the histogram do not merge together and become indistinguishable.

Histograms for volume data with two values at each point (i.e. feature space is $\mathbf{R}^2$) are functions from $\mathbf{R}^2 \rightarrow \mathbf{R}$, or height fields, where the two axes of the base represent the different data values and the height represents the frequency of that combination of values. See Fig. 4.3 for an example of a two-dimensional histogram.

while (max of $(H - G)$ is large enough)
  (a) find max to use as new gaussian center
  (b) fit new gaussian without moving center
  (c) fit new gaussian varying all parameters
  (d) fit "nearby" gaussians
end
(e) fit all gaussians

Figure 4.7: The histogram-fitting algorithm. We incrementally fit one gaussian at a time to the histogram, and perform a global optimization at the end □

4.3 Gaussian Fitting

We model histograms as a sum of gaussians because the set of samples in a dataset represents distributions of the sample values of the materials in the dataset. We make the assumption that the distributions will be gaussian, and that we can fit a parameterized gaussian to each peak in the histogram.

Our problem, then, is to find a set of parameters $\mathcal{P}$ and a $||\mathcal{P}||$ such that $G(v, \mathcal{P})$ fits a particular histogram $H(v)$. Given a particular number of gaussians, finding the $\mathcal{P}$ which globally minimizes

$$E_{ideal}(\mathcal{P}) = \int_{\mathcal{V}} w(v)(H(v) - G(v, \mathcal{P}))^2 dv.$$

gives us the best possible match to the histogram. $\mathcal{V}$ is all of feature space, and $w(v)$ is a weighting function used to indicate areas of interest to weight more heavily. In general, we use $w(v)$ to isolate the evaluation of E_{ideal} to the area immediately around a peak in the histogram so that only data from that histogram will affect the minimization. If $w(v)$ is set too wide, then a nearby peak may pull a gaussian away from a peak to which it should stay near. Note that E_{ideal} is defined so that it is small where the model and the histogram agree and increases as they disagree.

We approximate E_{ideal} as a sum over the bins of the histogram we are fitting:

$$E_{hist}(\mathcal{P}_j) = \sum_{b \in \mathcal{H}} w(v_b)(h_b - G(v_b, \mathcal{P}_j))^2 s_b. \tag{4.1}$$

Our algorithm is incremental, with the parameters for minimization j represented by $\mathcal{P}_j$. We first build up an initial guess at a global minimum, and then perform a global minimization starting at that guess. In addition to giving us a starting point with a reasonably small energy function for the global minimization, this incremental process helps us to determine the number of bumps in the model. By finding and locally fitting a single gaussian at a time, we are able to run minimizations with a small number of parameters. For each minimization we hold some parameters in $\mathcal{P}_j$, fixed and let others vary. We will give details on this process below. We also vary $w(v)$ to indicate different areas of interest. When we have fit enough gaussians, we can use the parameters in $\mathcal{P}_j$ as a starting point for a global minimization with all parameters varying. Since these parameters are relatively close to optimal already, we avoid many higher local minima and reduce search time to find a minimum. This final minimization over all parameters also catches any details we may have missed in the histogram by focusing on peaks with $w(v)$. Figure 4.7 summarizes the steps in the algorithm.

In all cases $w(v)$ is zero where there are no histogram bins. The gaussians tend to stick near maxima in the histogram, and so they are generally very small where there are no histogram bins. Ignoring their contribution there could allow a gaussian to move out into that region, but in practice this does not happen. Moving away from a histogram maximum out into the area where the weighting function $w(v)$ is zero would leave the maximum uncovered, thus increasing E_{hist}.

Step (a): In the first step of the algorithm we search $H(v) - G(v, \mathcal{P}^{j-1})$ for the most important peak to fit next, finding a global maximum by looping through all of the bins of the histogram. To avoid picking a point that, through noise, has an artificially high value, we require that the histogram points immediately neighboring the maximum be positive, but less than the chosen point. There can be sufficient noise in the histogram that we will find the algorithm over-fitting the data by trying to fit very short or very narrow maxima in $H(v) - G(v, \mathcal{P}^{j-1})$. In this case it is sometimes necessary to manually terminate the loop and move on to step (e).

Step (b): In the second step we attempt to fit to the histogram a gaussian centered at the maximum found in the first step, with the height of the gaussian set to the value of the histogram at that location, and with an initial width in each dimension a small multiple (3-10) of the histogram bin size. We wish to fit the shape of this gaussian without too much influence from any other peaks in the histogram, so we set the weighting function $w(v)$ to a copy of the trial gaussian with the widths decreased by some factor (we have found 25-50% to work well) and the height reduced to one. $w(v)$ does not change during the minimization. During this minimization we vary the width, height, and rotation parameters of the trial gaussian while the center parameters remain fixed.

Step (c): We now have an approximate shape and location for the gaussian. We perform a minimization where all the parameters of the trial gaussian are unconstrained, resetting $w(v)$ to a gaussian with a height of one and widths 25-50% those of the changed trial gaussian, again to avoid being adversely affected by nearby unrelated bins in the histogram.

Step (d): One more minimization within the loop minimizes all of the parameters of all the gaussians *near* the trial one. This step is necessary because the trial gaussian may have distorted the gaussian sum where it overlaps with other bumps, and could therefore make the next maximum difficult to find. We define two gaussians as *near* if either evaluates greater than $\epsilon_g = 10^{-4}$ at the center of the other. In this step we set $w(v)$ to the sum of all the nearby gaussians, again with their heights all reduced to one and with their widths all reduced by 25-50%.

Step (e): Once we are unable to find a new maximum, or the loop is manually terminated, we perform a final minimization where we set $w(v)$ to one for all non-zero histogram entries and allow all parameters to be varied. Figure 4.8 shows a one-dimensional histogram that this algorithm has fit with nine gaussians.

Minimization

We use a quasi-Newton minimizer [NAG]. It runs faster when the objective function is scaled to lie between zero and one. We scale the function in Eqn. 4.1 by

$$\frac{1}{\sum_{b \in \mathcal{H}} w(v_b) h_b^2 s_b}$$

This scaled version of $E_{hist}(\mathcal{P}_j)$ is one if $G(v, \mathcal{P}_j)$ is zero, and zero if $G(v, \mathcal{P}_j)$ fits the histogram perfectly.

The parameters in $\mathcal{P}_j$, which are gaussian heights, centers, widths, and rotations, are defined relative to feature space, and so do not lie within the range $(-1, +1)$. We calculate an approximate minimum and maximum for each parameter, and map the minimum to -1 and the maximum to

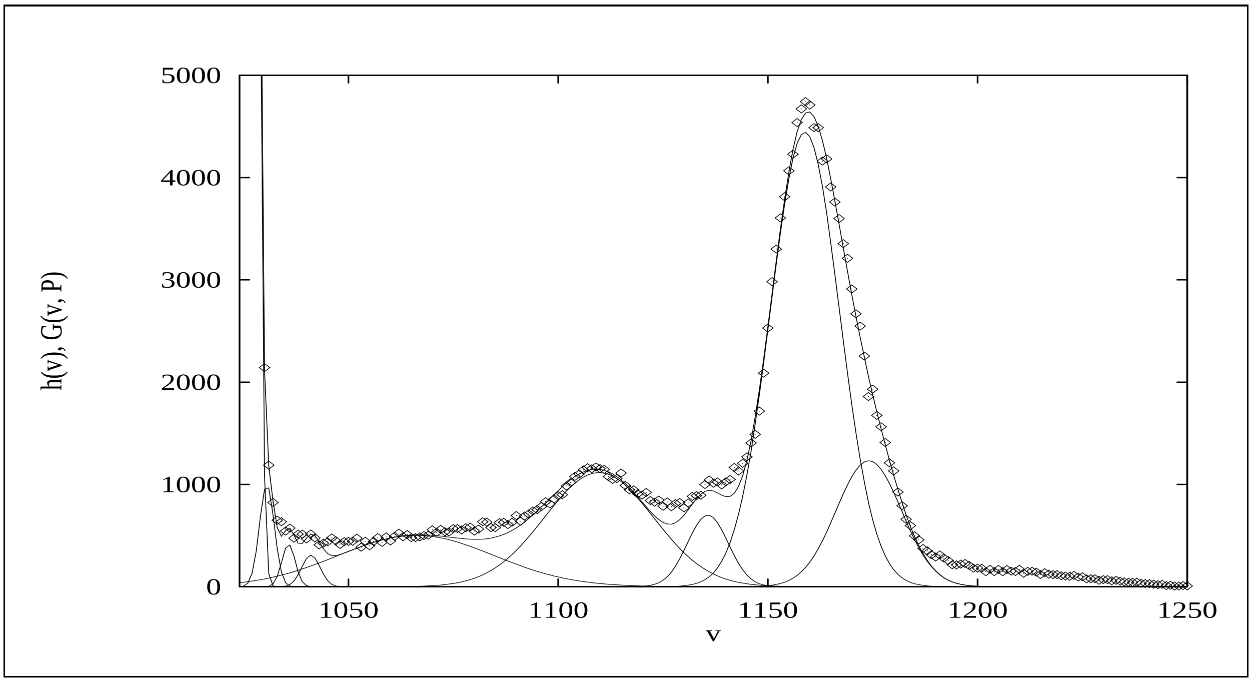

Figure 4.8: one-dimensional histogram and a nine-gaussian model fit to it. The diamonds represent histogram values, and each gaussian is drawn separately in addition to the sum $G(v, \mathcal{P})$. This figure corresponds to the histogram in Fig. 4.2. □

$+1$. For example, the center of the gaussian must lie within the range of the histogram in feature space, so we linearly map that range onto $(-1, +1)$.

The function that calculates E_{hist} is called many times for each minimization, and must integrate over all non-zero bins in the histogram for each evaluation of E_{hist}. As a speedup to each minimization but the final one, we examine the possible contribution of each histogram point to the resulting E_{hist} in advance and only sum over those bins where $w(v_b)h_b^2 s_b$ is greater than some small ϵ_w. The minimizations do not appear to be very sensitive to this parameter, which we have varied from 10^{-15} to 10^{-7}. Even if this introduces errors in the early steps, the final optimization corrects any problems.

Our minimizer can calculate numerical derivatives of the objective function $E_{hist}(\mathcal{P})$ with respect to its parameters. Because we have more information about the function, we can calculate the derivative more quickly. From Eqn. 4.1,

$$\frac{dE_{hist}(\mathcal{P})}{dp_i} =$$
$$-2 \sum_{b \in \mathcal{H}} w(h_b)(h_b - G(v_b, \mathcal{P}))G(v_b, \mathcal{P}))\frac{dG(v_b, \mathcal{P})}{dp_i} s_b$$

We can calculate this derivative in the same loop that calculates $E_{hist}(\mathcal{P})$. Since each parameter of $G(v, \mathcal{P})$ only affects one gaussian, we can calculate $\frac{dG}{dp_i}$ efficiently using finite differences on the single gaussian that p_i affects.

4.4 Classification

With the fitted gaussian model we can convert the original MR data into probabilities. Let M be the set of materials. There will be one material m_i for each gaussian in the model. We wish to find the probability of a particular material m_i at a particular point x in modeling space. Using Bayes' law we can calculate that probability as

$$P(m_j|V(x)) = \frac{g(V(x), p_j)}{G(V(x), \mathcal{P})}$$

Parts of an object may contain more than one material and so may be represented by more than one gaussian. In that case, the probabilities of the different materials can be added together to produce the probability of a such a compound material.

4.5 Summary

In this chapter we have presented an unsupervised algorithm for classifying vector-valued volumetric data. The algorithm creates a multi-dimensional histogram of the collected data, incrementally fits a sum of gaussian distributions to the histogram, and applies Bayes' law to infer material probabilities.

Chapter 5

Applications to Visualization

This chapter presents some application areas which can take advantage of datasets that have been produced using the techniques described in the previous chapters. These application areas are *computational staining*, or creating datasets of colors from datasets of other values; volume rendering, which creates images of the entire three-dimensional volumes; and model extraction, which produces three-dimensional models from the data.

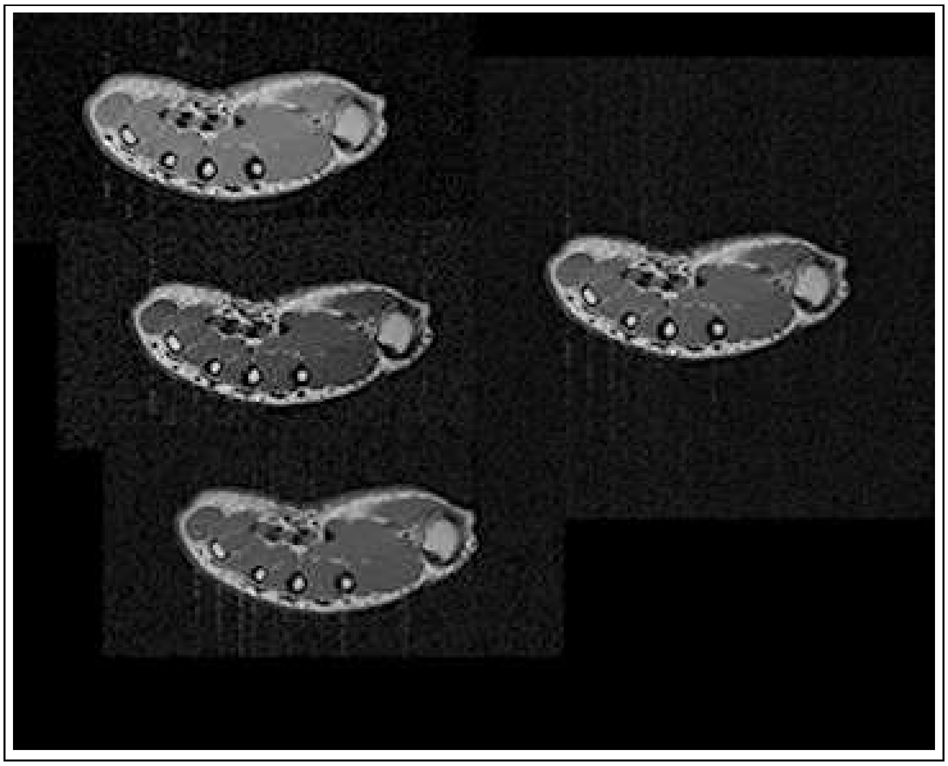

Figure 5.1: The three images on the left are three echos from one slice of a human hand dataset. The image on the right is a combination of the three on the left with one echo on each of red, green, and blue. □

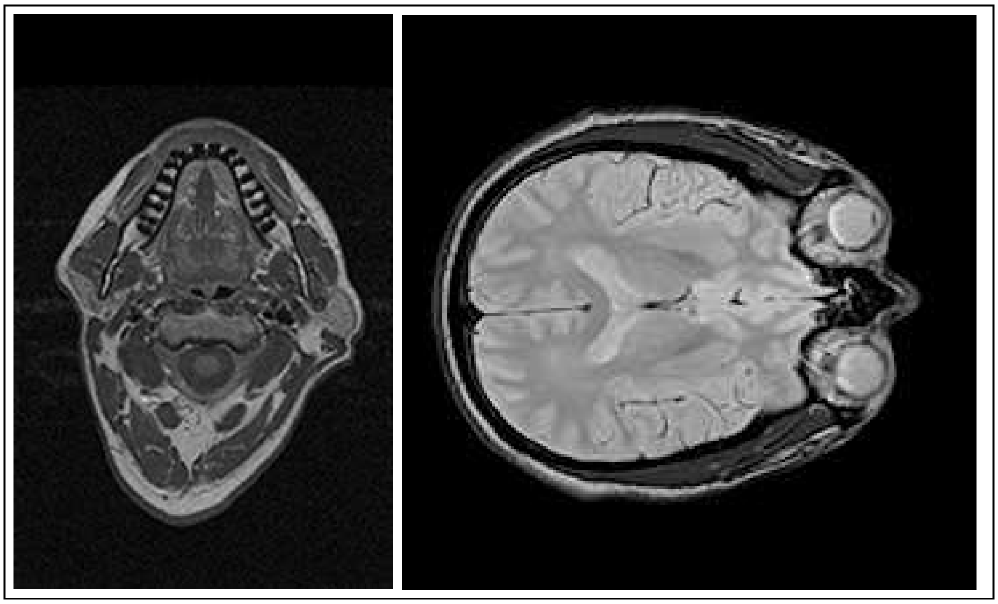

Figure 5.2: Each image is of a single slice from a multi-echo MR dataset with one echo on each of red, green, or blue. The left image is from a human neck and the right is from a human brain. □

5.1 Applications to Computational Staining

In this section we discuss several methods for staining a vector-valued MR dataset to produce color images. Some of these techniques are simple and do not require the classification step described in the previous chapter while others depend on it

The first method, which we call "RGB," combines three different images into one by simply displaying one as the red channel, one as the green channel, and one as the blue channel. The main advantage of this method is that it tends to preserve smooth variations in the original data; it does not introduce abrupt transitions in the data that could be interpreted incorrectly as boundaries between different materials. It is also quite simple. Its major disadvantage is that it cannot significantly increase differences between materials, and so two materials that appear similar before staining will appear similar after staining as well. Figures 5.1 and 5.2 show examples of this staining technique.

The "RGB" technique performs a linear transformation from feature space to RGB color space. Another staining technique, which we call "SVD," also performs a linear mapping. For this technique we interactively pick a number of points in the original vector-valued dataset and identify each point as a particular material. Each material is assigned a color, and so for each spatial point, we have a feature space point and a color. By arranging those points into a matrix and finding its pseudo-inverse using singular value decomposition (SVD), we get the least-squares "best" linear transformation from feature space to color space. Applying this mapping to an image slice gives images like that on the left of Fig. 5.3. See [Kirk 90] for more information.

This technique has the advantage of being able to better separate different tissues than the RGB technique. We can also choose particular colors for particular materials. Its primary disadvantages are that it requires classification of a number of data points, it is sensitive to the location of the origin in feature space, and it is a linear mapping, which prevents it from being able to separate materials that are represented by linearly dependent vectors in feature space.

A third staining method produces a non-linear mapping from feature space to color space. In

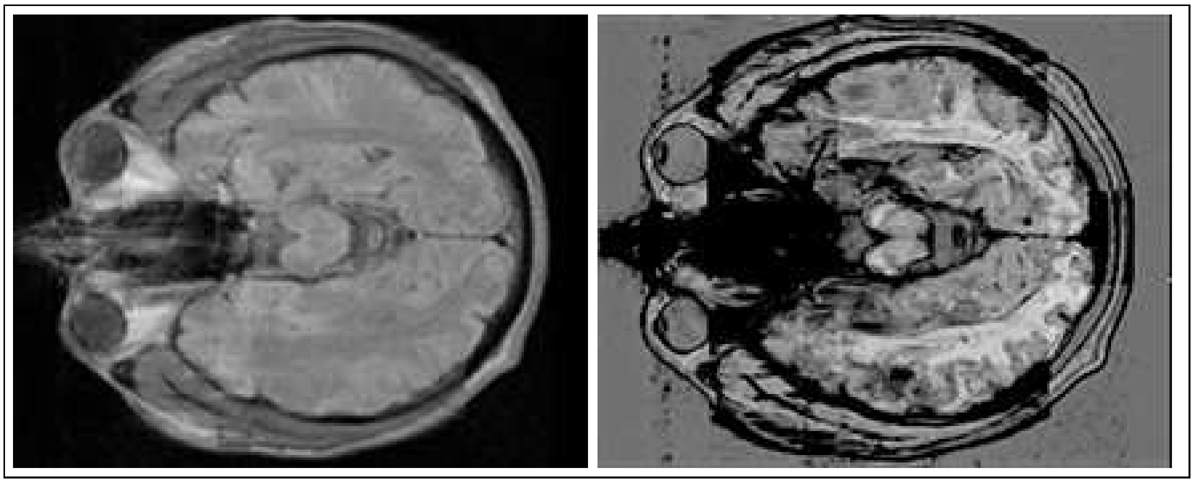

Figure 5.3: Computationally stained brain slice. The left image uses interactively selected tissue points and the SVD algorithm to find the optimal least-squares linear mapping from original data to colors. The right image uses interactively selected tissue points and the variable filter algorithm to select colors at each point. □

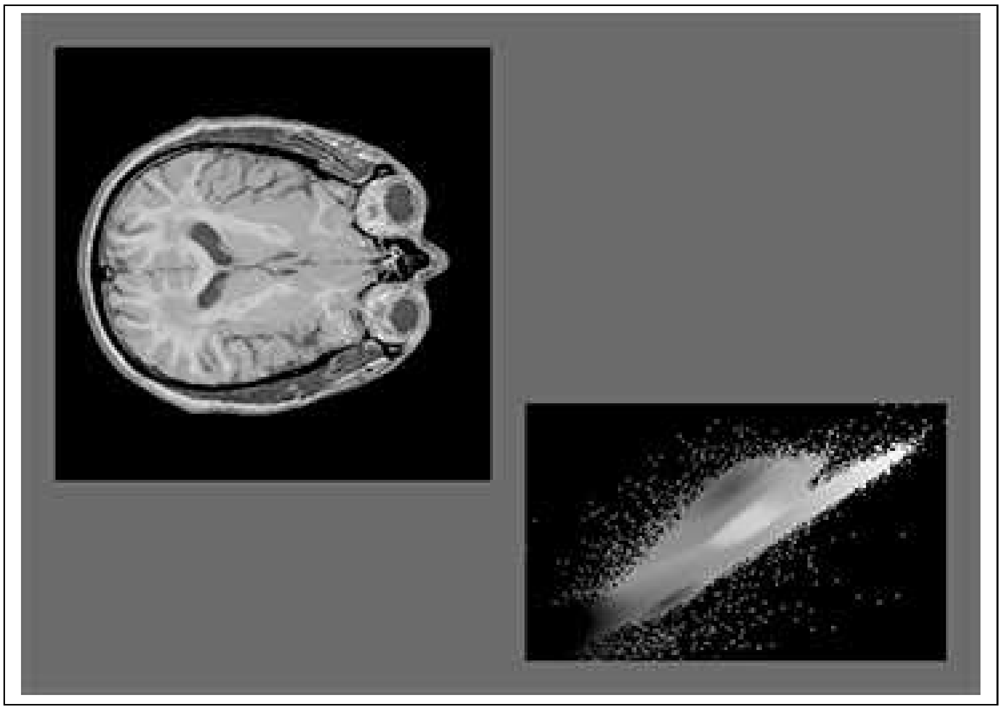

Figure 5.4: The lower right image shows the histogram of the dataset from the brain image on the right of Fig. 5.2. Each gaussian is assigned a constant hue. The color at each point is a combination of the colors for nearby gaussians. The saturation of the color decreases for points further from the center of each gaussian and the value increases for points further from the origin. The upper left image is a computationally stained slice of MR dataset of the same brain. Note the differentiation of different tissues made possible by the classification of the two-dimensional histogram. □

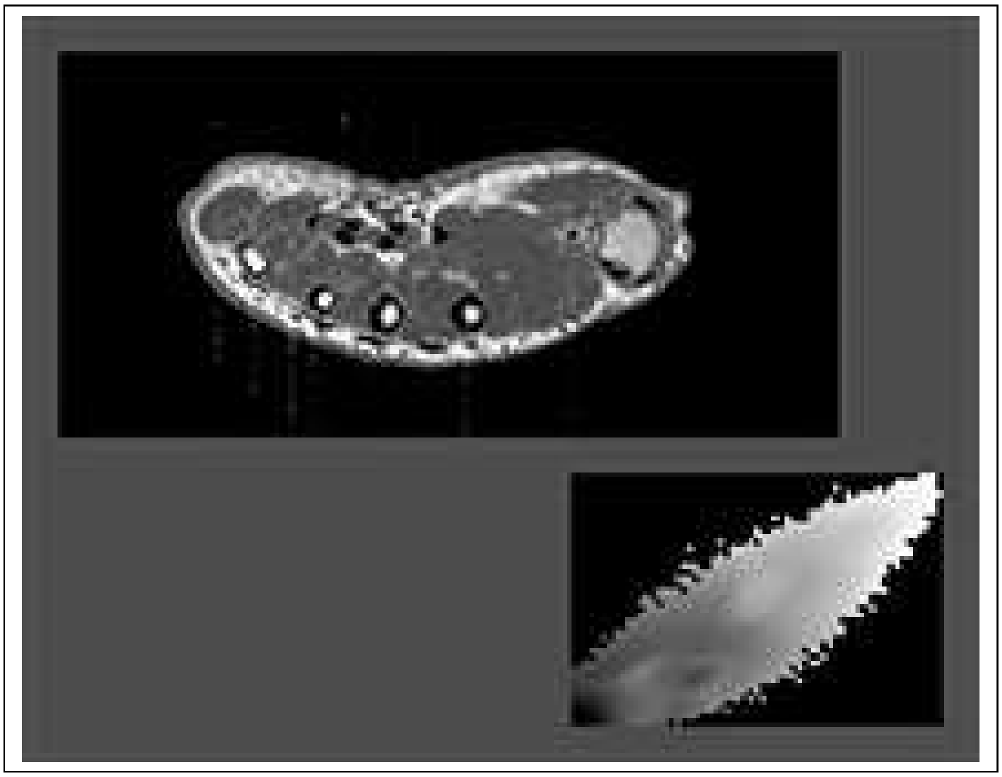

Figure 5.5: Colored histogram and slice from classified hand dataset □

this method, which we call "variable filter," we use the interactively classified points to calculate a mean and covariance matrix which define a simple probability distribution for each material. By assigning a color to each material, either algorithmically or interactively, and coloring each feature space point according to that distribution, we get images like that on the right of Fig. 5.3.

This method differentiates materials that are relatively similar in the original data much better. White matter and grey matter, different tissue types within the brain, are colored pink and yellow in the figure, illustrate the improved differentiation. The materials are much better differentiated than with the RGB and SVD staining techniques. One disadvantages of the variable filter method is that it tends to "lose" information for points that are not near any classified distribution. Note the black areas in the right image of Fig. 5.3. This method is also dependent on an interactive classification technique.

The fourth technique is similar to the variable filter technique. It uses the gaussian functions calculated in the previous section as probability distributions. Colors are interactively assigned to each gaussian. Figure 5.4 shows a human brain colored by this technique, together with the two-dimensional feature space colored using the same color map. This figure also describes how the images are colored using the distributions and the selected colors. The major advantages of this technique are that it avoids the interactive classification step and that it differentiates tissues with similar signal well. Its major disadvantage is shared with the variable filter technique: it "loses" data that is not near any of the gaussians in feature space. This can happen for materials that are very rare in an image or for samples that are very far from the mean of the material they represent, but is usually not critical for identifying materials that make up a significant portion of an object.

Figure 5.5 shows a similarly stained image of a slice through a human hand.

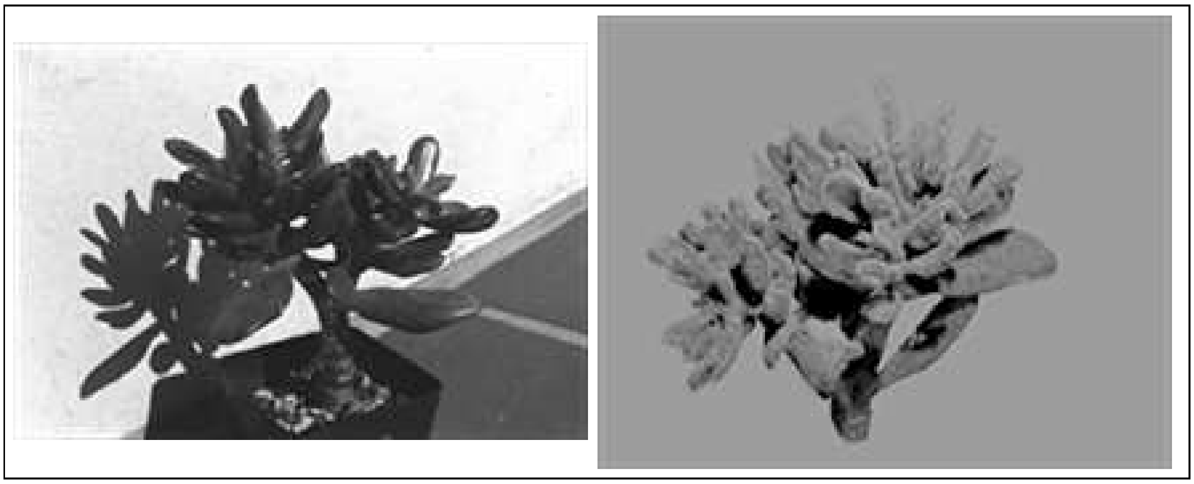

Figure 5.6: The left image shows a photograph of a jade plant. We collected magnetic resonance data of the plant, and volume rendered that data to create the image on the right. □

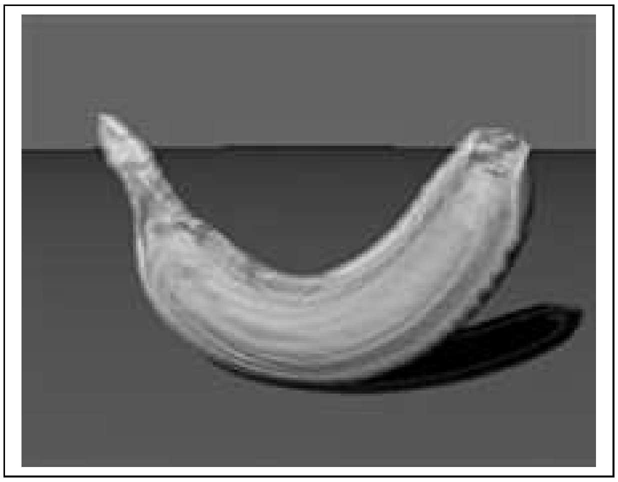

Figure 5.7: Volume-rendered image of banana with peel shown in transparent yellow and meat shown in opaque white. □

5.2 Applications to Volume Rendering

We have created initial volume-rendered images of some of objects we have scanned. Our volume renderer ray traces volume data, integrating through densities dependent on the amount and type of material present. See [Kajiya and von Herzen 84] and [Kay 92], among others, for more details on ray tracing volumetric data.

The first volume-rendered image we show is of data collected from a jade plant. The dataset is transparent where the data values are small and is progressively more opaque as the data values get larger. Note the correspondence between a photograph of the jade plant on the left of Fig. 5.6 and the volume-rendered image on the right.

The volume-rendered image in Fig. 5.7 is of data of a banana. Peel and flesh materials have been identified within the dataset, with the banana peel material rendered with a transparent yellow color, and the banana flesh with an opaque white.

Figure 5.8 shows two volume-rendered images of a human hand. The image on the left has

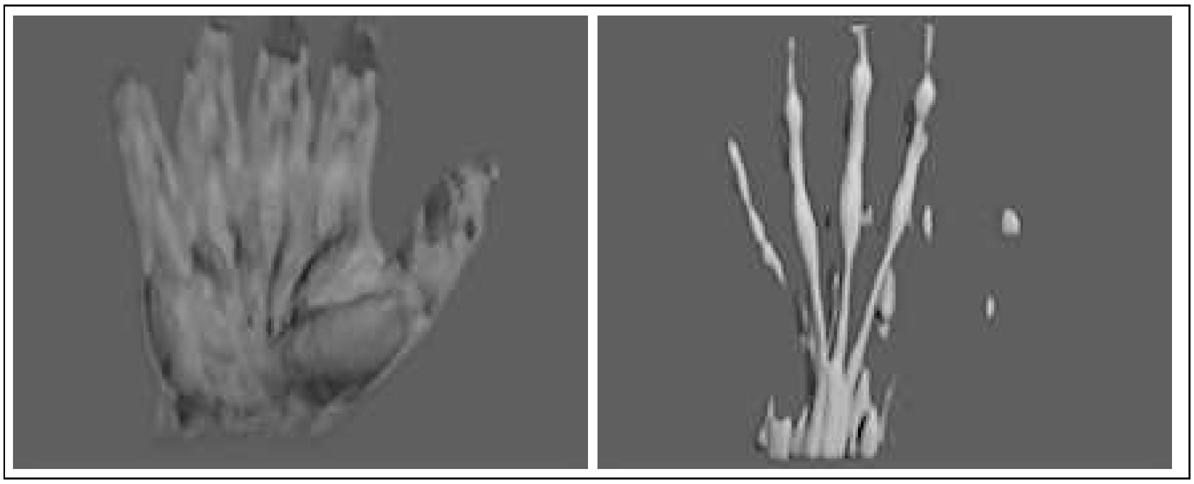

Figure 5.8: The volume-rendered image of a human hand on the left shows only muscle and fat, while the image on the right shows only tendon. □

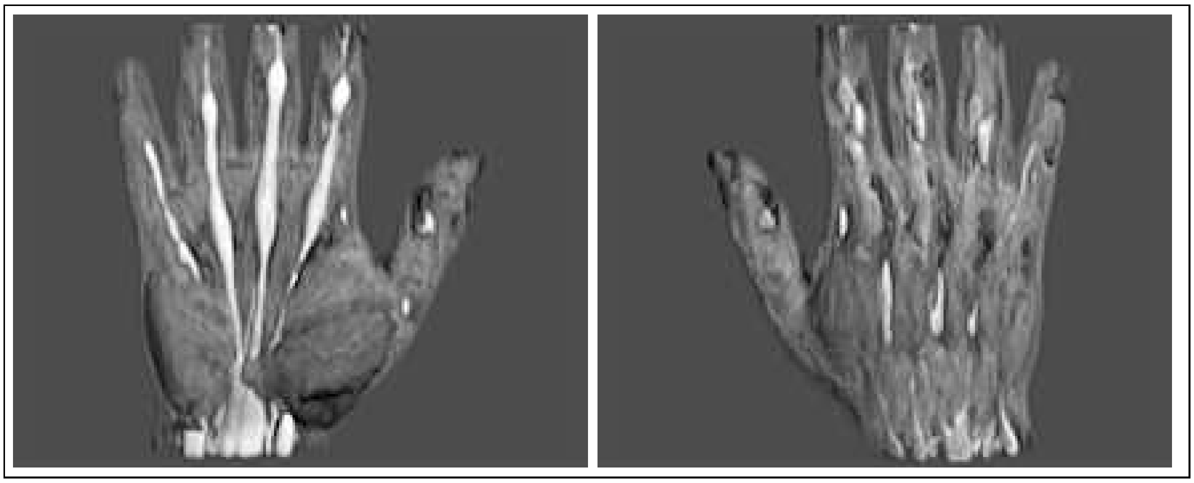

Figure 5.9: Volume-rendered images of human hand from the front and back, with muscle in red, fat in yellow, and tendon in white. □

opaque red material where the classified dataset indicates muscle, and semi-transparent yellow material where the classified dataset indicates fat. The volume-rendered image on the right is opaque white where the classified dataset indicates tendon.

Figure 5.9 shows volume-rendered images of the front and back of the same human hand, with the three materials combined.

5.3 Applications to Geometric Model Extraction

In this section we describe how we use the probabilities produced by the material classification algorithm in Chapter 4 to create preliminary geometric models. We first convert one scalar dataset out of our vector-valued dataset of materials probabilities into a continuous function by using tri-cubic b-spline interpolation. This corresponds to using a cubic gaussian approximation as a kernel for reconstructing the continuous function. A geometric model is then extracted by computing a polygonal approximation to an isosurface of this interpolation function.

The isosurface extraction is computed using the algorithm for implicit surface approximation described in [Snyder 91]. The algorithm requires a function that can bound the output of the interpolation function, given a bound on its input. Using interval arithmetic, this is relatively easy to implement for tri-cubic interpolation.

The algorithm subdivides the space containing the model in order to compute rectangular parallelepipeds that completely bound the iso-surface. The algorithm then computes the intersections of the isosurface with the edges of each region, and links the intersections into polygons.

In Fig. 5.10 we show an extracted geometric model of a jade plant on the left and separately extracted models of the peel and flesh of a banana on the right.

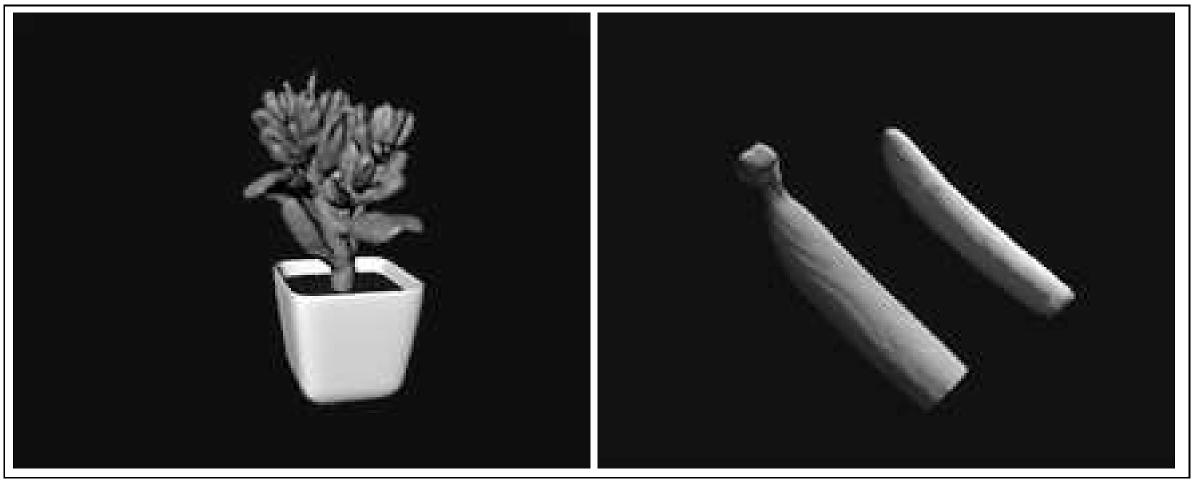

Figure 5.10: Geometric models extracted from volume data collected. A model of a jade plant is on the left. The model on the right is of banana skin and meat displayed separately. The banana model is cut off at the bottom because the real banana extended out of the uniform region of the MR coil. □

5.4 Applications to Dynamic Model Extraction

We have also extracted preliminary dynamic models from classified data. We use the same algorithm as in the previous section to create a polygonal isosurface, and in addition create a set of rectangular parallelepipeds that contain the isosurface and everything it encloses. By representing the vertices of these cubes as mass points connected by spring-dashpots we can dynamically simulate the models.

We simulate the motion of the flexible cubes, and then deform the polygonal model that they contain before rendering. Figure 5.11 shows, on the left, four frames of the dynamics of a simulation of a jade plant model twisted abruptly. On the right, the figure shows four frames of a simulation where the skin of the banana segment is peeled away from the flesh.

Figure 5.11: Geometric/dynamic models of a jade plant and a banana extracted from volume data. The left image shows four frames from an animation of the dynamics of a flexible model of a jade plant. The right image shows four frames from an animation of the dynamics of the skin being peeled off the banana. □

The classification process has helped to improve the extraction of dynamic models. Without the ability to distinguish between peel and flesh, for example, extracting a model of a banana that could be peeled would have been very difficult.

Chapter 6

Conclusion

6.1 Results

We have developed and presented techniques applicable to extracting models of real-world objects from volume data. Our techniques are primarily in the first steps of the model-extraction process: collecting volumetric magnetic resonance data, reducing artifacts in the data, and classifying different materials.

We have presented guidelines for choosing objects and for choosing magnetic resonance imaging collection parameters. While we advocate preventing artifacts in collected data when possible, we have also presented techniques for reducing aliasing artifacts from collected data, for correcting some causes of non-uniform response of a given tissue within a dataset, for registering datasets that are not aligned, and for improving contrast by calculating datasets for different values of the collection parameters.

Our unsupervised classification algorithm operates on the artifact-reduced data by creating a histogram of the data, incrementally fitting a sum of gaussians model to the histogram, and then using Bayes' law to calculate material probabilities.

From the processed data we have created preliminary static and dynamic models of objects, as well as computationally stained images of slices of the data and volume-rendered images of whole datasets.

6.2 Future Work

The motivation for this work has been to improve the quality of computer graphics models of real-world objects and to make the task of creating them more automated. Future work will ultimately lead to a system that will be able to create models of objects quickly. We would like to be able to take a real-world object, such as a cricket, a frog, a human face, or a human hand, and create a model of the muscles, bones, tendons, and other parts. The relationships and constraints between parts should be deduced from the data, or easily entered interactively. We would then like to be able to simulate these parts by contracting the muscles, which would pull on the tendons, which in turn would pull on the bones, moving them as they move in the real-world object.

Work in several areas is necessary to achieve the goals of a system for creating models of real-world objects. First, we need to collect higher-resolution data with better contrast between materials. We will be using data from the Beckman Imaging Center MR microscope, which is just becoming available. We would like to be able to computationally optimize the collection parameters that we use to improve the contrast, signal-to-noise ratio, collection time and other qualities of the collected data.

Second, the current classification process, which calculates material probabilities for a given voxel, should estimate material quantities in each voxel, and should take into account local geometric information to better determine what materials are in each voxel. Once data is classified this way, we can use it to make computationally stained images that will have smoother, more continuous appearance, and models with more accurate boundaries and relationships between parts.

Third, we would like to be able to volume-render models that have deformed as a result of simulations. Our volume-rendering technique is currently limited to rendering static volumes. Dynamic simulation can produce deformations of those volumes that represent, for example, a plant under some load, or a banana peel bending as it is peeled off. We would like to be able to produce volume-rendered images of these deformed objects in addition to the polygonal renderings we can currently produce.

Fourth, and finally, we would like to produce much more sophisticated and detailed static and dynamic models. We need algorithms which create dynamic models that are more accurately shaped than the cubes that we are currently using. We need to be able to extract models of "thin" object parts, like skin or tendons. We will need a more sophisticated testbed for simulating these flexible models with appropriate constraints, including constraints like a tendon attached to a muscle, skin lying above fat and sliding around, and flexible bodies colliding with one another.

Appendix A

Review of Digital Signal Processing

In this appendix we review some concepts from digital signal processing. Readers unfamiliar with the field should refer to a source such as [Oppenheim 83] or [Lim 90] for more detail, or [Blinn 89a] and [Blinn 89b] for a more intuitive description. We will *italicize* terms where they are first defined.

For our purposes digital signal processing consists of mapping from functions of continuous parameters to *sampled* functions and back again, as well as operating on those functions or the samples that represent them. We will first discuss the one-dimensional case, and generalize to higher dimensions in other sections. We start with a scalar function, $f(x)$:

$$f : \mathbf{R} \rightarrow \mathbf{R}.$$

Given a *kernel* function, $k(x)$,

$$k : \mathbf{R} \rightarrow \mathbf{R},$$

the *convolution transform* of $f(x)$ with $k(x)$ is defined as

$$s(x) = (f \circ k)(x) = \int_{-\infty}^{\infty} f(t)k(x-t)dt. \tag{A.1}$$

A *sample,* $s(x_0)$, of the function $f(x)$ at the point x_0 for the kernel function $k(x)$ is

$$s(x_0) = (f \circ k)(x_0) = \int_{-\infty}^{\infty} f(t)k(t-x_0)dt. \tag{A.2}$$

By sampling a function at n regularly spaced points, $x_i = x_0 + ih$, for $0 < i < n$ we produce a *dataset,* $\mathcal{D}(f) = \{f(x_i)\}$. We call h the *sample size.*

Using a kernel function, $k(x)$, we can create a new continuous function through a process called *reconstruction*, which is implemented as follows:

$$\hat{f}(x) = \sum_{i=1}^{n} f(x_i)k(x-x_i). \tag{A.3}$$

We write the *fourier transform* of $f(x)$ as $(\mathcal{FT}(f))(\omega)$. This new function is in *fourier space,* (also called *frequency space*) because each point represents the contribution of one angular frequency, ω, to the original function, $f(x)$. Analogously, $\mathcal{DFT}(\mathcal{D}(f))$ is the *discrete fourier transform* of a dataset, and each sample it contains represents a single frequency.

The fourier transform of a kernel function is its *frequency response.* The *Convolution Theorem,*

$$\mathcal{FT}(f \circ k) = \mathcal{FT}(f)\mathcal{FT}(k), \tag{A.4}$$

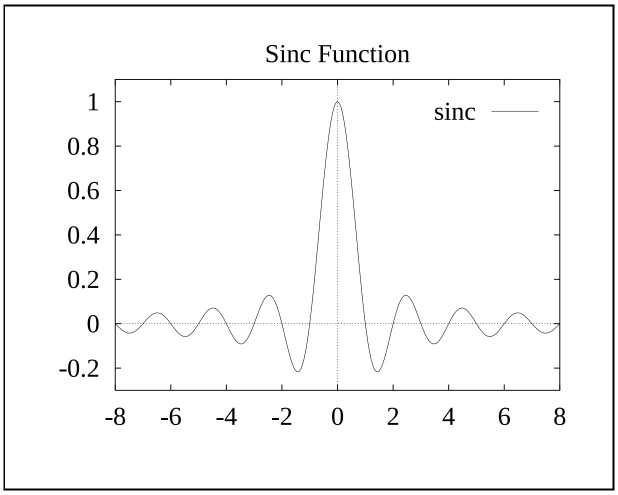

Figure A.1: Sinc kernel function $k_s(x) = \frac{sin(\pi x)}{\pi x}$. As the magnitude of the input of $k_s(x)$ gets arbitrarily large, the output of the function continues oscillating around zero. □

states that convolving with the function $k(x)$ attenuates each frequency component of the fourier transform of $f(x)$ by the corresponding value in the fourier transform, or frequency response, of function $k(x)$. Performing such a convolution is often called *filtering*, with the kernel function $k(x)$ known as the *filter kernel.*

As an example, the Dirac delta function, $\delta(x)$, which is zero everywhere except the origin, and which has an integral equal to 1, can be used as a kernel function. Assume, for this example, that $f(x)$ is a continuous function. For this kernel function, Eqn. A.2 evaluates to $f(x_0)$. Such sampling with a delta function kernel is known as *point sampling*, since it returns the function evaluated at a single point. The delta function has uniform frequency response for all frequencies, so convolving $f(x)$ with $\delta(x)$ does not modify $f(x)$. Creating a sampled dataset using a delta kernel function gives us a set of point samples. Reconstructing a function from those samples with the same delta kernel function gives us a comb-like function that is zero everywhere except for the sampling points; at those points it matches the original function.

If $f(x)$ is not continuous, the sample, convolution, and reconstruction operations will behave differently at any points of discontinuity, but will behave as described everywhere else.

A *sinc* kernel function, defined in [Oppenheim 83] as

$$k_s(x) = sinc(x) = \frac{sin(\pi x)}{\pi x}, \tag{A.5}$$

has some useful properties (Fig. A.1 and Eqn. A.5 show the *sinc* function for sample size $h = 1$). The frequency response of the *sinc* function is a box function, similar to that of Fig. A.2. Convolving a function with a *sinc* kernel function generates a new function that has no frequencies above a certain limit known as the *Nyquist limit.* This new function is referred to as *low-pass filtered.* Sampling a function using a *sinc* kernel and then reconstructing from those samples using a *sinc* kernel produces a low-pass-filtered version of the original function. Note that it is possible to exactly reconstruct the low-pass filtered function from the samples. The same samples can also be calculated by point sampling the low-pass filtered function.

Practically, the *sinc* function is difficult to use directly, either for sampling or for reconstruction, because its output continues oscillating around zero as its input gets arbitrarily large, and so a convolution requires integrating over all of **R**. The *sinc* function also tends to introduce *ring-*

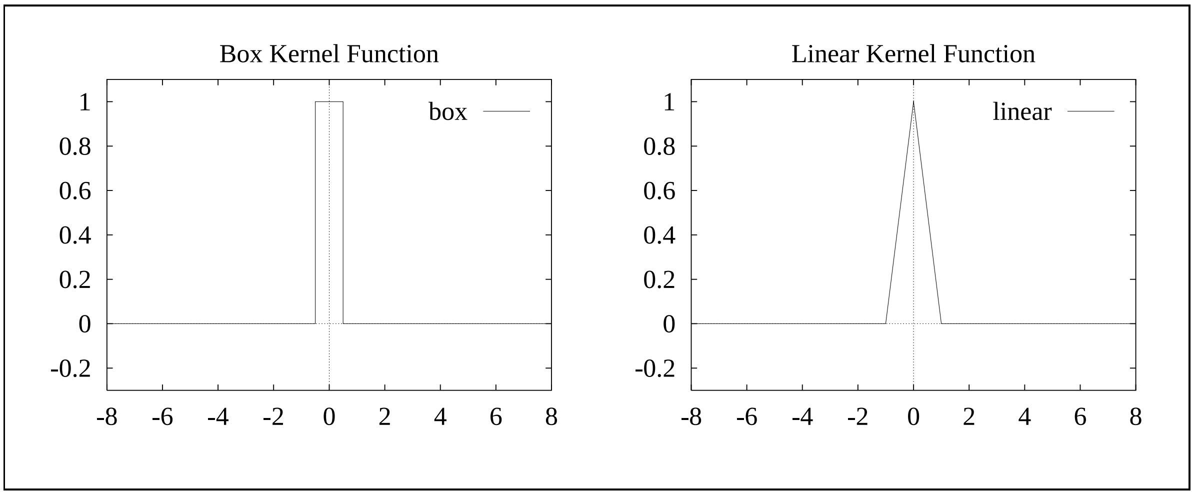

Figure A.2: Box and linear kernel functions $k_b(x)$ and $k_l(x)$. The box kernel function is zero outside the interval $[-\frac{1}{2}, \frac{1}{2}]$ and the linear kernel function is zero outside the interval $[-1, 1]$. $\square$

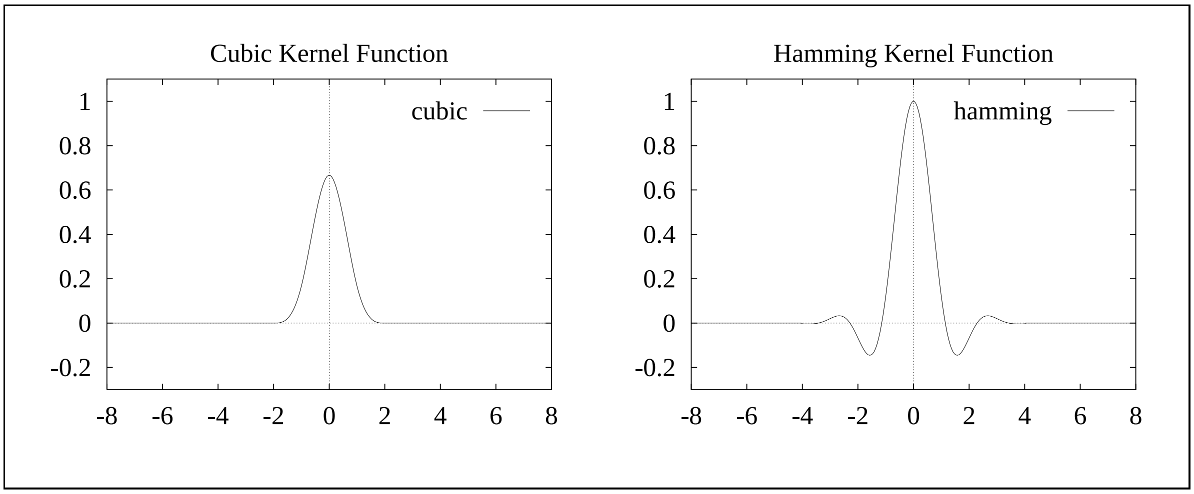

Figure A.3: Cubic and Hamming kernel functions, $k_c(x)$ and $k_h(x)$. The cubic kernel function is zero outside the interval $[-2, 2]$ and the Hamming kernel function is zero outside the interval $[-4, 4]$. $\square$

ing, or *Gibb's phenomenon*, when it is convolved with a function, because of the sharp cutoff of its frequency response. For these reasons, we approximate the *sinc* kernel function. Different approximations have different characteristics. Generally speaking, the approximations are *windowed* (are zero beyond a certain distance from their centers), which limits the range of a convolution integral. Wider approximations more faithfully replicate the frequency-limiting characteristics of the *sinc* function. Narrower approximations generally allow quicker calculations. Some of the approximations also tend to avoid the sharp frequency-domain cutoff of the *sinc* function.

Sampling kernel functions that pass frequencies above the Nyquist limit for a particular sampling rate will cause the samples to inaccurately *alias* those higher frequencies to lower frequencies. The artifacts caused are often referred to as *aliasing artifacts* and are difficult to remove because they cannot be distinguished from frequency information that is not aliased.

We reconstruct a function from our datasets using a variety of kernel functions, generalized to three dimensions. The first kernel function that we use is a box (see Fig. A.2), which effectively returns the value of the sample nearest to the point of evaluation. It is fast because it only depends on a single sample from the dataset, but it is inaccurate because it produces only a piecewise-constant approximation to the original function. The second fastest uses a linear kernel (see Fig. A.2), and returns a weighted average of the nearest $2^3 = 8$ neighbors. The third uses a cubic B-spline kernel (see Fig. A.3 and Eqn. A.6), which returns a weighted sum of the nearest $4^3 = 64$ neighbors.

$$k_c(x) = \frac{1}{6}\begin{cases} x^3 + 6x^2 + 12x + 8 & \text{if } x \in [-2, -1] \\ -3x^3 - 6x^2 + 4 & \text{if } x \in [-1, 0] \\ 3x^3 - 6x^2 + 4 & \text{if } x \in [0, 1] \\ -x^3 + 6x^2 - 12x + 8 & \text{if } x \in [1, 2] \\ 0 & \text{otherwise.} \end{cases} \tag{A.6}$$

The fourth uses an eight-sample-wide Hamming kernel function (see Fig. A.3 and Eqn. A.7), which returns a weighted sum of the nearest $8^3 = 512$ samples.

$$k_h(x) = sinc(\tfrac{7}{8}x)\begin{cases} 0.54 + 0.46cos(\frac{\pi}{4}x) & \text{if } -4 < x < 4 \\ 0 & \text{otherwise.} \end{cases} \tag{A.7}$$

Bibliography

[Barzel 92] Barzel, Ronen, “Physically-Based Modeling for Computer Graphics: A Structured Approach,” Academic Press, Boston, 1992 (in press).

[Blinn 89a] Blinn, James F., “What We Need Around Here Is More Aliasing,” *IEEE Computer Graphics and Applications*,9(1), January 1989, 75-79.

[Blinn 89b] Blinn, James F., “Return of the Jaggy,” *IEEE Computer Graphics and Applications*,9(2), March 1989, 82-89.

[Bradley 85] Bradley, William G., Jr., W. Ross Adey, and Anton N. Hasso, *Magnetic Resonance Imaging of the Brain, Head, and Neck,* Aspen Publishers, Rockville, Maryland, 1985.

[Cline 90] Cline, Harvey E., William E. Lorensen, Ron Kikinis, and Ferenc Jolesz, “Three-Dimensional Segmentation of MR Images of the Head Using Probability and Connectivity,” *Journal of Computer Assisted Tomography*, 14(6), November/December 1990, 1037-1045.

[Drebin et al. 88] Drebin, Robert, Loren Carpenter, and Pat Hanrahan, “Volume Rendering,” *Computer Graphics* (Proc. SIGGRAPH),vol. 22, 1988, 65-74.

[Duda and Hart 73] Duda, Richard P., and Peter E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, 1973.

[Foley et al. 90] Foley, James, Andries van Dam, Steven Feiner, and John Hughes, *Computer Graphics: Principles and Practice,* Addison-Wesley, Reading, Mass., 1990.

[Horwitz et al. 71] Horwitz, Harold M, Richard F. Nalepka, Peter D. Hyde, and James P. Morgenstern, “Estimating the Proportions of Objects within a Single Resolutions Element of a Multispectral Scanner,” *International Symposium on Remote Sensing of Environment*, (7th : 1971 : Ann Arbor, Michigan), 1307-1320.

[James 76] James, Glenn, ed., *Mathematics Dictionary*, Van Nostrand Reinhold Company Inc., New York, 1976.

[Kajiya and von Herzen 84] Kajiya, James T., and Brian P. Von Herzen, “Ray Tracing Volume Densities,” *Computer Graphics* (Proc. SIGGRAPH),vol. 18, 1984, 165-174.

[Kay 92] Kay, Timothy, *From Geometry to Texture: Experiments Towards Realism in Computer Graphics,* Ph. D. Thesis, California Institute of Technology, 1992.

[Keller 88] Keller, Paul, “Basic Principles of Magnetic Resonance Imaging,” GE Medical Systems Report, Milwaukee, 1988.

[Kikinis et al. 90] Kikinis, Ron, Rerenc A. Jolesz, Guido Gerig, Tamas Sandor, Harvey E. Cline, William E. Lorensen, Michael Halle, Stephen A. Benton, "3D Morphometric and Morphologic Information Derived From Clinical Brain MR Image," NATO Advance Research Workshop on 3D Imaging in Medicine, Springer-Verlag, 1990, 441-454.

[Kirk 92] Kirk, David B., Alan H. Barr, and David Laidlaw, "Registration and Computational Staining of MRI Data," Caltech Computer Science Technical Report, to appear.

[Kirk 90] Kirk, David B., "Representing Rough Surfaces and Volumes for Computer Graphics," M.S. Thesis, California Institute of Technology, 1990.

[Larson 82] Larson, Harold J., *Introduction to Probability Theory and Satistical Inference*, John Wiley and Sons, New York, 1982.

[Levoy 88] Levoy, Marc, "Display of Surfaces from Volume Data," *IEEE Computer Graphics and Applications*,vol. 8(3), May, 1988, 29-37.

[Lim 90] Lim, Jae S., *Two-dimensional Signal and Image Processing,* Prentice Hall, New Jersey, 1990.

[Lorensen and Cline 87] Lorensen, William, and Harvey Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *Computer Graphics* (Proc. SIGGRAPH),vol. 21, 1987, 163-169.

[Miller et al. 91] Miller, James, David Breen, William Lorensen, Robert O'Bara and Michael Wozny, "Geometrically Deformed Models: A Method for Extracting Closed Geometric Models from Volume Data," *Computer Graphics* (Proc. SIGGRAPH),vol. 25, 1991, 217-226.

[Moik 80] Moik, Johannes G., *Digital Processing of Remotely Sensed Images,* (NASA SP; 431), 1980.

[Muraki 91] Muraki, Shigeru, "Volumetric Shape Description of Range Data using 'Blobby Model'," *Computer Graphics* (Proc. SIGGRAPH),vol. 25, 1991, 227-235.

[NAG] NAG Fortran Library, Numerical Algorithms Group, Downers Grove, Illinois.

[Oppenheim 83] Oppenheim, Alan V., and Alan S. Willsky, with Ian T. Young, *Signals and Systems,* Prentice-Hall, New Jersey, 1983.

[Pizer 90] Pizer, S., "Toward Interactive Object Definition in 3D Scalar Images," *3D Imaging in Medicine*, NATO ASI Series F, vol. 60, Springer-Verlag, Berlin, 1990, 83-105.

[Platt and Barr 88] Platt, John, and Alan Barr, "Constraint Methods for Flexible Models," *Computer Graphics* (Proc. SIGGRAPH),vol. 22, 1988, 279-288.

[Siggraph Films 87] Siggraph Video Reels, vols. 36-37, 1987.

[Siggraph Films 88] Siggraph Video Reels, vols. 38-39, 1988.

[Siggraph Films 89] Siggraph Video Reels, vols. 51-52, 1989.

[Snyder 91] Snyder, John, *Generative Modeling: An Approach to High Level Shape Design for Computer Graphics and CAD,* Ph.D. Thesis, California Institute of Technology, 1991.

[Snyder 92] Snyder, John, *Generative Modeling for Computer Graphics and CAD,* Academic Press, Boston, 1992 (in press).

[Snyder 92a] Snyder, John, "Interval Analysis for Computer Graphics," submitted to Siggraph 92.

[Terzopoulos et al. 87] Terzopoulos, Demetri, John Platt, Alan Barr, and Kurt Fleischer, "Elastically Deformable Models," *Computer Graphics* (Proc. SIGGRAPH),vol. 21, 1987, 205-214.

[Terzopoulos and Fleischer 88] Terzopoulos, Demetri, and Kurt Fleischer, "Modeling Inelastic Deformation: Viscoelasticity, Plasticity, Fracture," *Computer Graphics* (Proc. SIGGRAPH),vol. 22, 1988, 269-278.

[Upson and Keeler 88] Upson, Craig, and Michael Keeler, "V-BUFFER: Visible Volume Rendering," *Computer Graphics* (Proc. SIGGRAPH),vol. 22, 1988, 59-64.

[Vannier et al. 85] Vannier, Michael W, Robert L. Butterfield, Douglas Jordan, William A. Murphy, Robert G. Levitt, Mokhtar Gado, "Multispectral Analysis of Magnetic Resonance Images," *Radiology*, 154, 1985, 221-224.

[Vannier et al. 88] Vannier, Michael W. Christopher M. Speidel, and Douglas L. Rickman, "Magnetic Resonance Imaging Multispectral Tissue Classification," Neural Information Processing Systems (NIPS), August 1988.

[Wehrli 88] Wehrli, Felix, "Advanced MR Imaging Techniques," GE Medical Systems Report, Milwaukee, 1988.

[Wehrli] Wehrli, Felix, "Introduction to Fast-Scan MR," GE Medical Systems Report, Milwaukee.

[Wiersma et al. 76] Wiersma, D. J. and D. Landgrebe, *The Use of Spatial Characteristics for the Improvement of Multispectral Classification of Remotely Sensed Data*, IEEE Symposium on Machine Processing of Remotely Sensed Data, 1976, 2A-18 2A-22.

[Yoo et al. 92] Yoo, Terry S., Ulrich Neumann, Henry Fuchs, Stephen M. Pizer, Tim Cullip, John Rhoades, Ross Whitaker, "Direct Visualization of Volume Data," *IEEE Computer Graphics and Applications*,vol. 12(4), July, 1992, 63-71.