SEMI-AUTOMATIC GENERATION OF TRANSFER FUNCTIONS FOR DIRECT VOLUME RENDERING

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

by

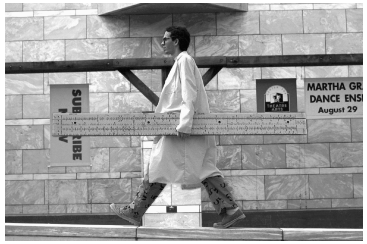
Gordon Lothar Kindlmann

January 1999

ABSTRACT

Finding appropriate transfer functions for direct volume rendering is a difficult problem because of the large amount of user experimentation typically involved. Ideally, the dataset being rendered should itself be able to suggest a transfer function which makes the important structures visible. We demonstrate that this is possible for a large class of scalar volume data, namely that where the region of interest is the boundary between different materials. A transfer function which makes boundaries readily visible can be generated from the relationship between three quantities: the data value and its first and second directional derivatives along the gradient direction. A data structure we term the *histogram volume* captures the relationship between these quantities throughout the volume in a position independent, computationally efficient fashion. We describe the theoretical importance of the quantities measured by the histogram volume, the implementation issues in its calculation, and a method for semi-automatic transfer function generation through its analysis. The techniques presented here make direct volume rendering easier to use, not only because there are much fewer variables for the user to adjust to find an informative rendering, but because using them is more intuitive then current interfaces for transfer function specification. Furthermore, the results are derived solely from the original dataset and its inherent patterns of values, without the introduction of any artificial structures or limitations. Examples with volume datasets from a variety of disciplines illustrate the generality and strength of the techniques.

Biographical Sketch



Gordon Kindlmann was born December 3, 1972 at Yale-New Haven Hospital in New Haven, Connecticut. Nineteen years later, he returned there for his first full-time job, in the Department of Laboratory Medicine, maintaining mainframe code and making it Y2K clean. In the intervening years he attended public schools in North Branford and Guilford, Connecticut. During summers, he played clarinet at the Point Counter Point and Tanglewood music programs. He attended Cornell University from 1991 to 1995, graduating with a BA in Mathematics. His freshman year stab at a physics major brought an appreciation of the power of symmetry arguments; his math major taught him the importance of diligence and consistency in academic work, qualities he is still striving for. Senior year he played drums in the hardcore band barcode. Between 1995 and 1997 he was an MS student in the Program of Computer Graphics at Cornell University, where his research on volume rendering was distracted and enriched by forays into color theory, computer vision, mathematical visualization, and theories of digital culture.

Dedicated to Professor Mark Scatterday

Just in case he ever doubts that his teaching and leadership do not profoundly effect the students he works with, this thesis is dedicated to Mark Scatterday, Chair of the Department of Music at Cornell University, and director of the wind ensembles in which I played clarinet during my two years at the Program of Computer Graphics. His expressivity, energy, and precision as a conductor inspired both a rededication to playing my instrument well, and a fresh realization of how important creating music is to me. When I overslept and dared walk into an hour-long dress rehearsal 45 minutes late, Professor Scatterday had the incomparable poise and trust to ask not "Where have you been?!", but simply, "Are you okay?" I can only hope that the amazing level of his care and devotion continues to leave as great an impression on his current students as it did on me.

Acknowledgements

It is not out of adherence to convention that I first acknowledge Donald Greenberg, Director of the Program of Computer Graphics. Quite simply, if not for him, my academic career would have finished with my undergraduate degree. It was he who saw past my colorful grades and sub-optimal GRE scores to someone bent on doing graphics research. The lab environment he created was the ideal place to explore and pursue my research goals. More than anyone else, I am indebted to him for fostering the beginning of what I hope will be a long career in academic research.

The lab owes its unique character to many other people as well, who directly or indirectly helped shape this work. James Durkin patiently provided assistance with mundane software problems, and whatever direction my day-to-day research had was largely thanks to repeated discussions with him. Officemates Jon Blocksom and Sing-Choong Foo were a source of endless motivation, energy, and amusement. Mitch Collinsworth and Hurf Sheldon kept the system infrastructure exciting and fun. Most importantly, the witty banter in the morning with staff members Ellen French and Linda Stephenson, and the janitor Mary Armstrong, made it worth while to stay up all night working.

My friends have endured an unconscionable amount of flakiness on my part

while grad school has consumed all too much of my time and attention. Cris Davis kept my priorities straight with frequent vegan cooking adventures, late night skateboarding, and heated discussions of all things scientific and ethical. Julian Myers arranged the soundtrack for this thesis by keeping me informed of new music to check out, and he continues to be a great example of how a successful grad student can keep school and real life in check. And despite my various lapses of both judgment and communication, Sheila Murphy has remained my best friend. Her kindness, supportiveness, and tolerance meant there was always someone to talk with. Only now coming to terms with how important this has been in maintaining my sanity, I hope to likewise repay her generosity in the coming years.

This thesis also owes its existence to my parents Peter and Marcia, since they are the ones who first taught me how to use my hands to make my ideas tangible. With my father's help they came to form as projects in wood and metal, like hand-wound DC motors; with my mother's help it was pottery and nourishing food. Now that this thesis is finally tangible, hopefully it can represent some fraction of the immense gratitude and love I have for them.

This work was supported in part by grants from the National Science Foundation (ASC 93-18180) to Mark Ellisman, Donald Greenberg, and Sid Karin, from the National Institutes of Health National Center for Research Resources (RR 04050) to Mark Ellisman, and by the NSF/ARPA Science and Technology Center for Computer Graphics and Scientific Visualization (ASC 89-20219). We gratefully acknowledge equipment grants from Hewlett-Packard, on whose workstations this research was conducted. The research presented here was also fueled in part by Collegetown Bagels, Shortstop Deli, and the GreenStar Cooperative Market.

It's noon. My friend's apartment floor isn't the most comfortable bed, but they don't let me sleep under my office desk anymore. Its been about 30 hours since I slept last; now falling asleep is hard because I'm stressed about managing to finish this thesis within the next few weeks. People are talking and laughing outside, do I know them? I've been away at University of Utah for a year; there are some old friends who might be living near here now.

Voices from outside mix with voices I hear upon starting to sleep, the sounds at the edges of dreams. These voices are of my friends, my life, all the people I've known here at Cornell. Even though I've been away at Utah for a year, I've been working too much to make many friends there. So it's mostly Cornell voices.

And it's noon. The middle of day, and I'm sleeping now, because besides working, that's all I do. The people outside are leaving now, I've missed them. And these voices I still hear, these voices I love, I miss them too, dearly. These voices I've slept through far too many times before, sleeping by day, working at night.

The voices I sleep to.

Table of Contents

1 Introduction **1**
1.1 The task of finding transfer functions . . . 1
1.2 Direct volume rendering of boundaries . . . 7
1.3 Relationship to edge detection, segmentation, and classification . . . 10

2 Previous work **14**

3 Mathematical foundations **19**
3.1 Boundary model . . . 19
3.2 Directional derivatives along the gradient . . . 21
3.3 Studying f, f', and f'' in a boundary . . . 23
3.4 The transformation from position domain to value domain . . . 25

4 Histogram volume calculation **31**
4.1 Histogram volume structure . . . 31
4.2 Histogram volume creation . . . 32
4.3 Implementation . . . 35
4.4 Histogram volume inspection . . . 40
4.4.1 Techniques for scatterplot visualization . . . 46

5 Opacity function generation **49**
5.1 Mathematical boundary analysis . . . 49
5.2 Opacity functions of data value . . . 53
5.3 Opacity functions of data value and gradient magnitude . . . 61
5.4 Comparison with Levoy's two dimensional opacity functions . . . 66
5.4.1 Visualizing isosurfaces in smoothly varying data . . . 66
5.4.2 Visualizing region boundary surfaces . . . 69

6 Results **72**
6.1 Turbine Blade: Effect of dataset blurring . . . 73
6.2 Engine Block: Effectiveness of two dimensional opacity functions . . . 78
6.3 CT Head: Effect of changing histogram volume size . . . 84
6.3.1 Comparison with renderings using Levoy's method . . . 88

6.4 Spiny Dendrite (hippocampus): Experimenting with g_{thresh} and $b(x)$ peak width for one dimensional opacity functions . . . 95
6.5 Spiny Dendrite (neostriatum): Comparison of one and two dimensional opacity functions, including Levoy's . . . 101

7 Conclusions and future work **109**

A Bandlimiting and the "thickness" of boundaries **114**
A.1 Bandlimiting and thickness in terms of σ . . . 114
A.2 Bandlimiting and thickness in the context of the Nyquist criterion . . . 117

B Generating masks for volume derivative measurement **120**
B.1 One dimensional masks from reconstruction filters . . . 120
B.2 Two common masks and a kernel which generates them . . . 123
B.3 Generalizing one dimensional masks . . . 127

C Evaluating the performance of the derivative measures **130**

Bibliography **140**

List of Figures

1.1 Opacity function demonstration for cylinder, engine, and dendrite . 3
1.2 Cornell Volume Rendering User Interface 5
1.3 Slice of neuron in tomographic plane 8
1.4 Comparison of volume rendering methods 9

3.1 Boundaries are step functions blurred by a Gaussian. 20
3.2 Isosurfaces tend to conform to object shape 21
3.3 ∇f is always normal to f's isosurfaces 23
3.4 Ideal boundary analysis 24
3.5 Relationships between f, f', f'' 26
3.6 f and f' versus position; three dimensional 27
3.7 f and f'' versus position; three dimensional 28
3.8 The underlying relationship of f, f', and f'' 30

4.1 Histogram volume structure 32
4.2 Sampling the boundary: from continuous to discrete 34
4.3 Using a mask to measure first derivative 39
4.4 Cross-section and scatterplots for cylinder 41
4.5 Cross-section and scatterplots for two material cylinder 42
4.6 Cross-section and scatterplots for turbine blade 43
4.7 Cross-section and scatterplots for engine block 44
4.8 Cross-section and scatterplots for neuron 45
4.9 Methods for scatterplot visualization 47

5.1 The error function, erf() 50
5.2 Relationship between thickness and the boundary function 51
5.3 Calculating $g(v)$ and $h(v)$ from the histogram volume 53
5.4 Precursors to $\alpha(v)$ calculation for nested spheres 55
5.5 Using $b(x)$ to control the position of the rendered boundary 58
5.6 Using $b(x)$ to control the character of the rendered boundary . . . 59
5.7 Calculating $h(v, g)$ from the histogram volume 62
5.8 Precursors to the $\alpha(v, g)$ calculation for nested cylindrical shells . . 63
5.9 Renderings of nested cylinders and corresponding $\alpha(v, g)$ 65
5.10 Slices of electron density dataset 67
5.11 Levoy's opacity function for isovalue surfaces 68

5.12 Levoy's opacity function for region boundary surfaces 71

6.1 Cubic spline used prior to down-sampling 74
6.2 Analysis of turbine blade dataset 75
6.3 Gaussian kernel used for blurring 76
6.4 Cross-section of blade showing struts 78
6.5 Rendering the turbine blade dataset 79
6.6 Renderings of engine block with one dimensional opacity functions 80
6.7 Renderings of engine block with two dimensional opacity functions 83
6.8 CT head analysis and rendering 86
6.9 CT head analysis and rendering using 64^3 histogram volume 87
6.10 Renderings of the CT head from Levoy's paper 89
6.11 Precursors to $\alpha(v, g)$ calculation for CT head 90
6.12 Renderings of CT head with two dimensional opacity functions . . 92
6.13 Rendering of CT head dataset displaying the air-bone boundary . . 94
6.14 Analysis of hippocampal neuron dataset 96
6.15 Renderings of hippocampal neuron dataset 99
6.16 Renderings of hippocampal neuron dataset, continued 100
6.17 Analysis of neostriatum neuron dataset 102
6.18 Renderings of neostriatum neuron dataset with one dimensional opacity functions 104
6.19 Renderings of neostriatum neuron dataset with two dimensional opacity functions 105
6.20 Rendering of neostriatum neuron dataset with Levoy's two dimensional opacity function 107

B.1 The quintic kernel $q(x)$ and its derivatives 126
B.2 Two examples of masks for volume derivative measurement 129

C.1 Renderings of two synthetic boundary volumes 132
C.2 Rotated ideal scatterplot curves 134
C.3 Derivative measure comparisons for thicknesses 0.33 to 2.66 135
C.4 Derivative measure comparisons for thicknesses 3.00 to 5.33 136
C.5 Derivative measure comparisons for thicknesses 5.66 to 8.00 137

Chapter 1

Introduction

1.1 The task of finding transfer functions

Direct volume rendering is a powerful technique for visualizing the structure of volume datasets. As devices for the acquisition of volume data are created and improved upon, the applications of direct volume rendering are growing from medicine and computational fluid dynamics to include astronomy, physics, meteorology, and geology. The most attractive aspect of direct volume rendering is actually its defining characteristic: it maps directly from the dataset to a rendered image without any intermediate geometric calculations. This is in contrast to traditional isosurface rendering, where the rendering step is proceeded by the calculation of an isosurface for a particular data value. Direct volume rendering can also avoid the time-consuming processes of segmentation and classification, because visualization can be done without any high-level information about the material content at each voxel.

Direct volume rendering is based on the premise that the data values in the volume are themselves a sufficient basis for creating an informative image. What

makes this possible is a mapping from the numbers which comprise the dataset to the optical properties that compose a rendered image, such as opacity and color. This critical role is performed by the transfer function. Because the transfer function is central to the direct volume rendering process, picking a transfer function appropriate for the dataset is essential to creating an informative, high-quality rendering.

The most general transfer functions are those that assign opacity, color, and emittance [LCN98]. However, much direct volume rendering uses transfer functions assigning only an opacity, with the color and intensity derived from simulated lights which illuminate the volume. For the purposes of this thesis, the term *opacity functions* is used to refer to this limited subset of transfer functions. Once the opacity function has been applied to the volume, the surfaces of the opaque regions are then shaded according to a simple shading model, such as the Phong model [Pho75]. Although this thesis explores opacity functions only, its results would be helpful in creating more general types of transfer functions as well.

In order to gain some intuition about the role opacity functions play in visualizing the structure of a volume dataset, it helps to look at opacity functions applied to two dimensional slices of sample datasets. Figure 1.1 shows the process for three datasets, with (from top to bottom) a slice of the dataset, a plot of the opacity function, the result of applying the opacity function to the slice, and an image rendered using the shown opacity function. The dataset analyzed in the left column is a synthetic dataset of a cylinder, which will be used repeatedly for illustrative purposes. In the middle column is a computed tomography (CT) scan of an engine block[1]. The last column shows one of the neuron datasets from the

[1] From the Chapel Hill Volume Rendering Test Datasets, Volume II

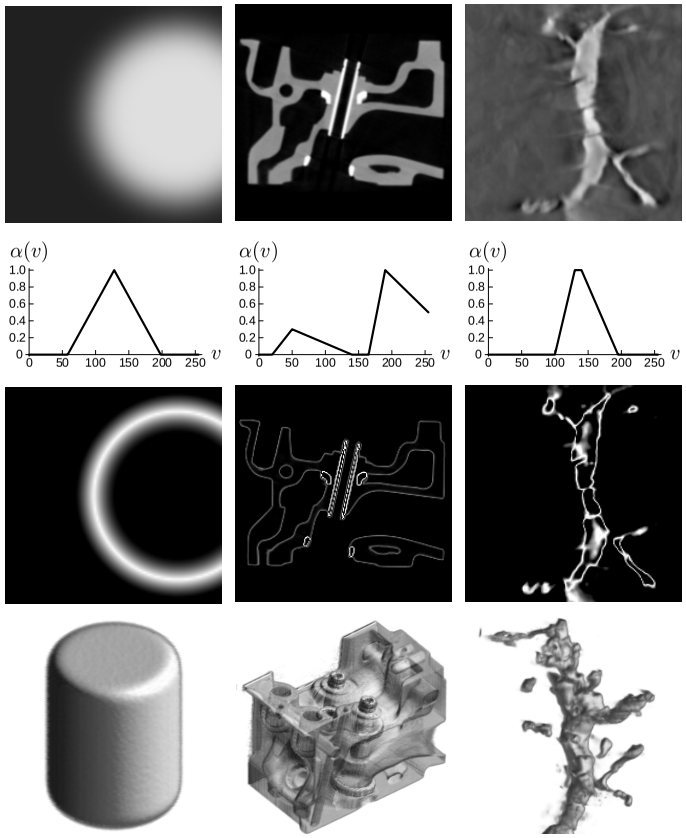


Figure 1.1: Opacity function demonstration for (from left to right) synthetic cylinder, CT engine block, and EM dendrite. Going from top to bottom: a slice of the dataset, a plot of the opacity function which assigns opacity α according to data value v, the result of applying the opacity function to the slice, and an image rendered using the shown opacity function.

Collaboratory for Microscopic Digital Anatomy (CMDA), in which the Program of Computer Graphics is a research participant [EGLea96]. The neuron data was tomographically generated from a sequence of transmission electron microscopy (EM) images[2].

Note that in each of these cases, the primary purpose of the opacity functions is to make opaque those values that occur around the boundaries of objects. This allows the shape of the objects to be seen in the rendered image without being obscured by the surrounding medium. If the opacity function makes these boundaries partially transparent, we can look through outer surfaces in the object (such as the engine block) to see internal structure. Since the opacity function is applied to the whole volume in a position-independent way, volume rendering relies on the premise that the interesting structural components of the volume, such as boundaries, have a reliable correlation with data value. If this is the case, then there should be an opacity function which produces an informative rendering.

The tools used to find opacity functions and render images might look something like Figure 1.2, which shows the direct volume rendering interface that was built for the CMDA project[3]. The user sets the viewpoint in window (a), which provides feedback with a wireframe box indicating the orientation and aspect ratio of the volume. The rendered image appears in window (b). Not shown are the interface for setting the position and color of the lights, and a tool for inspecting two dimensional slices of the data. The tool used to set the opacity function is shown in window (c). The user sets the number and location of control points

[2]Dataset courtesy of National Center for Microscopy and Imaging Research; specimen courtesy of Prof. K. Hama of the National Institute for Physiological Sciences, Okazaki, Japan

[3]The interface widgets are created with Tcl/Tk [Ous94], and the actual rendering is performed by the Stanford VolPack volume a rendering library [LL94]

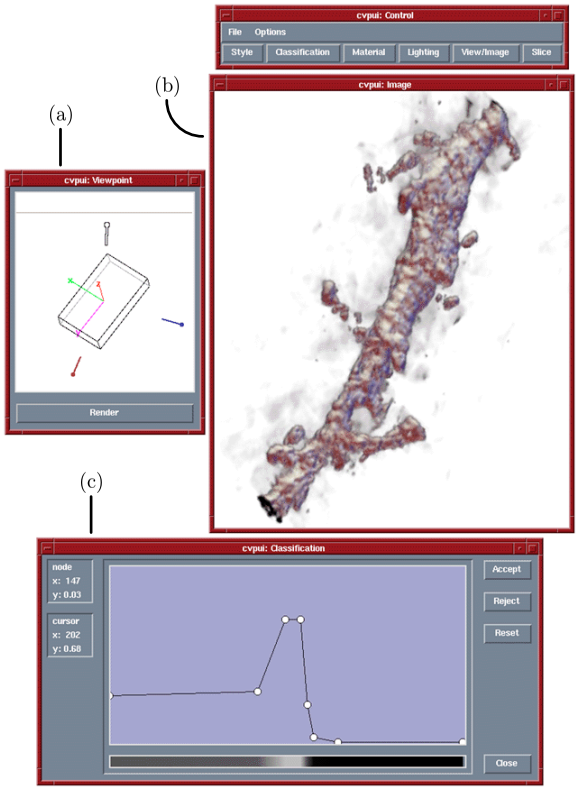


Figure 1.2: Cornell Volume Rendering User Interface. The viewpoint is set in window (a), and the rendered result is shown in window (b). In window (c) is the tool for adjusting the opacity function. The user sets the opacity function by directly editing its graph, a series of linear ramps joining adjustable control points. Data value is on the horizontal axis, and opacity is on the vertical.

which define an opacity function as a series of linear ramps.

In practice, a frustrating amount of time is spent adjusting the location of these points to make the image look "right". The general problem of *transfer* function specification is very difficult, but even simply finding a good *opacity* function is challenging. To make a high-quality rendering, an opacity function must satisfy at least two guidelines:

- It has to be correctly localized with respect to data value. That is, if the boundary is always associated with some value v, the opacity function should give v (and not a high or lower value) maximum opacity.
- The opacity for the remaining data values should be assigned in such a way that minimizes the creation of misleading artifacts in the rendering. For instance, if too few values are made opaque, the surface may appear broken or jagged.

Unfortunately, satisfying these constraints is an unintuitive task. Users looking at slices of the dataset can easily spatially locate features of interest, but finding data values which consistently correspond to the features of interest throughout the volume is not straight-forward. The second guideline is related to the fact that every direct volume rendering technique has strengths and weaknesses, and poorly designed opacity functions can reveal the weaknesses. Again, the selection of opacity functions is not intuitive, so typically the user searches for an opacity function by trial and error, adjusting it and repeatedly re-rendering to see the effects on the rendered output.

1.2 Direct volume rendering of boundaries

The question could be asked- "If the goal is to render the boundaries of objects, why is direct volume rendering being used, as opposed to isosurface rendering?" The standard argument for direct volume rendering as opposed to isosurface rendering is that in the former, "every voxel contributes to the final image", while in the isosurface rendering, only a small fraction of voxels (those containing the isovalue) contribute to the final image [Lev88]. This argument is not very convincing, since there is no consistent correlation between the quality of the image and the number of voxels contributing to its formation. Too many opaque voxels result in a cloudy image, or an image where an interesting portion of the structure is unintentionally obscured. It is just as important that the transfer function leave some parts of the volume transparent as it is that it make the interesting parts opaque; otherwise the volume rendered image would provide no additional insight into the structure of the volume.

The basic benefit of direct volume rendering over isosurface rendering is that it provides much greater flexibility in determining how the voxels contribute to the final image. Voxels over a range of values can all contribute to the image, with varying amounts of importance, depending on the transfer function. Also, while an isosurface can only show structure based solely on data value, the transfer function can do so based on other quantities as well, such as gradient magnitude.

This motivates why direct volume rendering can be used in situations where the structure of the data is amorphous, as in gaseous simulations [Max95]. More importantly, it motivates the use of direct volume rendering in medical imaging situations where there noise or measurement artifacts distort the isosurfaces away from the shape of the object boundary. To the extent that the objects' surfaces are

associated with a range of values, the transfer function for direct volume rendering can make a range of values opaque or translucent.

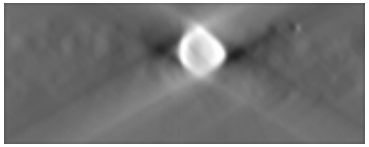
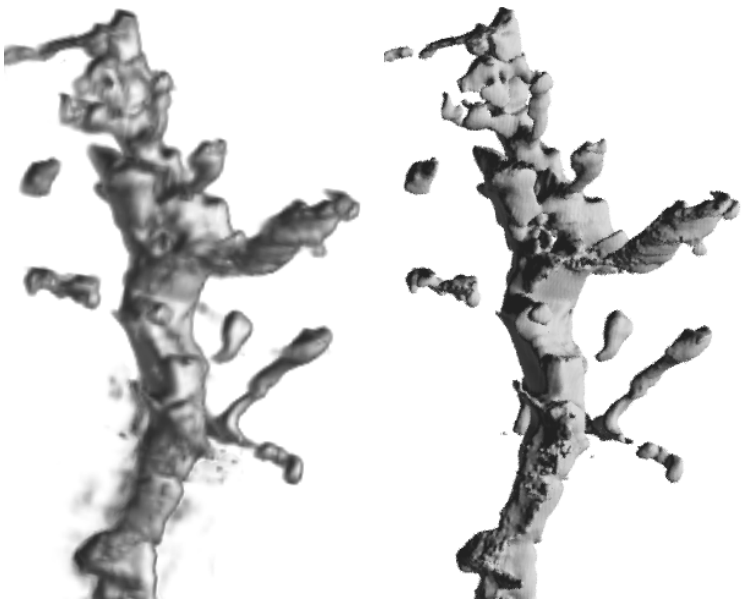


Figure 1.3: Slice of neuron in tomographic plane. Artifacts from the lack of projection data at some angles are visible as the bright spots on either side of the dendrite, as well as the light streaks.

As an example to exhibit the usefulness of direct volume rendering versus isosurface rendering, consider some neuron data from the CMDA project. Figure 1.3 shows a slice of a spiny dendrite dataset. Note the dark regions on either side of the neuron, and the light streaks which emanate from its top and bottom. These are artifacts from the tomographic process which reconstructs three dimensional information from a series of two dimensional projections. Because there are ranges of angles in which projection data cannot be obtained, there are orientations for which the quality of the tomographic reconstruction is poor, causing the surface of the neuron to be blurred or distorted. A further difficulty is the fact that the radio-opaque dye which renders the neuron visible is sometimes absorbed unevenly. Fig. 1.4 shows two renderings of a mammalian neuron dataset, using the same viewing angle, shading, and lighting parameters, but rendered with different algorithms: a shear-warp direct volume rendering produced with the Stanford VolPack rendering library[LL94] and a non-polygonal ray-cast isosurface rendering. Towards the bottom of the direct volume rendered image, there is some fogginess surrounding the surface, and the surface itself is not very clear. As can be con-



(a) Direct volume rendered (b) Isosurface rendered

Figure 1.4: Comparison of volume rendering methods

firmed by looking directly at slices of the raw data, this corresponds exactly to a region of the dataset where the material boundary is in fact poorly defined. The surface rendering, however, shows as distinct a surface here as everywhere else, and in this case the poor surface definition in the data is manifested as a region of rough texture. This can be misleading, as there is no way to know from this rendering alone that the rough texture is due to measurement artifacts, and not a feature on the dendrite itself.

1.3 Relationship to edge detection, segmentation, and classification

The over-all goal of this work is to allow the user of a direct volume rendering system to visualize the structure of a volume more easily. This thesis presents methods to automatically detect, and help visualize, the boundaries of objects in the volume. With this goal, it may seem that the task is similar to that of edge detection in computer vision, or that of segmentation or classification in medical imaging. However, there are significant differences. Each of these comparisons are considered in turn.

Edge detection methods are fundamental to computer vision, because edge detection is often the first stage of a lengthy image interpretation process. The goal of edge detection is to locate the pixels in the image that correspond to the edges of the objects seen in the image. This is usually done with a first and/or second derivative measurement following by a test which marks the pixel as either belonging to an edge or not[Fau93]. The result is a binary image which contains only the detected edge pixels. However, for the purposes of opacity function generation, the

spatial location or orientation of the edges is not relevant. The opacity function we apply to the volume will be applied everywhere the same way: it is a single function that assigns opacity with no respect to position. As was demonstrated in Figure 1.1, our goal is to locate boundaries not in the *spatial* domain, but in the *data value* domain.

Another difference between opacity function generation and the task of edge detection is that volume visualization works with a more limited class of data. In order for a dataset to be visualized by direct volume rendering, there needs to be some correlation between the data value and the interesting structure, otherwise no transfer function will be successful. On the other hand, this is not a necessary restriction for edge detection, since the inevitable variations in illumination mean that individual objects (and their edges) take on wide ranges of values. While these kinds of intensity fluctuations are naturally handled in edge detection, direct volume rendering of a volume dataset with similar data value variations would be very difficult.

Segmentation is the process of distinguishing objects in the dataset from their surroundings so as to facilitate the creation of geometric models. For example, in medical imaging it is often important to measure the shape, surface area, or volume of tissues in the body. Once the dataset is segmented, those quantities are easily measured. Segmentation is an active area of ongoing research. Manual segmentation is a slow and laborious process requiring expert knowledge, often involving the tracing of object outlines in a series of dataset slices. On the other hand, automated techniques are rarely robust and reliable, often requiring extensive supervision[WGKJ96].

On a very general level, both segmentation and the problem of transfer function

specification are faced with having to distinguish between the major structural components in a dataset. But these two tasks differ fundamentally in their result. Segmentation leads to a three dimensional *model* based on volume data, while a transfer function is just one parameter of the direct volume rendering process, which creates simply an *image*. Both processes are important. Useful science can be accomplished without segmentation, and the CMDA project is an excellent example. The biologists may eventually go through the work of segmenting their datasets, but they can learn about the neurons' size and structure from direct volume renderings of unsegmented datasets.

Closely related to segmentation is the process of classification, which assigns a material occupancy to each voxel based on any of a wide variety of data characteristics, such as data value (scalar or vector), derivative measures, or local histograms. The material occupancy assignment so created is called a "classification function". The classification function is often binary (either a voxel is wholly material x or it contains no material x), though more continuous classification functions better support the possible later steps of modeling and rendering[Lai95]. Thus, the classification function is like an opacity function in that it maps from the data space to a number between 0.0 and 1.0 in a position independent way.

Since the goal in direct volume rendering is to make objects in the volume visible, one obvious approach would be to make the interior of the objects opaque. Thus, one could simply use a classification function for the opacity function. Indeed, some authors used these terms synonymously[Lev88]. However, the speed of many rendering algorithms (such as ray tracing [Nov94] or the run-length encoding used by Lacroute [LL94]) varies with the number of voxels which contain opacity. In this situation it may be better for the opacity function to keep interior

voxels transparent, though that is precisely where the classification function would be highest. In this light, material classification functions do not make very good opacity functions. Accordingly, the methods in this thesis seek to find opacity functions which make the objects' *boundaries* opaque, not their interiors.

Chapter 2

Previous work

The canonical references for direct volume rendering of scalar fields are Levoy [Lev88] and Drebin et al.[DCH88]. Levoy presents two transfer function designs for visualizing two distinct classes of data. One transfer function is for visualizing isovalue contours in smoothly varying data; the other is for displaying boundaries in datasets containing adjoining regions of relatively constant value. Both methods assign opacity as functions of data value and gradient magnitude. Drebin et al. use a transfer function which sets opacity as a function of data value alone, and strives to differentiate between materials by assigning different colors to separate ranges of data values. Today, most transfer functions used in volume rendering are still set according to the methods presented by these two authors. Examples of recent work employing Levoy-style transfer functions is described in [Mur95] and [YESK95]. An example of transfer functions based on data value alone is described in [PMea96].

Direct volume rendering research today tends to focus either on making the rendering algorithms faster, or extending existing rendering methods to work with a wider variety of data sets. Exciting developments include the shear warp fac-

torization [LL94], and the use of texture maps for rendering at interactive rates on machines with 3-D texture memory [GK96]. Examples of existing techniques applied to a new domain are the extension of splatting [Wes90] to non-rectilinear volumes [Mao96], and using ray casting for vector field visualization [Fru96]. Often, direct volume rendering is not the tool of choice for visualization of scalar fields, and isosurface rendering is used instead, as in geology [PBL96] or computational medicine [SJM95]. The generation of polygonal isosurfaces is still an area of active research [IYK96, SHLJ96].

To date there has been little research into the generation of transfer functions for direct volume rendering. Two recent papers describe methods for assisting the user in the exploration of possible transfer functions. The first method [HHKP96] uses genetic algorithms to "breed" a good transfer function for the dataset in question. The system randomly generates a set of transfer functions, and renders small images for each one. Presented with this set of renderings, the user then picks the few renderings that seem to best display the volume data, and a new population of transfer functions is stochastically generated based on those that the user picked. This process iterates until the user feels that the best transfer function has been found. Alternately, a image-processing metric like entropy, variance, or energy is used as an objective fitness function to evaluate the rendered images without human guidance, and the process eventually converges on a transfer function which maximizes the fitness function. The method succeeds in generating good renderings, and frees the user from having to edit the transfer function manually.

The second method [MABea97] addresses the problem of "parameter tweaking" in general, with applications to light placement for rendering, motion control for articulated figure animation, as well as transfer functions in direct volume rendering. Here, the goal is not to find the one best parameter setting, but to find as wide a variety of parameter settings as possible, relying on a user-specified metric to determine similarity between rendered images. The system randomly generates a very large set of transfer functions, and then formats small thumbnail renderings resulting from these transfer functions into a two dimensional arrangement called a "design gallery". The user peruses these thumbnail images, selecting the most appealing rendering. When applied to the problem of volume rendering, this method also seems to find reasonable transfer functions, but like the genetic algorithm method, it is at the cost of completely divorcing the user from first-hand experimentation with transfer functions.

It is important to identify the idea behind both of these approaches. Instead of interacting with the parameter space of transfer functions directly, the user explores it indirectly, seeing only the results of the parameter setting on the rendering. Rather than creating a better interface for transfer function editing, the methods avoid the issue entirely by having the user pick transfer functions based on the resulting rendered images. In both cases, the process which generates the transfer functions is random, so the user is resigned to choose among those presented, and the quality of the final transfer function is always uncertain. The genetic algorithm method is not guaranteed to find a good transfer function, regardless of how long the user "breeds" them. The design gallery algorithm may or may not happen upon a good setting in its initial random generation of transfer functions, and the user would have to start the process over from the beginning to try to find better settings. Significantly, in both cases, the generated transfer functions are not at all constrained by any measured properties of the data.

It should also be pointed out that both methods only generate good transfer

functions for renderings *from one particular viewpoint*. If the viewpoint changes, the genetic algorithm will iterate differently, and the layout algorithm for the design gallery may generate completely different results. Because of this, it could be said that these two methods are not so much for finding good transfer functions as they are for finding good *renderings*.

While the two methods above could be termed "image-driven", the approach described in this thesis is "data-driven". It applies solely to those domains of volume rendering where the user's goal is to visualize the surfaces of materials represented in the volume. The method for transfer function generation is entirely derived from that goal. In this sense, the method described here has a level of validity or correctness not attained by either of the previous methods.

Although the previous work described above on the specific problem of finding transfer functions for direct volume rendering is not suitable for our needs, there is other work on related problems that is relevant. Bergman et al. [BRT95] address the problem of colormap selection in visualizing two dimensional scalar fields, to aid in the production of "false color" images. Because the human perceptual system perceives variations in hue differently than variations in intensity, the authors have developed a tool which chooses a colormap based on the user-specified goal of the visualization task, as well as the data's spatial frequency characteristics. More recently, Bajaj et al. [BPS97] describe a tool for assisting the user in selecting isovalues for effective isosurface volume visualizations of unstructured triangular meshes for isosurface rendering. By exploiting the mathematical properties of the mesh, important measures of an isosurface such as surface area and mean gradient magnitude can be computed with great efficiency, and the results of these measurements are integrated into the same interface which is used to set the isovalue.

Because the interface provides a compact visual representation of the metrics evaluated over the range of possible isovalues, the user can readily decide, based on their rendering goals, which isolevel to use. The importance to this thesis is that the contour spectrum is a good example of how an interface can use measured properties of the data to guide the user through the parameter space controlling the rendering.

Chapter 3

Mathematical foundations

3.1 Boundary model

The methods presented in this thesis make the task of volume rendering surfaces more intuitive and less labor intensive than previous methods. The algorithm is based on the computation of boundary characteristics in a volume and using this information to guide the user towards opacity function settings which readily display the boundaries. Towards that end, a mathematical model has been chosen for what constitutes an ideal boundary. We now describe the boundary model upon which the techniques of this thesis are based.

We assume that at their boundary, objects have a sharp, discontinuous change in the physical property measured by the values in the dataset. In MRI this property might be density of hydrogen nuclei, in CT and EM it is radio-opacity of the tissue. Also, we assume that the data is band-limited prior to sampling, but not with the usual Nyquist criterion. Rather, the measurement process is assumed to have a Gaussian frequency response, which effectively blurs the sharp edges with a Gaussian filter. Figure 3.1 shows a step function representing an ideal boundary

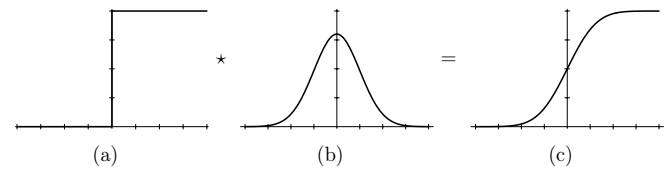


Figure 3.1: The ideal physical boundary (a) is convolved with a Gaussian filter (b) representing the frequency response of the measurement process, producing the measured boundary prior to sampling (c) which exhibits the shape of the erf() function.

prior to measurement, the Gaussian filter which performs the band-limiting by blurring, and the resulting measured boundary (prior to sampling). The resulting curve (Figure 3.1(c)) happens to be the integral of the Gaussian kernel, which is called the *error function*, notated "erf()"[KK68]. Actual measurement devices necessarily band-limit, so they always blur boundaries somewhat. Their frequency response is never exactly a Gaussian function, however, since this requires infinite support, and infinitely high frequencies cannot be measured. Although certain mathematical properties of the Gaussian function are exploited later, the inexact match of real-world sampling to the Gaussian ideal has not been found to severely limit application of the techniques presented here. A final assumption made for the purposes of this analysis is that the blurring is isotropic, that is, uniform in all directions. Again, the algorithms presented in this thesis will often work even if a given dataset doesn't have this characteristic, but results may be improved if it is pre-processed to approximate isotropic blurring. Having previously assumed that objects are made of material which attains the same data value everywhere in their interior, this is effectively equivalent to assuming that the boundaries of objects will be uniform over their entire surface. The consequences of these various

assumptions will be evident when histogram volume calculation is discussed in Chapter 4.

3.2 Directional derivatives along the gradient

Although it was suggested in Section 1.2 that isosurfaces are not always sufficient for visualizing objects in real world volume data, the method presented in this thesis still indirectly employs them as an indicator of object shape. That is, based

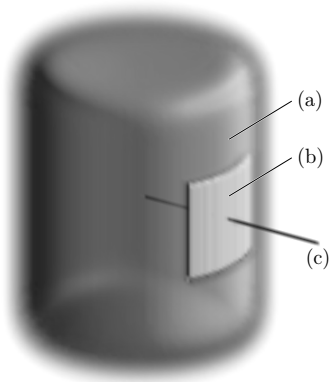
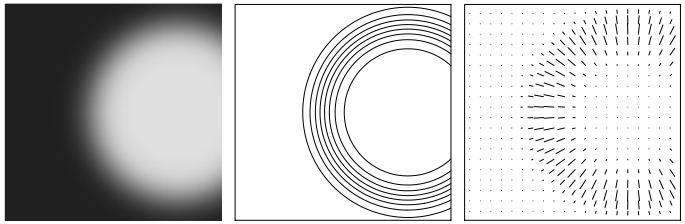


Figure 3.2: Isosurfaces tend to conform to object shape. For a cylindrical object (a) with a soft boundary, a small patch of an isosurface (b) is shown within the boundary, together with a line segment (c) which follows the gradient direction, perpendicular to the isosurface patch and to the object boundary.

on the mathematical property that the gradient vector at some position always points perpendicular to an isosurface through that position, we use the gradient vector as a way of finding the direction which passes perpendicularly through the

object boundary. Figure 3.2 demonstrates, for a simple cylindrical object with a soft boundary, how a line segment along the gradient direction lies perpendicular to a patch of the boundary isosurface, and how the isosurface patch conforms to the shape of the boundary. However, in real datasets, the isosurfaces don't always perfectly conform to the local shape of the underlying object. But on average, over the whole volume, the gradient vector does tend to point perpendicular to the object boundary. Therefore, our approach will be to rely on the statistical properties of the histogram to provide the overall picture of the boundary characteristics throughout the entire volume.

The directional derivative of a scalar field f along a vector $\mathbf{v}$, denoted $\mathbf{D}_{\mathbf{v}}f$, is the derivative of f as one moves along a straight path in the $\mathbf{v}$ direction. This thesis studies f and its derivatives along a path which is normal to the object boundary — moving along the gradient direction — in order to create an opacity function. Because the direction along which we're computing the directional derivative is *always* that of the gradient, we employ a mild abuse of notation, using f' and f'' to signify the first and second directional derivative along the gradient direction, even though these would be more properly denoted by $\mathbf{D}_{\widehat{\nabla f}}f$ and $\mathbf{D}^2_{\widehat{\nabla f}}f$, where $\widehat{\nabla f}$ is the gradient direction. We treat f as if it were a function of *just one variable*, keeping in mind that the axis along which we analyze f always follows $\widehat{\nabla f}$, which constantly changes orientation depending on position. Figure 3.3 shows how the gradient direction changes with position to stay normal to the isosurfaces of a simple object.



(a) $f(\mathbf{x})$ (b) Isosurfaces of f (c) ∇f

Figure 3.3: The gradient direction ∇f (c) changes with position in the data (a), but it maintains its perpendicular orientation with respect to the isosurfaces (b).

3.3 Studying f, f', and f'' in a boundary

Even though the first and second directional derivatives are quantities defined in three dimensions, the significant relationship between them can be reduced to a one dimensional case, because we only care about the value and its derivatives as we move along the gradient direction. Thus, in analyzing the relationship between f, f', and f'', it suffices to study a one dimensional sampling perpendicular to the surface. Figure 3.4 analyzes one segment of a slice from the synthetic cylinder dataset. Shown are plots of the data value, and the first and second derivatives, as we move across the cylinder's boundary. The direction of the gradient obviously varies throughout the volume, but the observed relationship between f and its directional derivatives is constant, because the boundary is assumed uniform everywhere. Note that at the mid-point of the boundary, the first derivative is at a maximum, and the second derivative has a zero-crossing. Because of blurring, the boundary is spread over a range of positions, but the maximum in f' and/or the zero-crossing in f'' provides a way to define an exact spatial location of a bound-

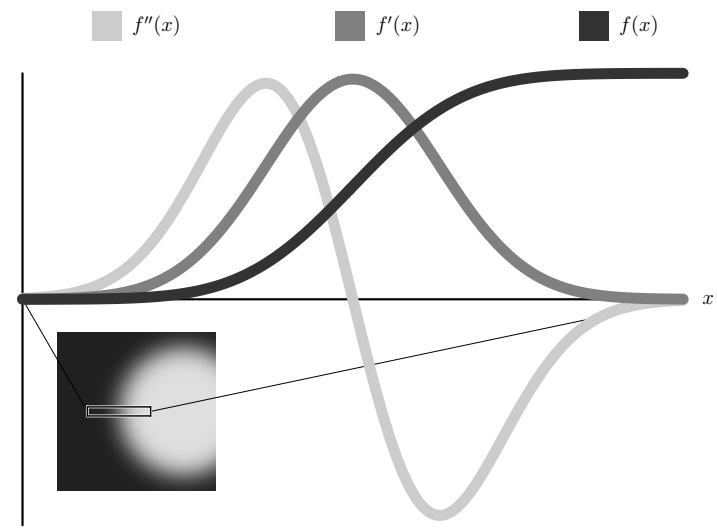


Figure 3.4: Ideal boundary analysis. The data value and its first and second derivatives are sampled along a short segment which passes through an object boundary

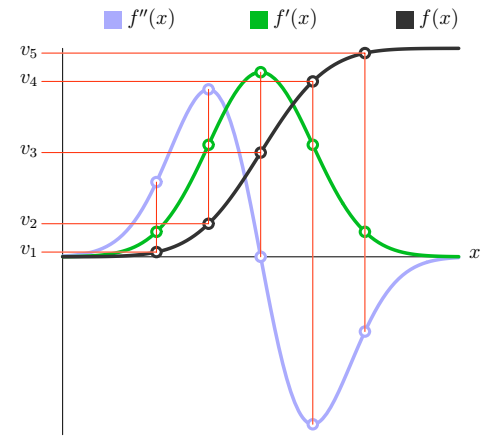
ary. Indeed, two edge detectors common in computer vision, Canny [Can86] and Marr-Hildreth [MH80], use the f' and f'' criteria, respectively, to find edges.

3.4 The transformation from position domain to value domain

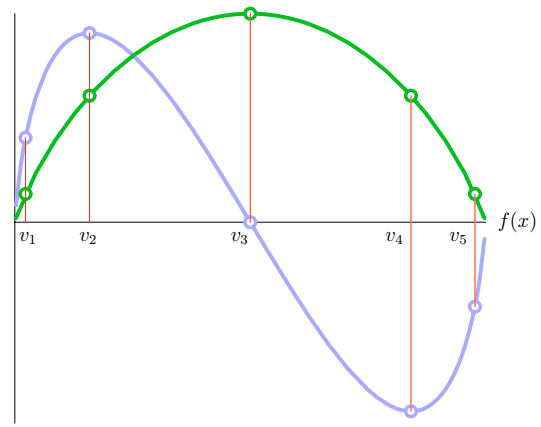
Remember that our ultimate goal is to arrive at a transfer function that makes the values at the boundary opaque. Observe that in Figure 3.5(a), the data value increases monotonically[1]. This means there is a one-to-one relationship between the data value and position. Therefore, it is possible to transform the first and second derivatives from functions of *position* to functions of *data value*. Five data values, v_1 through v_5, have been indicated on the vertical axis. The pattern of horizontal and vertical lines shows that for each of these five data values, we have an associated first and second derivative values. This association is formed through the relationship between data value and position which is governed by the graph of f. If we were to plot the derivatives associated with data value as we moved through the range of data values, we would get the curves seen in Figure 3.5(b). Data value, not position, is on the horizontal axis, on which v_1 through v_5 are shown. The crucial change is that *information about position has been entirely removed from the picture, and we are left with the relationship between the data value and its derivatives.*

There is a more elaborate conception of the graphs portrayed in Figure 3.5(b)

[1]This is, of course, the ideal situation. If the segment along which f and its derivatives has been measured does not lie directly between large regions of two distinct materials, the monotonicity condition may not hold. Again, we rely on the statistical properties of the histogram to provide the overall picture of the boundary characteristics.



(a) f, f', f'' versus position



(b) f', f'' versus f

Figure 3.5: Relationships between f, f', f''

that is well worth describing, since it can give us a more intuitive feel for why those graphs have their particular shape. For the time being, consider just the relationship between data value and the first derivative. Both of these quantities are functions of position, so we could draw a three dimensional graph showing this, as in Figure 3.6. Below the three dimensional curve we see data value versus

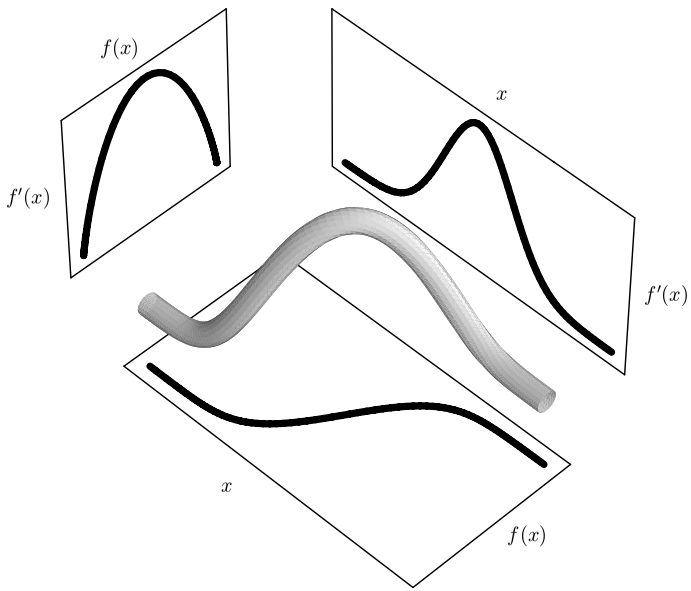


Figure 3.6: The three dimensional plot of f and f' as functions of position x is projected along the three different axes so as to demonstrate the relationship between each pair of variables.

position, and to the right, first derivative versus position. Both of these graphs are projections of the three dimensional curve. But there is a third way to project this curve: parallel to the position axis, so as to cast away all position information. We

are then left with the relationship between data value and first derivative, exactly the curve seen in Figure 3.5(b).

The same can be done for the relationship between data value and its second derivative, as seen in Figure 3.7. In the projections which are below and to the

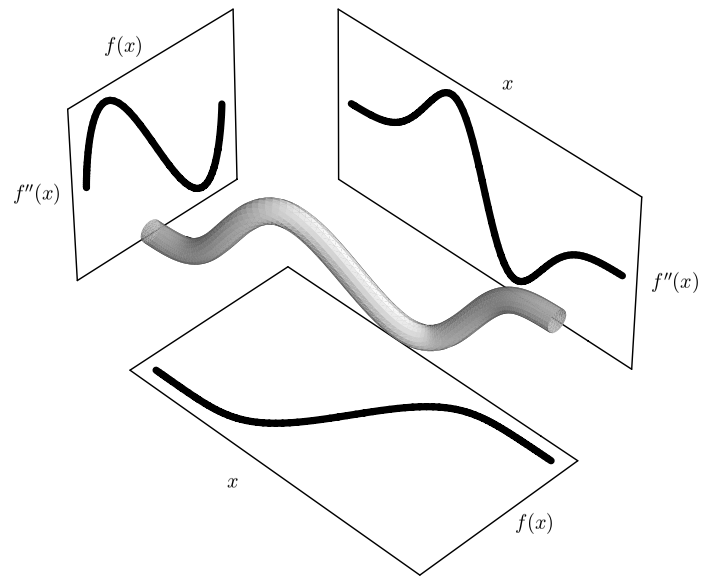


Figure 3.7: The three dimensional plot of f and f'' as functions of position x is projected along the three different axes so as to demonstrate the relationship between each pair of variables.

right of the three dimensional curve, we recognize the graphs of data value versus position and second derivative versus position. But again, we can project the three dimensional parametric curve along the position axis, distilling out the relationship between data value and its second derivative, to get the second curve in

Figure 3.5(b)

Now that Figure 3.5(b) is more thoroughly motivated, we can represent its content in a different way. Since we have "projected away" position information and reformulated the first and second derivatives as functions of data value, we can create a three dimensional plot of the first and second derivatives as functions of *data value*, as seen in Figure 3.8. We recognize that two of the projections of this curve are the same as in Figure 3.5(b). The full three dimensional curve has an approximately helical shape, as evidenced by the fact that the third projection of the curve (upper right) closes back on itself.

The significance of these curves, the three dimensional one as well as its projections, is that they provide a basis for automatically generating opacity functions. By analyzing an ideal boundary we have arrived at a position-independent relationship between the data value and its derivatives. Therefore, *if a three dimensional record of the relationship between f, f' and f'' for a given dataset contains curves of the type shown in Figure 3.8, we can assume that they are manifestations of boundaries in the volume.* With a tool to detect those curves and their position, one could generate an opacity function which makes the data values corresponding to the middle of the boundary (indicated with cross-hairs) the most opaque, and the resulting rendering should show the detected boundaries. Short of that, one could use a measure which responds to some specific feature of the curve (the zero crossing in f'', or the maximum in f') and base an opacity function on that. This is the approach taken in this thesis.

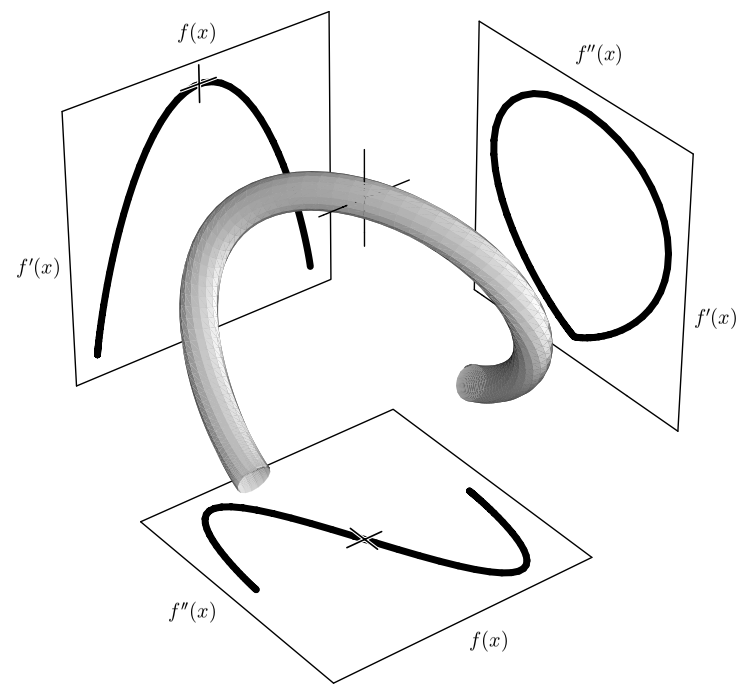


Figure 3.8: The three dimensional parametric plot of f, f', and f'' is projected along the three different axes so as to reveal the position-independent relationships between these three variables. The cross-hairs indicate the positions along the curves corresponding to the middle of the boundary.

Chapter 4

Histogram volume calculation

4.1 Histogram volume structure

A histogram is a structure for representing a discrete approximation of a probability distribution function. Given some variable x, a histogram of x can be thought of as a series of "bins" which collect occurrences of x based on the value x attains. The value of the histogram at some bin is the number of occurrences of x ("hits") in that bin. The usefulness of histograms is that they can provide a compact summary of large amounts of data, in the sense that statistical quantities (such as the mean or variance) which can be measured directly from the data can often be measured more easily from a histogram of that data.

In our case, we have three variables, f, f', and f'', but we want to do more than measure their individual probability distributions. We are interested in the probabilities associated with the relationship between the three variables. The most straight-forward way to do this is with a three dimensional histogram, with one axis for each of the variables. The span of each axis represents a fixed range of values for the corresponding variable. Each axis is divided into some number of

(one dimensional) bins, causing the interior volume is to be divided into a three dimensional array of rectilinear bins, not unlike the voxels of a standard volumetric dataset. Each bin in the three dimensional histogram represents the combination of a small range of values in each of the three variables. The value stored in each bin records the number of voxels in the original dataset which had a combination of f, f', and f'' values covered by that bin. This structure is termed the "histogram volume", seen in Figure 4.1.

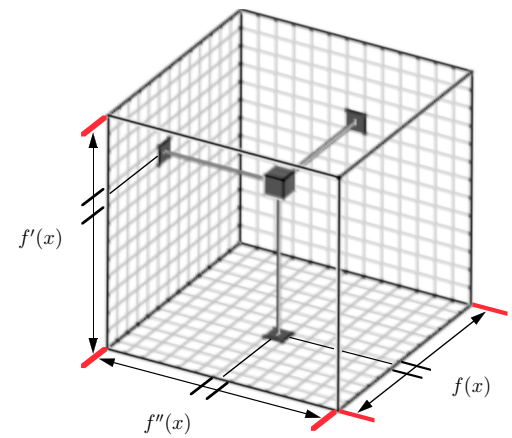


Figure 4.1: Histogram volume structure. Each of the three axes (for f, f', and f'') are divided into bins; the volume is thereby divided into a three dimensional histogram.

4.2 Histogram volume creation

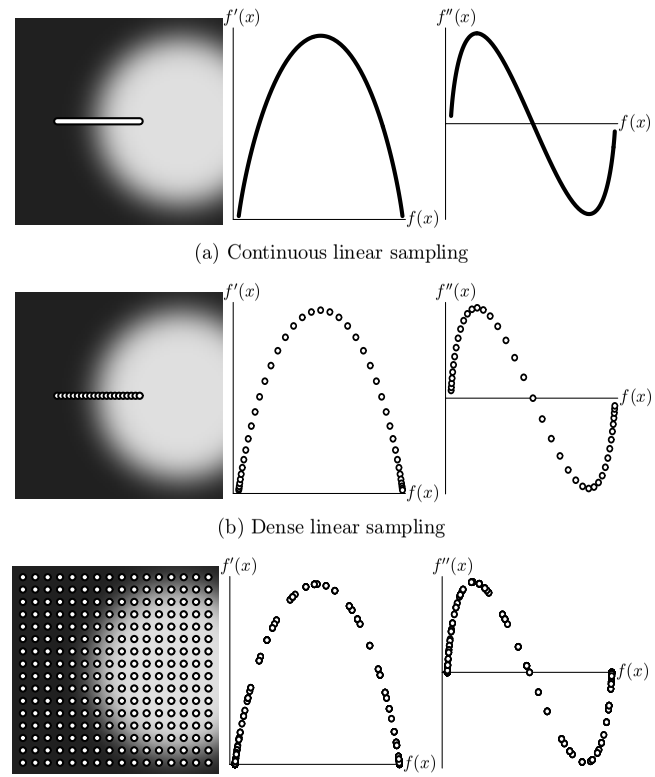
Having decided on a structure with which to represent the probabilistic relationship between f, f', and f'', we now must find a way to measure that relationship as it

occurs in a given dataset.

In Chapter 3, we found a position-independent relationship between f, f', and f'' that characterized an ideal boundary. However, to find that relationship, we afforded ourself the luxury of first knowing *where* the boundary was in the idealized dataset. For example, when plotting the data value and its derivatives as a function of position in Figure 3.4, we knew exactly where to place a path which crossed through the boundary. In the case of real volume data, we do not know where the boundaries are, and as discussed previously, we should not need to know. Yet we need some way to measure f, f', and f'' so as to reveal the same relationship.

This thesis has taken the simple approach of measuring f, f', and f'' at each point of a uniform lattice. Figure 4.2 helps illustrate why this can work. In 4.2(a), the thickness of the boundary is sampled continuously to produce smooth graphs of the data value versus derivative relationships, as was done in Figure 3.4. In 4.2(b), the boundary is sampled at a high frequency along the same path. Instead of producing a smooth plot of first or second derivative versus data value, we now have a scatterplot, but the sequence of measurements traces out the same relationships as before[1]. Finally, in 4.2(c) the boundary is sampled everywhere on a grid which uniformly covers the dataset, and which has no particular spatial relationship to the boundary. Nonetheless, the scatterplots of first and second derivative versus data value have the exact same shape as before, although now the points are distributed differently. Specifically, a much greater number of sample points have accumulated near the $f(x)$ axis, where the derivative values are nearly

[1]For visual clarity in the depiction of samples along the grayscale boundary (Figure 4.2(b), left), fewer samples are indicated than are plotted in the two scatterplots.



(a) Continuous linear sampling

(b) Dense linear sampling

(c) Sampling at gridpoints

Figure 4.2: Sampling the boundary: from continuous to discrete. Even when the measurements are taken over a grid which has no particular spatial relationship to the boundary, the relevant curves in the scatterplot are recorded, indicating the presence of a boundary.

zero, because there are many more samples within the homogeneous regions inside and outside the cylinder.

Figure 4.2(c) encapsulates the ramifications of our previous assumptions. The reason why curves appear in the scatterplots at all is our initial assumption of band-limited data acquisition, which causes all sharp physical boundaries to be measured as smooth transitions. We also made an assumption about material uniformity by stating that when materials are measured, they attain the same data value everywhere. This implies that the scatterplots will be coherent along the data value axis. Variations in the data value associated with the exterior or interior of the object would "smear" the scatterplots horizontally. In addition, we had made the assumption that the band-limiting was isotropic, which causes the measured boundaries to be uniform regardless of position or orientation. Variations in the thickness of the boundary would change the values of the derivatives, which would smear the scatterplots vertically.

The sole purpose of the histogram volume is to capture and represent the scatterplots of Figure 4.2(c) in a single structure. Thus, to the extent that our various assumptions about boundaries and band-limiting are valid, for every boundary that occurs in a dataset, we should be able to discern in the histogram volume the relationship between the data value and its derivatives. These relationships were first seen in Figure 3.8 and are evident in the scatterplots of Figure 4.2(c).

4.3 Implementation

Before we can create histogram volumes, there are several implementation issues that need to be resolved, starting with the spatial resolution of the grid on which

we sample f, f', and f'' throughout the volume. If the grid is too sparse, we risk missing the boundary. Instead of having a dense and continuous distribution of "hits" in the histogram volume, we would have only sparse or broken coverage of the curves we are looking for. If the grid is too dense, we have excess computation.

The approach taken in this thesis is to measure f, f', and f'' exactly once per voxel, at the original sample points of the dataset. There are two reasons for this. First is ease of computation. Because we are working only with the original data points of the volume, there is no need to perform reconstruction to measure f, and the measurements of f' and f'' are greatly facilitated by the use of *masks*, a computational tool to measure properties of discrete data. These will be explained shortly. Second, because bandlimiting ensures that there is always some boundary blurring, and because the boundaries of real objects are apt to assume a variety of positions and orientations relative to the sampling grid, sampling over the whole dataset will insure that the sample points fall at a variety of positions within the boundary region, leading to more complete coverage of the curves in the scatterplots and histogram volume. Of course, the more bandlimiting there is, the more blurred the boundaries will be, leading to a greater accumulation of hits along the curves in the histogram volume and in the scatterplots. A more mathematical treatment of this relationship is given in Appendix A.

The remaining implementation issues of histogram volume creation are unfortunately matters of parameter setting. Though the goal of this research is to reduce the total number of parameters that need to be adjusted before informative renderings can be achieved, there are in fact a few input parameters necessary for the creation of the histogram volume itself. This is why the generation of transfer functions is (at best) *semi*-automatic.

One variable in the histogram volume creation is the histogram resolution, that is, how many bins to use. The trade-off here is between the large storage and processing requirements caused by having many bins, and the inability to resolve patterns in the histogram volume caused by having too few. In our experiments, good results have been obtained with histogram volumes of sizes between 80^3 and 256^3, though there is no reason that the histogram volumes need to have equal resolution on each axis. In general, better results have been obtained from using larger histogram volumes. While a 256^3 volume does not represent an exceedingly large storage or computational challenge to modern systems, the ability to get decent results from, for instance, a 128^3 histogram volume, with the associated nearly 8-fold speed increase, makes the use of smaller histogram volumes an attractive alternative. A brief comparison of results from using different size histogram volumes is given in Chapter 6.

A more significant issue is the the range of values that each axis of the histogram volume should represent. Obviously along the data value axis, we should include the full range, since we want to capture all the values at which boundaries might occur. But along the axes for the first and second derivatives, it may make sense to include something less than the full range. This is because derivative measures are by nature very sensitive to noise, and thus will take on very high (or very low) values wherever noise occurs. If significant noise is present in the volume data, and if we included the full range of derivative values in the histogram volume, then the important and meaningful sub-range of derivative values might be compressed to a relatively small number of bins. This would hamper the later step of detecting patterns in the histogram volume. Of course, we have no *a priori* way of knowing what the meaningful ranges of derivatives values are. Currently, the derivative

value ranges are set by hand, and this is a subject in need of further research.

The most significant implementation issue is the method of measuring the first and second directional derivatives. The first derivative is actually just the gradient magnitude. From vector calculus [MT96] we have:

$$\mathbf{D}_{\mathbf{v}} f = \nabla f \cdot \mathbf{v}, \tag{4.1}$$

thus

$$\mathbf{D}_{\widehat{\nabla f}} f = \nabla f \cdot \widehat{\nabla f} = \nabla f \cdot \frac{\nabla f}{\|\nabla f\|} = \|\nabla f\|. \tag{4.2}$$

Unfortunately there is no similarly compact formula for $\mathbf{D}^2_{\widehat{\nabla f}} f$, the second directional derivative along the gradient direction. Twice applying Equation 4.2 gives:

$$\begin{aligned} \mathbf{D}^2_{\widehat{\nabla f}} f &= \mathbf{D}_{\widehat{\nabla f}}(\|\nabla f\|) = \nabla(\|\nabla f\|) \cdot \widehat{\nabla f} \\ &= \frac{1}{\|\nabla f\|} \nabla(\|\nabla f\|) \cdot \nabla f \end{aligned} \tag{4.3}$$

Or, using the Taylor expansion of f along $\widehat{\nabla f}$ [Fau93] gives:

$$\mathbf{D}^2_{\widehat{\nabla f}} f = \frac{1}{\|\nabla f\|^2} (\nabla f)^{\mathrm{T}} \, \mathbf{H} f \, \nabla f \tag{4.4}$$

where $\mathbf{H} f$ is the Hessian of f, a 3×3 matrix of second partial derivatives of f [MT96]. Alternatively, we can use the Laplacian $\nabla^2 f$ to approximate $\mathbf{D}^2_{\widehat{\nabla f}} f$:

$$\mathbf{D}^2_{\widehat{\nabla f}} f \approx \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \tag{4.5}$$

The approximation is exact only where the isosurfaces have zero mean surface curvature [Fau93].

As all these derivative measures require taking first or second partial derivatives of the data along the axial directions, we need to develop masks which can measure these quantities. Masks are vectors of scalar weights, designed to measure

certain properties of sampled data. The mask is centered at the sample point of interest, and the products between corresponding mask weights and sample values are summed to produce the final measurement. The combination of weights in the mask determine its functionality. As a simple example, Figure 4.3 demonstrates a mask ($[-0.5 \;\; 0 \;\; 0.5]$) which measures the first derivative of one dimensional data. Derivative measurement with this mask is called *central differencing* and is very

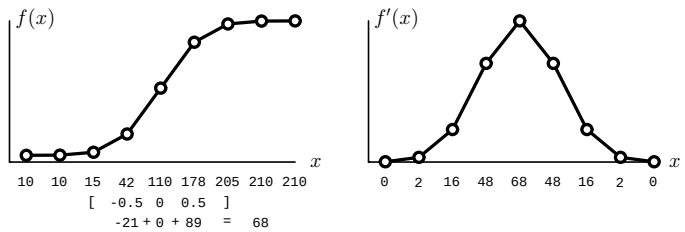


Figure 4.3: Using a mask to measure first derivative. The data values 42, 110, and 178 are multiplied by the mask values -0.5, 0, and 0.5 respectively, and the products are summed to produce the final measurement.

commonly used in image processing[GW93]. Appendix B describes how central differences can be generalized to measure first and second partial derivatives in three dimensions.

Even though we have a way of measuring all the necessary *partial* derivatives in the volume, we still need to chose among the different measurement approaches for $\mathbf{D}^2_{\widehat{\nabla f}} f$. Each of the three expressions for $\mathbf{D}^2_{\widehat{\nabla f}} f$ given above suggest different implementations. While the Hessian method (Equation 4.4) has been found to be the most numerically accurate, the others have proven sufficiently accurate in practice to make them appealing by virtue of their computation efficiency. The Laplacian (Equation 4.5) computation is direct and inexpensive, but the most sen-

sitive to quantization noise. The gradient of the gradient magnitude (Equation 4.3) is better, and its computational expense is lessened if the gradient magnitude has already been computed everywhere for the sake of volume rendering (e.g., as part of shading calculations). A more thorough characterization of the different second derivative measures is given in Appendix C. Deciding which derivative measures to use is the final parameter for histogram volume creation.

We can now give an algorithm for histogram volume creation:

- Initialize the histogram volume to all zero values.
- Make one pass through the volume looking for the highest values of f' and f'', and the lowest value of f''. (Assume zero for the lowest value of f'). Set ranges on the histogram volume axes accordingly.
- Make a second pass through the volume:
  - Measure f, f', and f'' at each sample point,
  - Determine which bin in the histogram volume corresponds to the measured combination of f, f', and f'', and
  - Increment the bin's value.

4.4 Histogram volume inspection

In Section 4.2 we stated that if there are boundaries in the original dataset that conform to our boundary model, there should be curves like that of Figure 3.8 in the histogram volume. We now verify this fact by looking at histogram volumes from datasets known to contain boundaries. Although one may be tempted to volume render the histogram volume from various viewpoints in order to visualize

its content, this turns out to be difficult and unrevealing due to the speckled nature of the histogram volume. A better way to visualize histogram volumes is to project them along either the first or second derivative axis. This produces two dimensional scatterplots of data value versus first derivative, or data value versus second derivative, as were seen in 4.2(c).

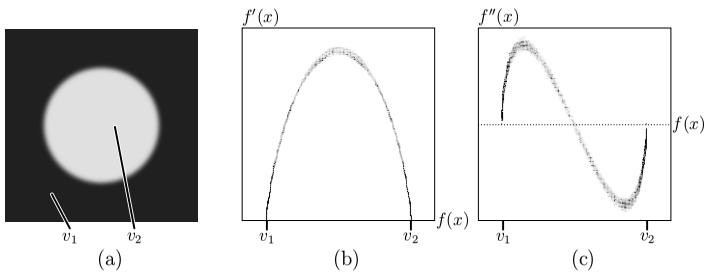


Figure 4.4: Dataset cross-section and histogram scatterplots for cylinder. A cross-section of the dataset is shown in (a), illustrating that there is only one boundary in the volume, with the two material values tagged v_1 and v_2. (b) is a projection of the histogram volume showing the relationship between f and f'; (c) likewise shows the relationship between f and f''. In both (b) and (c) the data values v_1 and v_2 are indicated on the $f(x)$ axis.

We start by looking at synthetically generated datasets, beginning with a simple dataset— a cylinder like that of Figure 3.4. Figure 4.4(a) shows the cylinder in cross-section. The histogram volume is then projected along the f'' axis to produce a f' versus f scatterplot (Figure 4.4(b)) and along the f' axis to produce a f'' versus f scatterplot (Figure 4.4(c)). In these scatterplot images, the data value and its derivatives are oriented on the axes as they were in Figure 4.2 to facilitate comparison. The darkness in the scatterplots encodes the number of hits for a given (value, gradient) pair— the darker, the more hits. These scatterplots

conform closely to the curves that were seen in Figure 4.2(a).

As a second verification of our discussion, and as a demonstration of the utility of the histogram volume to capture information about objects' boundaries, we analyze a second synthetic dataset which contains *two* materials distinct from the background. Figure 4.5(a) shows the dataset in cross-section. An outer shell with

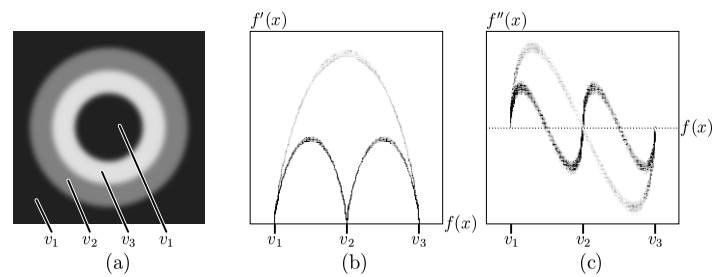


Figure 4.5: Dataset cross-section and histogram scatterplots for two material cylinder. (a) is a cross-section of the data, (b) and (c) are projections of the histogram volume showing f' versus f and f'' versus f, respectively. As there are three distinct values v_1, v_2, v_3 in the dataset, with a boundary between each pair of values, there are three curves in the scatterplots. The curves start and end at the three data values v_1, v_2, and v_3, indicated on the $f(x)$ axis.

an intermediate data value (v_2) surrounds an inner shell with a high data value (v_3), and the core is the same as the background value (v_1). Hence there are three boundaries. This fact is also revealed in Figure 4.5(b) and Figure 4.5(c) by the presence of three curves, one for each boundary.

Now we look at histogram volumes for real datasets, starting with a CT scan of a turbine blade[2]. As the blade is made of a single kind of metal, we would expect to find a single boundary indicated by the histogram volume or its projections.

[2]Dataset courtesy of GE Corporate Research and Development

Figure 4.6 illustrates this. The two large dark spots in the histogram projections

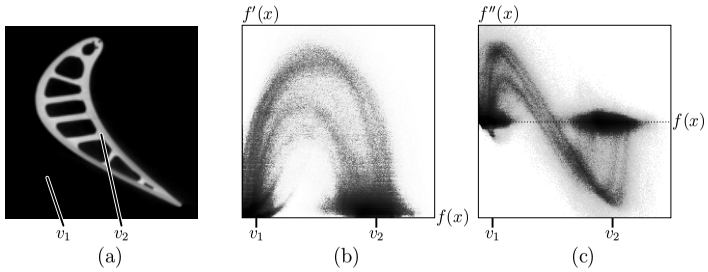


Figure 4.6: Dataset cross-section and histogram scatterplots for turbine blade. The cross-section (a) shows that there is only one boundary. This is supported by the scatterplots of f' versus f (b) and f'' versus f (c).

arise from the large regions of nearly constant value within the air or the metal. The curves are the result of the boundary between the two, and their shape matches those of Figure 4.4, indicating a good match between the properties of an ideal boundary, and the actual properties of this dataset. Another exemplary dataset is the engine block CT scan first seen in Figure 1.1. Figure 4.7 shows a dataset slice and the two scatterplots for the engine dataset. The most prominent boundary curve is for the transition from the background value (v_1) to the intermediate value (v_2), which comprises the majority of the engine block's interior. A fainter curve is discernible between the intermediate value (v_2) and a high value (v_3). Finally, a very faint and diffuse curve is evident arching over the other two boundaries, spanning from the background value (v_1) to the highest value (v_3). It makes sense that these last two curves should be so much fainter than the first, because, as we can tell from looking at the image of the dataset cross-section, the surface area of the boundary between the background value and the intermediate value is much

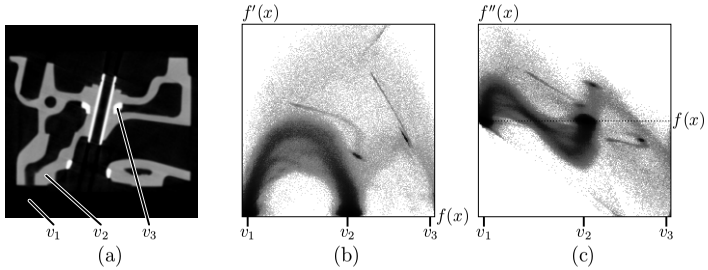


Figure 4.7: Dataset cross-section and histogram scatterplots for engine block. The cross-section in (a) shows that there are three kind of material, tagged with v_1, and v_2, and v_3. We can also see that like the nested cylinders in Fig. 4.5, there is a boundary between every pair of values. Though not especially clear, the scatterplots of f' versus f (b) and f'' versus f (c) support this.

greater than the area of either of the other two boundaries. Thus, there are fewer voxels contributing to corresponding curves on the scatterplot.

We finish with Figure 4.8 which shows the scatterplots for one of the CMDA neuron datasets, generated by electron microscope (EM) tomography. Unlike the previous datasets, there is no obvious evidence of clean boundaries in the scatterplots, only the vague shape of the curves we are looking for. The main for reason for this is that CT scans provide a better match to our assumed ideal boundary characteristics than does electron microscope tomography. It is these EM datasets, where the boundary is less than ideal, which will be the most severe test of the algorithms developed in this thesis.

It should be noted that a related technique has been used in computer vision for feature identification. Panda and Rosenfeld [PR78] use two dimensional scatterplots of data value and gradient magnitude to perform image thresholding for

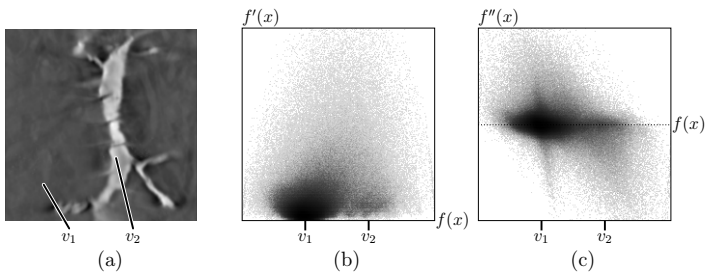


Figure 4.8: Dataset cross-section and histogram scatterplots for neuron. The cross-section of the data (a) shows that there is no one constant value either inside or outside the neuron, though two data values v_1 and v_2 have been tagged and indicated on the $f(x)$ axes of the f' versus f (b) and f'' versus f (c) scatterplots. The neuron boundary seen in the cross-section is irregular because a range of values occur on either side of the boundary. Correspondingly, the scatterplots are much more diffuse than for previous datasets. However, the scattering of hits still roughly conforms to the over-all shape of the curves we are looking for.

night vision applications. They, however, do not assume a particular boundary model, instead limiting their analysis of the scatterplot to identifying particular distributions within regions of low and/or high gradient magnitude.

4.4.1 Techniques for scatterplot visualization

The techniques which have made possible the display of the various scatterplots shown so far should be described in greater detail, since the process for scatterplot visualization has a large effect on how informative they are. We use the first derivative versus data value scatterplot for the turbine blade as our example.

Figure 4.9 illustrates how simple approaches to scatterplot display are not effective. In Figure 4.9(a), a linear mapping was used to determine gray level from the number of hits in the scatterplot. The problem is that the image is too faint because the number of voxels in the background material (air) overwhelms the number of voxels everywhere else. This can be alleviated somewhat with a gamma correction (Figure 4.9(b)), but important detail within the boundary region curve is not visible.

To solve this problem we have use a standard contrast enhancement technique from image processing called *histogram equalization*[3][GW93]. Histogram equalization flattens the histogram of the gray levels in an image, so that all gray levels are utilized approximately equally. Figure 4.9(c) shows the result of histogram equalization on Figure 4.9(a). This is an improvement, but since the image is now too dark, we apply a gamma correction of 3.0 to make the image lighter. The fine structure with the boundary curve region of the scatterplot is now visible. The

[3]The fact that the image we are processing in this situation is itself a histogram is not especially meaningful or important.

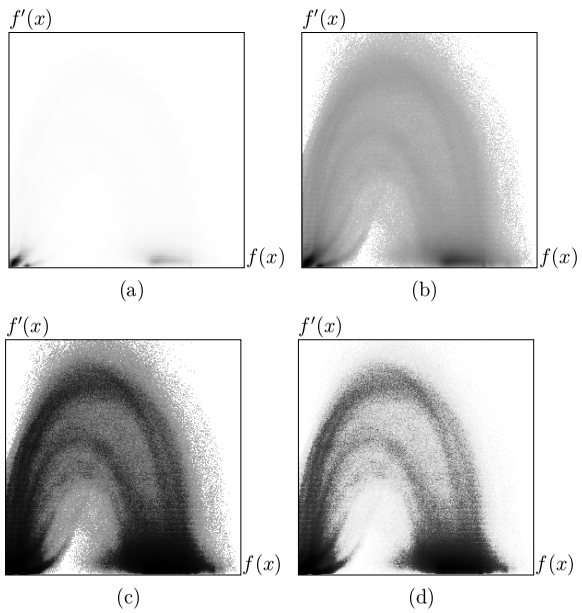


Figure 4.9: Methods for scatterplot visualization. In (a), the mapping from number of voxels accumulated to gray level is linear. The dark spot in the lower left represents the voxels within background material, air. In (b), a gamma correction of 0.22 was applied to make the image darker, but detail is still lacking. (c) shows the result of applying histogram equalization to (a), and (d) shows the result of following histogram equalization with a gamma correction of 3.0.

amount of gamma correction applied after histogram equalization was chosen by hand for each of the different datasets shown in this thesis.

Chapter 5

Opacity function generation

5.1 Mathematical boundary analysis

In order to develop a method for opacity function generation that uses our boundary model and the information stored in the histogram volume, we need to look at the equation used to describe the ideal boundary data value as a function of position, as plotted in Figure 3.4:

$$v = f(x) = v_{min} + (v_{max} - v_{min})\frac{1 + \mathrm{erf}(\frac{x}{\sigma\sqrt{2}})}{2} \tag{5.1}$$

In its current form, the ideal boundary gets its characteristic shape from the erf() function. We define this important function and plot it in Figure 5.1.

$$\mathrm{erf}(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt \tag{5.2}$$

Note in Figure 5.1 that since erf() is centered around the origin, the boundary function $f(x)$ is also centered around where position x is zero. There are two essential differences between $f(x)$ and $\mathrm{erf}(x)$. First, in $f(x)$ the argument of erf() is scaled by $1/\sigma\sqrt{2}$. The σ parameter controls the thickness, or spread, of the

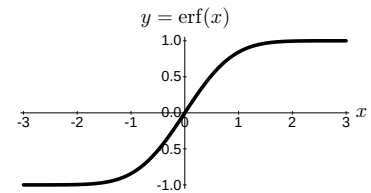


Figure 5.1: The error function, erf()

boundary. For large σ, the argument of erf() is reduced, and the boundary is stretched out more. The boundary is much narrower and sharper with a small σ. Second, erf()'s range has been shifted and scaled so that the range of $f(x)$ is between v_{min} and v_{max}. As x approaches negative infinity, $\mathrm{erf}(\frac{x}{\sigma\sqrt{2}})$ approaches -1, and thus $f(x)$ approaches v_{min}. Conversely, as x approaches positive infinity, $f(x)$ approaches v_{max}. At $x = 0$, the middle of the boundary, $f(x)$ is half-way between v_{min} and v_{max}.

The first and second derivatives of f are as follows:

$$f'(x) = \frac{v_{max} - v_{min}}{\sigma\sqrt{2\pi}}\, e^{-\frac{x^2}{2\sigma^2}} \tag{5.3}$$

$$f''(x) = -\frac{x(v_{max} - v_{min})}{\sigma^3\sqrt{2\pi}}\, e^{-\frac{x^2}{2\sigma^2}} \tag{5.4}$$

Our choice of boundary parameterization means that $f'(x)$ is a normalized Gaussian, with σ being the usual standard deviation. Since the Gaussian has inflection points at $\pm\sigma$, this is where $f''(x)$ attains its extrema. The same positions can serve as (somewhat artificial) delimiters for the extent of the boundary. We *define* the "thickness" of the boundary to be 2σ. Figure 5.2 shows how the the definition of thickness roughly accounts for the region in which the data value transitions between the values on either side of the boundary.

The equations for $f'(x)$ (Equation 5.3) and $f''(x)$ (Equation 5.4) look very

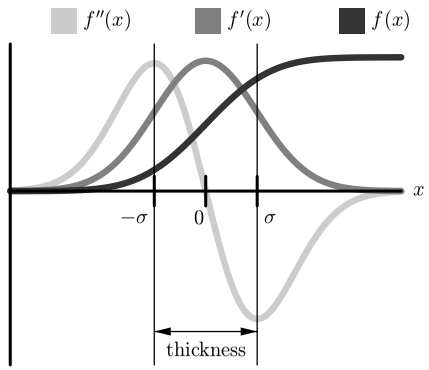


Figure 5.2: Relationship between thickness and the boundary function. The thickness of material boundaries is defined to be 2σ, which extends from one extremum in f'' to the other. The middle of the boundary is defined to be where $x = 0$.

similar because of the special derivative properties of e^x. In fact, the only difference between $f'(x)$ and $f''(x)$ is a scaling factor $\frac{1}{-\sigma^2}$ and the position x:

$$\frac{f''(x)}{f'(x)} = -\frac{x}{\sigma^2} \tag{5.5}$$

This is intriguing because it means there is a way to infer the position x along a boundary knowing just σ and the first and second derivatives. σ can be recovered if the extremal values of $f'(x)$ and $f''(x)$ are known. Recall from Figure 5.2 that $f'(x)$ has its maximum at 0, and that $f''(x)$ has a maximum at $-\sigma$ and a minimum

at σ. This gives us two different ways of calculating σ.

$$\begin{aligned}
\frac{f'(0)}{f''(-\sigma)} &= \frac{\frac{v_{max}-v_{min}}{\sigma\sqrt{2\pi}}\ e^{-\frac{0}{2\sigma^2}}}{-\frac{-\sigma(v_{max}-v_{min})}{\sigma^3\sqrt{2\pi}}\ e^{-\frac{(-\sigma)^2}{2\sigma^2}}} = \sigma\sqrt{e} \\
\frac{f'(0)}{f''(\sigma)} &= \frac{\frac{v_{max}-v_{min}}{\sigma\sqrt{2\pi}}\ e^{-\frac{0}{2\sigma^2}}}{-\frac{\sigma(v_{max}-v_{min})}{\sigma^3\sqrt{2\pi}}\ e^{-\frac{\sigma^2}{2\sigma^2}}} = -\sigma\sqrt{e} \\
\Rightarrow \sigma &= \frac{f'(0)}{\sqrt{e}\, f''(-\sigma)} = -\frac{f'(0)}{\sqrt{e}\, f''(\sigma)}
\end{aligned} \tag{5.6}$$

Knowing σ, one can then recover position from $f'(x)$ and $f''(x)$ with Equation 5.5:

$$x = \frac{-\sigma^2 f''(x)}{f'(x)} \tag{5.7}$$

Section 3.4 described the ability to transform the first and second derivatives as functions of position into functions of data value. That transformation takes on special importance in light of Equation 5.7. f' and f'' are functions of position in that equation, but there is no reason why they couldn't also be functions of data value. In that case, we would have a way to map from data value back to position along a boundary. Since our goal is to produce an opacity function which makes the middle of boundaries opaque, this information is exactly what is necessary for opacity function generation.

Fortunately, the histogram volume contains that information. The histogram volume was created by recording the position-independent relationship between the data value and its derivatives. In that case, the position that was "projected out" in the histogram formation was the (x, y, z) position within the volume. Now, having created the histogram volume, in the next section we will analyze it in order to re-create "position" information with Equation 5.7. This time, the "position" in question is actually a signed distance to the nearest boundary, which will be extremely helpful in the creation of opacity functions.

5.2 Opacity functions of data value

We now define some important functions of data value. $g(v)$ is the average first directional derivative of f over all the positions $\mathbf{x}$ at which $f(\mathbf{x}) = v$. $h(v)$ is likewise the average second directional derivative at value v. These two functions can be

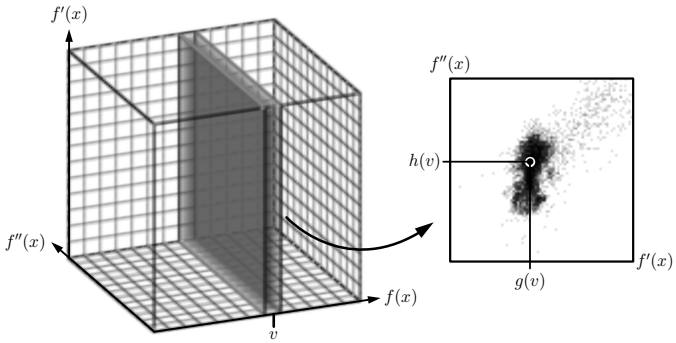


Figure 5.3: Calculating $g(v)$ and $h(v)$ from the histogram volume. A schematic of the histogram volume is shown, with the axes labeled. The slice of the histogram volume at data value v is extracted, and the accumulation of sample points within this slice is shown as a diffuse dark region. The coordinates of the centroid of the region along the f' and f'' axes are the $g(v)$ and $h(v)$ values.

obtained from the histogram volume for a given dataset as Figure 5.3 illustrates. The slice of the histogram volume at data value v records the distribution of the measured first and second derivative values for all the sample points with data value v. The centroid of the distribution on the slice is then calculated. The coordinate of the centroid along the f' axis is $g(v)$, and along the f'' axis it is $h(v)$. If there are data values v which received no hits in the histogram volume (that is, value v never occurs in the original volume dataset), then $h(v)$ and $g(v)$ are left

undefined.

Knowing $g(v)$ and $h(v)$ for the range of data values, we can calculate σ with Equation 5.6. For more robust results, instead of using just the maximum or minimum of $h(v)$, we can use both[1]:

$$\sigma = \frac{2\sqrt{e}\ \max_v(g(v))}{\max_v(h(v)) - \min_v(h(v))} \tag{5.8}$$

Now we can produce the mapping $p(v)$ that was referred to earlier, giving a position along a boundary as a function of data value:

$$\begin{aligned} p(v) &= \frac{-\sigma^2 h(v)}{g(v)} \\ &\approx \frac{-\sigma^2 f''(f^{-1}(v))}{f'(f^{-1}(v))} = x \end{aligned} \tag{5.9}$$

$p(v)$, like $f(v)$ and $g(v)$, is undefined for values v which never occur in the original volume dataset.

Mathematically, we see that $p(v)$ is a local inverse of $f(x)$, since $p(v) = x$, and by definition $f(x) = v$. Intuitively, $p(v)$ implies which side of the nearest boundary a data value v tends to fall. For values closer to v_{min}, the position $p(v)$ will be negative; for values closer to v_{max}, $p(v)$ will be positive. At the value halfway between v_{min} and v_{max}, $p(v)$ will be zero, the position at the middle of the boundary. In practice, it is useful to modify Equation 5.9 to account for the fact that due to low-level measurement noise, the gradient magnitude at the interior of materials is rarely exactly zero. Knowing how it differs from zero is a matter of experience, but assuming one can find a scalar quantity g_{thresh} which is higher than the ambient gradient magnitude[2], Equation 5.9 is re-formulated, with a slight

[1] An implicit assumption in this equation is that g attains its maxima at $f(0)$, and that h attains its extrema at $f(\pm\sigma)$.

[2] As will be shown in the next chapter, there is some freedom in which value is chosen for g_{thresh}, but this is the defining constraint.

loss of mathematical accuracy, as

$$p(v) = \frac{-\sigma^2 h(v)}{\max(g(v) - g_{thresh}, 0)} \tag{5.10}$$

To demonstrate the steps involved in the calculation of the position function $p(v)$, we consider another synthetic dataset, containing two concentric spheres with distinct data values. Figure 5.4 shows a central cross-section of the dataset,

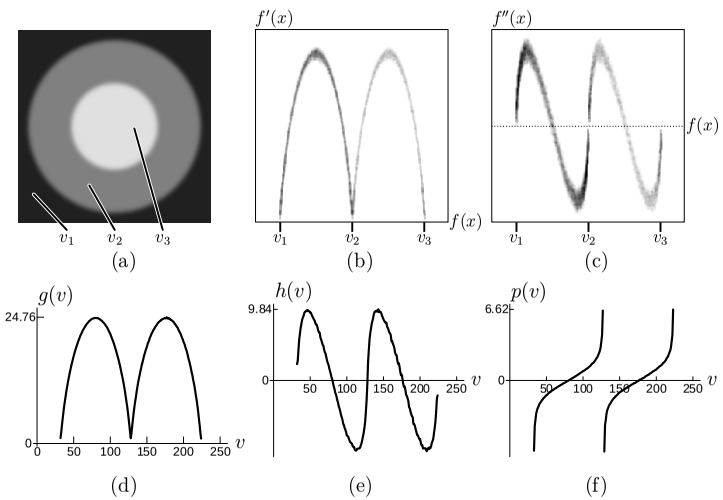


Figure 5.4: Precursors to $\alpha(v)$ calculation for nested concentric spheres. A dataset cross-section and the usual scatterplots are shown in (a), (b), and (c). The three data values v_1, v_2, v_3 are tagged in the dataset cross-section and marked on the horizontal axes of the two scatterplots. The functions $g(v)$ and $h(v)$ were calculated from the histogram volume and are plotted in (d) and (e). From $g(v)$ and $h(v)$, $p(v)$ is calculated with Equation 5.10, and shown in (f).

as well as the usual scatterplots. The boundaries that occur in the dataset can be learned by inspecting the cross-section or by interpreting the scatterplots. The

background value v_1 shares a boundary with the lower material value v_2, and the lower material value has a boundary with the higher one v_3. The figure next shows plots of the functions $g(v)$, $h(v)$, and $p(v)$. Note that for this dataset, the graph of $g(v)$ closely mimics the shape of the f' versus f scatterplot, and the graph of $h(v)$ closely mimics the f'' versus f scatterplot. The plot of $p(v)$, calculated with Equation 5.9, also has the expected shape, similar to two graphs of $f(x)$ rotated ninety degrees. This is a graphical demonstration of the concept that $p(v)$ is a local inverse of data value $f(x)$. Because $p(v)$ does the opposite of $f(x)$ by mapping from *data value* to *position*, we can see from the graph of $p(v)$ that there are two data values which occur in the middle of boundaries. As expected, they are the data values half-way between v_1 and v_2, and half-way between v_2 and v_3.

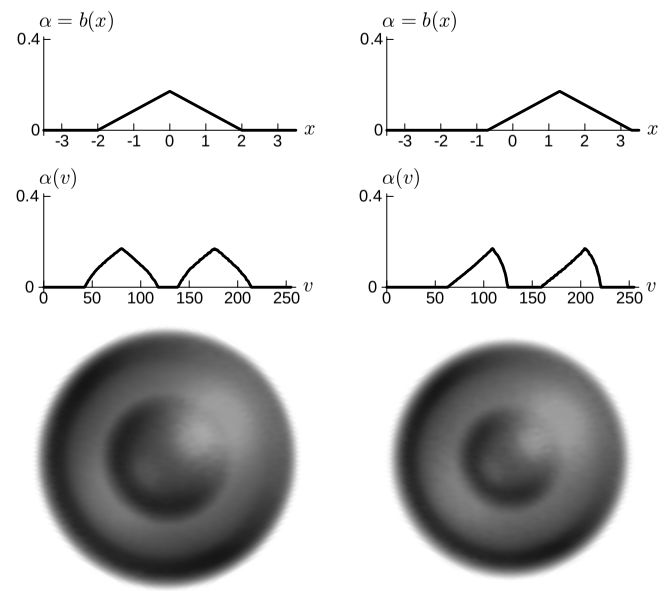
Once $p(v)$ is known, it is a relatively simple matter to generate an opacity function which makes the boundaries of objects visible in the rendered image. Since the middle of a boundary is always where the position is zero, we need only make opaque those data values v for which $p(v)$ is near zero. Exactly how this is done is determined by the user, who specifies a function $b(x)$, which we term the *boundary emphasis function*. The function $b(x)$ maps from position along a boundary to opacity. Since $b(x)$ should be non-zero only near zero, we have not been especially careful in preventing $p(v)$ from attaining infinite values due to a low $g(v)$; such a data value v should not contribute to the final image. With $b(x)$, the user can directly control the proximity of the rendered boundary to the object interior, and whether rendered boundaries will appear thick or thin, sharp or fuzzy. The final opacity function $\alpha(v)$ is then defined as

$$\alpha(v) = b(p(v)) \tag{5.11}$$

For those values v where $p(v)$ is undefined, we define $\alpha(v)$ to be zero.

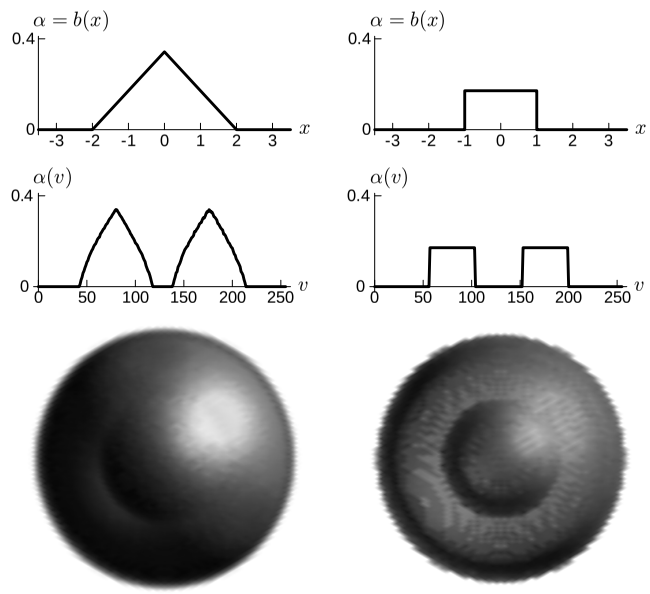
Figures 5.5 and 5.6 illustrate and discuss how the choice of the boundary emphasis function affects the opacity function and the rendered image, using the same concentric spheres dataset analyzed in Figure 5.4. Instead of exploring the parameter space of all possible opacity functions $\alpha(v)$, the user explores the parameter space of $b(x)$ and lets the information from the histogram volume, embodied in $p(v)$, constrain the search to those opacity functions which display object boundaries. Defining opacity as a function of position within a boundary then becomes a more intuitive task than defining opacity as a function of data value, as there is a more predictable relationship between changes made to the boundary emphasis function and the corresponding change in rendered results.

It should be stressed that the user does not set the initial location of the peaks in $\alpha(v)$, since this is determined by the information in $p(v)$. However the user can *modify* the location of the peaks, as well as their width, height, and shape. This is the main benefit of the method presented in this thesis: *if the histogram volume has successfully captured information about the boundaries in the dataset, the user enjoys high-level control over the character of the rendered boundaries without being required to give an exact specification of* $\alpha(v)$. For instance, the user can specify (as in Figure 5.5) that the opacity linearly ramp up and down near the boundary. Unless the user somehow has an intuition for the position function $p(v)$, it is unlikely that he or she would be able to manually create the opacity function $\alpha(v)$ which achieves this. Furthermore, the $\alpha(v)$ generated by this method is usually sensible enough that it can be manually edited if desired. For example, since this technique will attempt to make *all* boundaries opaque, a useful supplement to the interface would be a feature which allows removal of the peaks in $\alpha(v)$ for one or more boundaries, so as to remove the corresponding boundaries



(a) Semi-transparent (b) Closer to interior

Figure 5.5: Using $b(x)$ to control the proximity of the rendered boundary to the object interior. In (a), a simple boundary emphasis function $b(x)$ (at top) makes the positions between -2 and 2 opaque, peaking at zero, the middle of the boundary. Below (middle), the resulting opacity function $\alpha(v)$ is shown. The data values near 80 and 176 for which $p(v)$ (Fig.5.4) was zero receive maximum opacity. The rendering (bottom) nicely shows the boundaries of the two spheres. In (b), the peak in $b(x)$ (top) has been shifted upward, to emphasize values closer to the interior of the spheres. Correspondingly, the peaks in $\alpha(v)$ (middle) have shifted upward, and in the rendering (bottom), the spheres appear slightly smaller.



(a) More opaque (b) Sharper boundary

Figure 5.6: Using $b(x)$ to control the character of the rendered boundary. In (a), the boundary emphasis peak (top) has been raised to make the rendered boundary more opaque. Accordingly, the peaks in $\alpha(v)$ (middle) have raised, and the rendered boundary (bottom) is indeed more opaque. In (b), the shape of the boundary emphasis function (top) has been changed to be more discontinuous. This characteristic is reflected in the resulting $\alpha(v)$ (middle), as well as in the rendering (bottom).

from the rendering.

Even though we have made some strong assumptions about the boundary characteristics in the volume dataset, the technique described here typically works well even if the material boundaries are not "ideal". Essentially, by taking the quotient of the second and first derivatives, and by having $b(x)$ assign opacity to positions around zero, we are more apt to make opaque those data values associated with both low second derivatives and high first derivatives, consistent with the criteria usually used in computer vision edge detectors. Or, even if $p(v)$ is not a perfect indicator of "position relative to boundary", the sign change in f'' around its zero-crossing affords us some control over whether we want to emphasize regions closer to or further from the object's interior.

In the case of poor data, there are some adjustments to the method presented here which may yield better results. When the material boundaries are far from ideal, the calculation of σ from Equation 5.6 is apt to produce a poor results. Evaluating Equation 5.9, we can see that σ^2 appears as a scaling factor, so at worst, this will require the user to experiment with different scalings in the domain of $b(x)$. Also, the $g(v)$ and $h(v)$ calculated from the histogram volume may be very noisy. Some improvements have been noted from smoothing these slightly. Results from these experiments, and from utilizing the technique on a variety of datasets are presented in Chapter 6.

5.3 Opacity functions of data value and gradient magnitude

The opacity functions considered so far have assigned opacity based on data value alone. Value-based opacity functions are easy to implement and facilitate simple and efficient rendering algorithms. Higher quality renderings can sometimes be obtained, however, if the opacity is assigned as a function of both data value *and* gradient magnitude. We term these *two dimensional* opacity functions. Defining these two dimensional opacity functions by hand is especially challenging because there are even more degrees of freedom than in one dimensional, value-based opacity functions. Fortunately, the ideas presented so far easily generalize to allow semi-automatic generation of two dimensional opacity functions.

Analogous to the definition of $h(v)$, we define $h(v,g)$ to be the average second derivative over all locations where the data value is v and the gradient magnitude is g. As before, this information can be extracted from the histogram volume, as illustrated in Figure 5.7. For data value v and first derivative g there is a one dimensional histogram which records the distribution of second derivative values for those sample points which had both data value v and gradient magnitude g. $h(v,g)$ is the average of the distribution of second derivative values. In direct emulation of Equations 5.10 and 5.11, we define a new position function, and from that an opacity function:

$$p(v,g) = \frac{-\sigma^2 h(v,g)}{\max(g - g_{thresh}, 0)} \tag{5.12}$$

$$\alpha(v,g) = b(p(v,g)) \tag{5.13}$$

$\alpha(v,g)$ is defined to be zero for those (v,g) pairs which never occur in the volume. σ is calculated exactly the same as before, using Equation 5.8. But whereas before

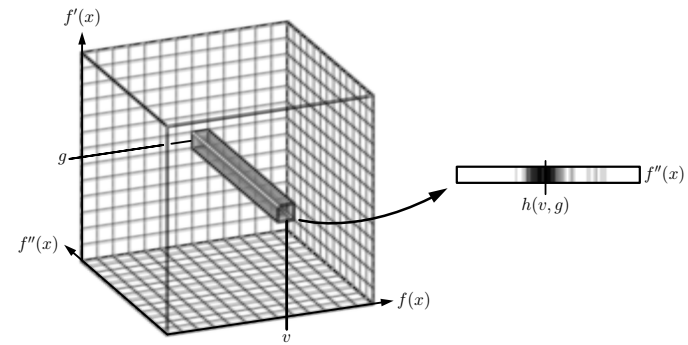


Figure 5.7: Calculating $h(v,g)$ from the histogram volume. As in Figure 5.3, a histogram volume schematic is shown, but now a single scanline of the volume is extracted where data value is v and the first derivative is g. The accumulation of sample points along this line is shown as a diffuse dark region. The location of the center of the region is $h(v,g)$.

a position was found for each data value, we are now finding position for every combination of data value and gradient magnitude. The resulting opacity function is thus two dimensional instead of one dimensional. As in the case of value-based opacity functions, the method presented here will generate two dimensional opacity functions which make all detected boundaries opaque. A simple "lasso" tool could then be used to select different regions in the two dimensional opacity function to render one boundary at a time.

To demonstrate the generation of two dimensional opacity functions, consider the dataset first seen in Figure 4.5, containing two concentric cylindrical shells each at distinct data values. As can be seen in Figure 5.8(a), this dataset is similar to the nested spheres used in the previous section, in that it has a background value v_1, and two material values v_2 and v_3. However, here the v_3 region shares a boundary

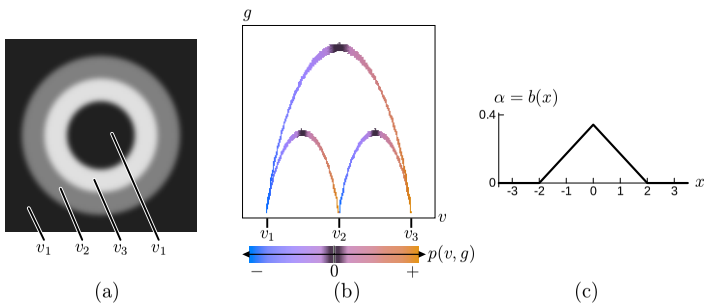


Figure 5.8: Precursors to the $\alpha(v, g)$ calculation for nested cylindrical shells. (a) shows a cross-section of the data, labeling the regions of the three distinct data values. The color image in (b) represents the $p(v, g)$ function that was computed from Equation 5.12. The values of $p(v, g)$ are shown with a colormap, the legend of which is shown. In (c) a simple boundary emphasis function is plotted.

not just with the v_2 region, but also with the v_1 background region. If one wanted to visualize just the v_1–v_3 boundary with a value-based opacity function, the opacity function would have to make the values halfway between v_1 and the v_3 opaque. Unfortunately, these are precisely the values near v_2, the value within the outer cylindrical shell. Thus the outer shell would obscure the v_1–v_3 boundary in any rendering. For this reason, one dimensional opacity functions are not sufficient to selectively render the boundaries in this dataset.

Figure 5.8(b) shows $p(v, g)$ as calculated from Equation 5.12, with the aid of a colormap which is orange for positive values, blue for negative values, and purple in between the two extremes. Additionally, there is a small band of dark gray for values very close to zero. Where $p(v, g)$ is undefined, the image is white. Note that as one moves along the arches from left to right, $p(v, g)$ changes from negative to positive, hitting zero at the peak of the arches. This is closely analogous to the

plot of $p(v)$ for the nested spheres (Figure 5.4(f)) in the previous section, where position went from negative to positive as we moved up the data value axis. The position information $p(v, g)$ is then fed into a simple boundary emphasis function $b(x)$, shown in Figure 5.8(c). Even though the user-specified $b(x)$ is still a simple function of one variable, it facilitates the creation of a two dimensional opacity function.

The relationship between the two dimensional opacity function and the final rendering is shown in Figure 5.9. The initial opacity function calculated with Equation 5.13 (and the $b(x)$ plotted in Fig. 5.8(c)) is shown at the bottom of Figure 5.9(a). As there were three boundaries recorded in the histogram volume, there are three bright regions in the opacity function, one for each boundary. To allow seeing internal structure, none of the boundaries have been rendered fully opaque. When we remove one of the regions of high opacity from $\alpha(v, g)$ (Figure 5.9(b)), the corresponding boundary disappears from the rendering. In Fig. 5.9(b), what remains is the surface of the inner cylindrical shell. Since this object has boundaries with both the background and the lower object value, we can selectively render one or the other part of the surface, as demonstrated in Fig 5.9(c).

Though a contrived example, this serves to demonstrate the power of the method for two dimensional opacity function generation. In the one dimensional case, each detected boundary was marked by a spike in the opacity function of data value. Here, each detected boundary is marked by a bright spot in the two dimensional opacity function. This immediately gives the user a feel for what types of boundaries occur in the volume. By making only the middle of boundaries opaque, the different significant regions in the opacity function can be easily differentiated

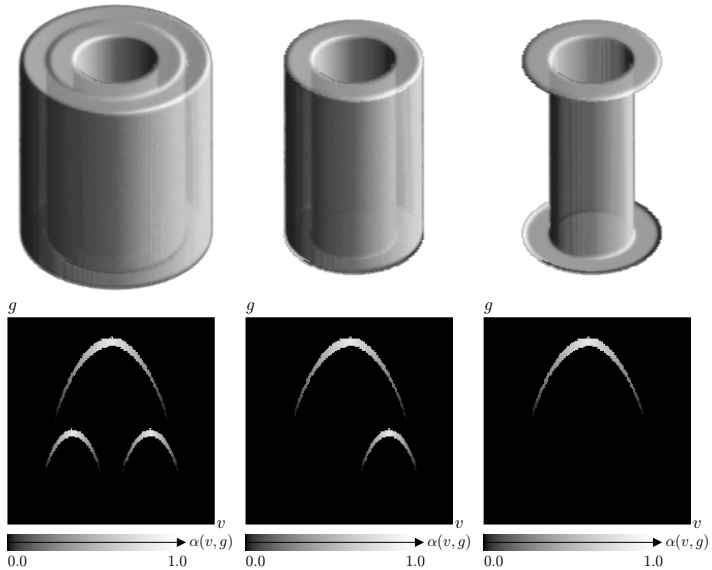


Figure 5.9: Renderings of nested cylinders and corresponding $\alpha(v, g)$. (a) shows the two dimensional opacity function $\alpha(v, g)$ initially generated from Equation 5.13 and a simple $b(x)$. As with the scatterplots, the data value is on the horizontal axis, and gradient magnitude on the vertical. Black areas represent transparency; white is fully opaque. All three boundaries are visible in the corresponding rendering. By zeroing out areas in $\alpha(v, g)$, other boundaries can be removed with ease, as seen in (b) and (c).

from each other by the user, and then edited to control which boundaries are rendered. Other examples of the effectiveness of this technique in generating two dimensional opacity functions are given in Chapter 6.

5.4 Comparison with Levoy's two dimensional opacity functions

The main result of Levoy's influential paper, "Display of Surfaces from Volume Data"[Lev88], was to demonstrate the usefulness of volume rendering for what might otherwise be considered the domain of isosurface rendering— the visualization of surfaces in regularly sampled volume data. He demonstrated that not only is direct volume rendering appropriate for this task, but that in some aspects it is superior to isosurface rendering. It avoids the explicit creation of a surface geometry, and it displays ambiguous or under-sampled features as "faint wisps", a more accurate representation than that given by isosurfacing. His success can be attributed to the careful design of the opacity functions which he developed for the two visualization tasks described in his paper. Because there are many similarities between the goals addressed in Levoy's paper and in this thesis, we review the opacity functions which Levoy designed and the visualization problems which motivated them. We conclude the chapter with some observations about the differences between Levoy's methods and the ones advanced in this thesis.

5.4.1 Visualizing isosurfaces in smoothly varying data

Levoy's first task was visualizing isosurfaces of electron density functions. These datasets are characterized by smooth and slow variations of the data value through-

out the volume. From the four sample cross-sections of such a dataset shown in

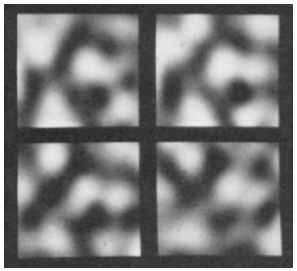
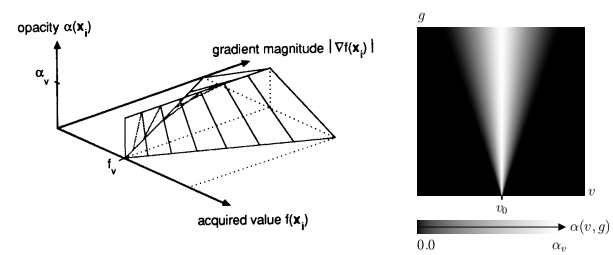


Figure 5.10: Slices of electron density dataset

Figure 5.10 (taken from Levoy's paper), we can see that these datasets do not exhibit any of the characteristics assumed for this thesis. There are no regions of relatively constant value contained by boundary regions where the data value changes quickly. The discontinuities which exist with material boundaries are not present in electron density functions. Still, it is important to review the opacity function Levoy developed for this type of data because it is the *only* style of two dimensional opacity function which has since received any significant attention in the volume rendering literature[Mur95, YESK95, LCN98]. Also, the opacity function he designed for this application has proven to have utility for a wider variety of datasets.

Because Levoy uses volume rendering instead of true isosurface extraction to visualize the isovalue contours, the rendered surfaces will necessary have some thickness. Levoy states that the "most pleasing image" results from making the isosurface appear to have equal thickness everywhere. One way to achieve this is with a opacity function which takes into account the gradient of the data value. Specifically, if we assume that the data value varies *linearly* with position near the

isosurface, then the range of values which needs to be made opaque is *proportional* to the magnitude of the gradient. When the data value changes quickly as a function of position, a wider range of values need to be made opaque so as to create an opaque region of fixed thickness, and conversely low rates of change in data value require opacity for a narrower ranges of values. The plot of the two dimensional opacity function which achieves this is shown in Figure 5.11(a), a diagram taken from Levoy's paper. Also, to facilitate comparison with the gray-scale graphical representation of the two dimensional opacity functions presented in this thesis, a gray-scale representation of Levoy's function is shown in Figure 5.11(b). We continue the convention used in Chapter 5, letting $v = f(\mathbf{x})$ and $g = |\nabla f(\mathbf{x})|$, though Levoy's figures employ a slightly more complicated notation.



(a) (b)

Figure 5.11: Levoy's opacity function for isovalue surfaces. In (a), a three dimensional plot of the function, from Levoy's paper. The data value axis extends to the lower right, and the gradient magnitude axis extends to the upper right, with opacity on the vertical axis. In (b), a gray scale representation of the same function. In (a), the isovalue is notated f_v, while in (b) it is notated v_0.

There are three degrees of freedom in this type of opacity function which need to be defined before stating its formula:

- v_0: the isovalue around which the opacity will be assigned (Levoy refers to this quantity as f_v)
- r: the desired thickness of the isosurface, in units of voxel edge length
- α_0: the maximum opacity of the isosurface (notated α_v by Levoy)

Strictly speaking, Levoy defines his opacity as a function of *position* $\mathbf{x}$ within the volume, but the terms which appear in his formula are the data value at $\mathbf{x}$, $v = f(\mathbf{x})$, and the gradient magnitude at $\mathbf{x}$, $g = |\nabla f(\mathbf{x})|$. Thus the opacity function can be expressed as:

$$\alpha_{iso}(v,g) = \alpha_0 \begin{cases} 1 & \text{if } g = 0 \text{ and } v = v_0 \\ 1 - \frac{1}{r}\left|\frac{v-v_0}{g}\right| & \text{if } g > 0 \text{ and } v - rg \leq v_0 \leq v + rg \\ 0 & \text{otherwise} \end{cases} \tag{5.14}$$

It is the second line of Equation 5.14 which establishes the proportionality between the gradient magnitude and the range of data values which are assigned opacity. This in turn creates the "V" shape of the opaque region in (v, g) space, flaring linearly with gradient magnitude (Figure 5.11)

5.4.2 Visualizing region boundary surfaces

The second visualization task addressed in Levoy's paper is the rendering of "region boundary surfaces" in the specific context of CT medical imaging technology. Like the visualization task discussed in this thesis, Levoy's goal is to visualize the interfaces between adjacent regions of uniform material. However, since Levoy restricts himself to the domain of CT datasets, he makes strong assumptions about the spatial inter-relationship between the material regions. Quoting from this paper:

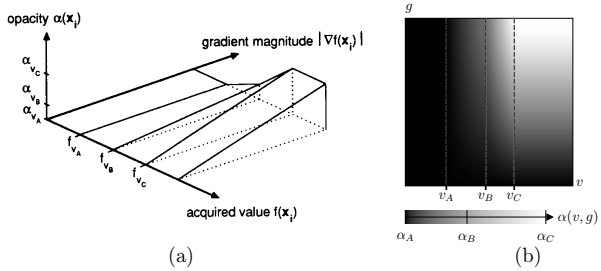
> "We assume that scenes contain an arbitrary number of tissue types bearing CT numbers falling within a small neighborhood of some known value. We further assume that tissues of each type touch tissues of at most two other types in a given scene. Finally, we assume that, if we order the types by CT number, then each type touches only types adjacent to it in the ordering."

To characterize the opacity function Levoy developed for this case, we notate the "CT numbers" (the materials' data values in the CT scan) as v_i, where each material is tagged with an integer i. Next, an opacity α_i is assigned to material i so that the various materials can be visually differentiated in the rendering. Values which fall between the known material values v_i and v_{i+1} are assigned an opacity by linearly interpolating between α_i and α_{i+1}. Finally, and most importantly, to accentuate the boundaries *between* the regions, the opacity is scaled by the gradient magnitude. Figure 5.12(a) shows the plot of the opacity function which appeared in Levoy's paper, and Figure 5.12(b) shows the gray-scale representation.

The final opacity function is then:

$$\alpha_{bound}(v,g) = g \begin{cases} \alpha_{i+1}\frac{v-v_i}{v_{i+1}-v_i} + \alpha_i\frac{v_{i+1}-v}{v_{i+1}-v_i} & \text{if } v_i \leq v \leq v_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{5.15}$$

In comparing our method with the second of Levoy's two opacity functions, it should be stressed that he assumes the CT numbers v_i are *known* for each type of biological material. This thesis, however, aims to be more general and require less prior knowledge. Because the opacity function generation presented in this thesis is grounded in the measured properties of the data, the user is (ideally) freed from setting material data values by hand. Levoy is nonetheless justified in making his assumption, because in medical imaging situations where CT is being used, there



(a) (b)

Figure 5.12: Levoy's opacity function for region boundary surfaces. In (a), a three dimensional plot of the function, using the same axes as in Figure 5.11(a). In (b), a gray scale representation of the function. The three known material values are labeled f_{v_A}, f_{v_B}, and f_{v_C} in (a), while in (b) they are notated simply v_A, v_B, and v_C. Similarly, the material opacities α_{v_A}, α_{v_B}, α_{v_C} in (a) are notated α_A, α_B, and α_C in (b).

is a high degree of consistency between scans in the data values which arise from human tissue types.

The assumption about having a strict progression of nested regions with increasing data values is stronger than any assumption made in this thesis. For instance, the nested cylindrical shells in Section 5.3 violate Levoy's assumption but are well handled by our method for two dimensional opacity function generation. On the other hand, this thesis makes the strong assumption that all boundaries have the same thickness. Levoy makes no assumptions whatsoever about the the thickness of boundaries, nor does he rely on any particular boundary model for defining the opacity function. He simply requires that the boundary region have higher gradient magnitude that the material interior, so that scaling the opacity by the gradient magnitude (Equation 5.15) causes material interiors to be suppressed in the rendering.

Chapter 6

Results

In this chapter we apply the techniques previously described to a variety of "real world" volume datasets. In addition, the examples will demonstrate the effects of various adjustments to the algorithm which have been alluded to in previous chapters without illustration. We will analyze the following datasets:

- Turbine Blade: A CT dataset is blurred slightly as a pre-process to make the blade's boundary better match the ideal boundary, and to improve rendered surface quality.
- Engine Block: The effectiveness of two dimensional opacity functions is shown. Two dimensional opacity functions reveal structural details of the dataset not visible with any one dimensional opacity function.
- CT Head: This example illustrates the effect of changing histogram volume size on the quality of the generated opacity functions. Also, renderings made with our two dimensional opacity functions permit a comparison between our method and Levoy's method for boundary visualization.

- Spiny Dendrite (hippocampus): Given a dataset in which the measured boundaries are not ideal, some experimentation with g_{thresh} and the boundary emphasis function is needed in order to create an informative rendering.

- Spiny Dendrite (neostriatum): Some experimentation with the boundary emphasis function is needed to produce an informative rendering with a one dimensional opacity function. This dataset also demonstrates the alteration to a two dimensional opacity function necessary to produce a clear rendering of the dendrite boundary. Finally, Levoy's opacity function for isovalue contours is used to produce a comparable rendering.

6.1 Turbine Blade: Effect of dataset blurring

We start with the "turbine blade" dataset, a CT scan of a propeller blade from the engine of a fighter jet. The turbine blade itself is only about three inches tall. The dataset is available from GE Corporate Research and Development[1]. The raw data consists of unsigned 16-bit values; for this analysis it was quantized to eight bits by linearly scaling the range $0 - 8612$ to $0 - 255$ and then clamping values above 255. The voxels in the original data are not isotropic; in the Z direction, along the length of the blade, the sample spacing was half that in the X-Y plane. The data was down-sampled in the X-Y plane by blurring slightly with the following cubic polynomial kernel $h(x)$, shown in Figure 6.1, and then sampled at every other data

[1]For information contact Bill Lorensen, `lorensen@crd.ge.com`

point.

$$h(x) = \begin{cases} \frac{1}{2}|x|^3 - |x|^2 + \frac{2}{3} & \text{for } |x| \leq 1 \\ -\frac{1}{6}|x|^3 + |x|^2 - 2|x| + \frac{4}{3} & \text{for } 1 < |x| \leq 2 \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

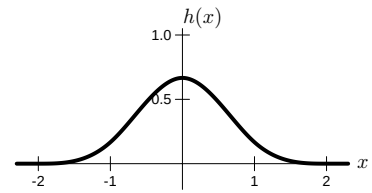
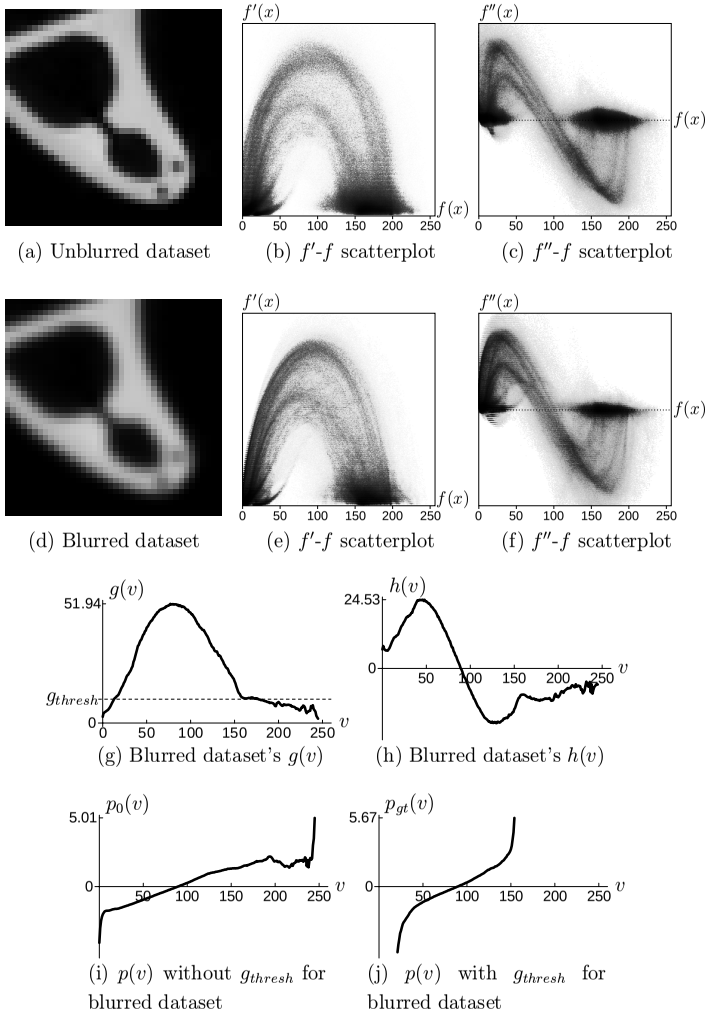


Figure 6.1: Cubic spline used prior to down-sampling

A close-up of a slice of the downsampled dataset, as well as the usual scatterplots are shown in Figures 6.2(a-c). The histogram volume was computed using the Hessian second derivative measure, at a resolution of 256^3. The scatterplots show the curves which indicate the presence of a boundary between the two major material data values, but the curves do not quite have the shape of those for an ideal boundary. In the scatterplot of first derivative versus data value, for example, (Figure 6.2(b)), we can see that the boundary curve is slightly wider and more rounded than the more pointed parabolic shape of the ideal curve, like that seen in Figure 4.2(a).

To demonstrate that judicious blurring of the dataset causes the boundaries to better match the boundary model, we convolve along each axis with the normalized Gaussian kernel $b(x)$ seen in Figure 6.3, and then calculate a new histogram volume. As a result of the blurring, the curves in both scatterplots are closer to the ideal



(a) Unblurred dataset (b) f'-f scatterplot (c) f''-f scatterplot

(d) Blurred dataset (e) f'-f scatterplot (f) f''-f scatterplot

(g) Blurred dataset's $g(v)$ (h) Blurred dataset's $h(v)$

(i) $p(v)$ without g_{thresh} for blurred dataset (j) $p(v)$ with g_{thresh} for blurred dataset

Figure 6.2: Analysis of turbine blade dataset

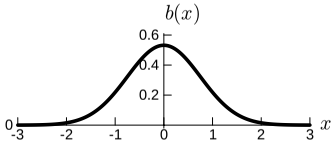


Figure 6.3: Gaussian kernel used for blurring. The standard deviation σ is 0.75

shape. However, as is visible in Figure 6.2(d), creating a better match to the boundary ideal reduced some clarity in the fine detail of the dataset.

Figures 6.2(g) and 6.2(h) show plots of $g(v)$ and $h(v)$ as calculated from the histogram volume of the blurred dataset. Their form is consistent with the fact that there is a single boundary: at the data value half-way between the values for air (approximately 10) and metal (approximately 175), $g(v)$ has a single peak and $h(v)$ has a zero-crossing. The plot of $g(v)$ illustrates the g_{thresh} quantity which originated in Section 5.2. Within the range of values for air and metal, the gradient magnitude is not zero because of slight measurement noise. g_{thresh} is set (by hand) to the value of gradient magnitude within the materials, as indicated in Figure 6.2(g) with a dotted line. Next, two calculations of $p(v)$ (with Equation 5.10) are plotted. Figure 6.2(i) shows $p_0(v)$ calculated *without* the g_{thresh} modification described in Section 5.2 and defined in Equation 5.10. Figure 6.2(j), showing $p_{gt}(v)$ calculated *with* the indicated g_{thresh}, is a much better match to the curve we would expect for a distinct boundary. It is clear that an appropriate setting for g_{thresh} is important for making $p(v)$ accurately reflect position within the boundary.

Having obtained an appropriate $p(v)$ function, it is now easy to obtain a rendering of the dataset which shows only the surface of the turbine blade. Figure 6.5(a) shows the boundary emphasis function $b(x)$ used, which strives to makes the opaque boundary region about five units thick (where one unit is the length of a voxel

edge). The opacity linearly ramps up to 0.8 at the middle of the boundary, and is 0.0 at 2.5 units on either side of the middle. Figure 6.5(b) shows the opacity function $\alpha(v)$ calculated (with Equation 5.11) from the $p_{gt}(v)$ and $b(x)$ just described. As we would expect for a nearly ideal boundary, the opacity function mirrors the triangular shape of the boundary emphasis function. Since the specified $b(x)$ is highest at $x = 0$, $\alpha(v)$ is highest for the data value v such that $p(v) = 0$, which is approximately 89.

A minor problem with the analysis performed so far is that the σ calculation was not optimal. According to Equation 5.8, with the $g(v)$ and $h(v)$ calculated, $\sigma = 1.284$ for this dataset, implying a boundary thickness of $2\sigma = 2.568$. Judging from the cross-section seen in Figure 6.2(d), this may be plausible. However if we inspect the result of applying the opacity function $\alpha(v)$ to the same piece of the cross-section in Figure 6.5(c), it does not look like the thickness of the opaque boundary region is five voxels, as prescribed by the $b(x)$ used. It was found that a boundary emphasis function which makes the opaque boundary region any thinner led to a rendering with gaps in the surface. Rendered surface quality is also a problem when trying to visualize the *unblurred* version of the dataset, since its boundary region is so thin. Using the $b(x)$ shown with the blurred dataset, however, the rendered surface seen in Figure 6.5(d) appears smooth and solid. By making the maximum opacity in $b(x)$ only 0.8, instead of 1.0, we can see the support struts inside through the outer surface of the blade. For comparison, Figure 6.4 shows a slice through the blade which reveals the position of the struts inside.

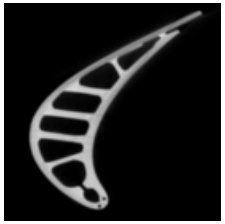


Figure 6.4: Cross-section of blade showing struts

6.2 Engine Block: Effectiveness of two dimensional opacity functions

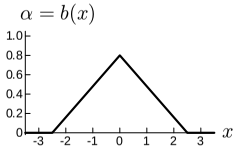
The CT scan of the "engine block" has become a reference dataset which appears throughout the volume rendering literature. While the method for one dimensional opacity function generation works passably with the dataset, the two dimensional opacity functions are very effective.

The dataset is available with the Stanford VolPack volume rendering library distribution, or can be downloaded from the Stanford Computer Graphics Laboratory web page[2]. It is a $256 \times 256 \times 110$ volume of 8-bit values. Because the CT scan contains a lot of background (air), the dataset can be significantly cropped without impinging on the engine block itself[3]. We use the Hessian second derivative measure to produce a 256^3 histogram volume which includes first derivative values between zero and 126.7 and second derivative values between -127.6 and 127.6.

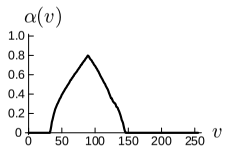
Figures 6.6(a) through 6.6(c) show a slice of the dataset and the two scat-

[2] `ftp://www-graphics.stanford.edu/pub/volpack/data/engine/`

[3] Along the X axis, we use voxels 56 through 211 inclusively; along the Y axis, 16 through 224.



(a) Boundary emphasis function



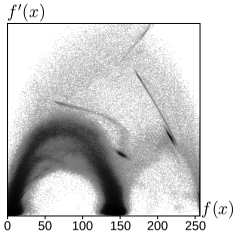
(b) Opacity function

(c) $\alpha(v)$ applied to dataset

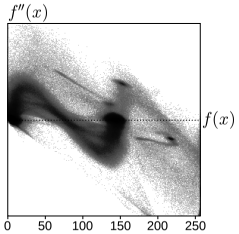
(d) Rendered image

Figure 6.5: Rendering the turbine blade dataset

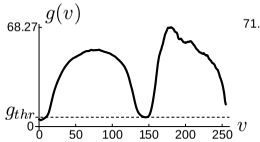
(a) Dataset slice



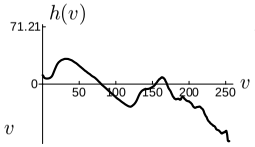
(b) f'-f scatterplot



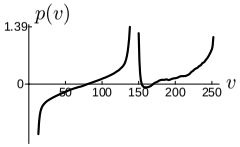
(c) f''-f scatterplot



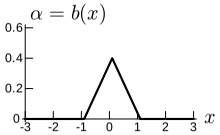
(d) Graph of $g(v)$



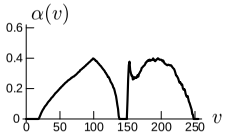
(e) Graph of $h(v)$



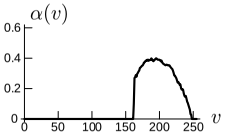
(f) Graph of $p(v)$



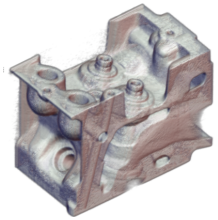
(g) Boundary emphasis



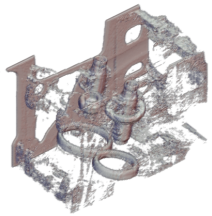
(h) Resulting $\alpha(v)$



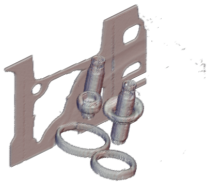
(i) $\alpha(v)$ modified



(j) Rendered with $\alpha(v)$ in (h)



(k) Rendered with $\alpha(v)$ in (h), without lower spike



(l) Rendered with $\alpha(v)$ in (i)

Figure 6.6: Analysis and renderings of engine block with one dimensional opacity functions

terplots. As mentioned in Section 4.4, there is the obvious mark of a boundary between the data values around 10 and 140, as well as fainter indications of boundaries between 140 and 255, and between 10 and 255. Incidently, from the fact that the curves ending at value 255 appear to be clipped on their right side, we can infer that the quantization of the raw data to eight bit values involved a somewhat aggressive clamping of high values.

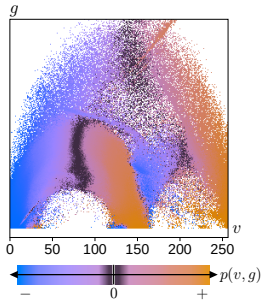
Figures 6.6(d) through 6.6(f) are the plots of $g(v)$, $h(v)$, and $p(v)$ calculated from the histogram volume. The plot of $g(v)$ includes an indication of the g_{thresh} which was used for this dataset. Like the nested cylindrical shells which were analyzed in Section 5.2, the engine block has three boundaries, including one spanning the highest and lowest values, from about 10 to 255. As is visible in the scatterplot of second derivative versus data value, the presence of this boundary exerts a negative influence on the average second derivative for the data values above 140, which pushes the plot of $h(v)$ downward for the same range of values. Offsetting the location of the sign change in $h(v)$ in turn causes the curious shape of the $p(v)$ graph— around value 150 the graph of $p(v)$ turns upward, not downward as we would expect. Because $h(v)$ is negative instead of positive at the data value v where $g(v) - g_{thresh}$ is zero, in accordance with Equation 5.10, $p(v)$ is positive instead of negative. However, the lower boundary, between 10 and 140, has such a greater surface area that it is not influenced by the other boundary, thus for lower data values, the plots of $g(v)$, $h(v)$, and $p(v)$ look as they should.

This demonstrates that datasets containing boundaries with overlapping ranges of data value cannot be correctly analyzed with this algorithm to produce *accurate* opacity functions of data value alone. In spite of this, the algorithm still generates passable results. Using a simple triangular spike centered at $x = 0.1$ and extend-

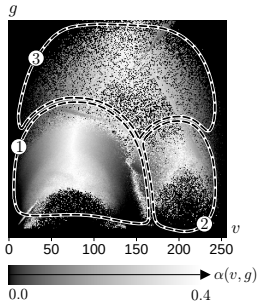
ing one unit on each side for the boundary emphasis function (Figure 6.6(g)), the opacity function generated (Figure 6.6(h)) produced the rendering seen in Figure 6.6(j). The outer boundary is rendered well. The boundary emphasis function was centered at $x = 0.1$ instead of $x = 0$ to reduce the amount of air which is erroneously made opaque. Manually removing the lower of the two bumps in $\alpha(v)$ produced the rendering seen in Figure 6.6(k). We see that the surface boundaries are partially occluded by voxels which are not part of the boundary and should have been made opaque. This is caused by the small spike on the left side of the higher bump in $\alpha(v)$. Manually removing it (Figure 6.6(i)), we see the higher material boundary clearly (Figure 6.6(l)).

We now render the same dataset using two dimensional opacity functions. The first step is to calculate the function $p(v, g)$ from Equation 5.13. The result is shown in Figure 6.7(a), with the aid of a colormap which is blue for negative values, orange for positive values, with a dark portion near zero. Because of this dark portion, we can see in the image of $p(v, g)$ those places in (v, g) space where the position is near zero, which corresponds to the middle of a boundary. The three dark regions visible in the image of $p(v, g)$ help clarify the fact that there are three boundaries in the engine block dataset, even if the three curves in the first derivative versus data value scatterplot (Figure 6.6(b)) aren't easily visible. The same boundary emphasis function used earlier for one dimensional opacity function generation is used again to generate the two dimension opacity function seen in Figure 6.7(b). Three regions in the opacity function have been outlined and labeled; the placement of these outlines was established by the locations of the positive and negative positions visible in the $p(v, g)$ image.

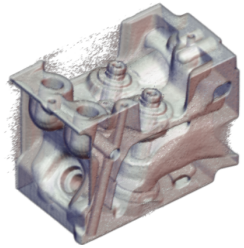
The opacity function was used unchanged to produce the rendering in Fig-



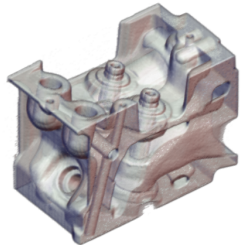
(a) Colormapped $p(v, g)$



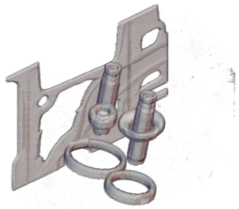
(b) $\alpha(v, g)$; boundary regions are outlined and labeled



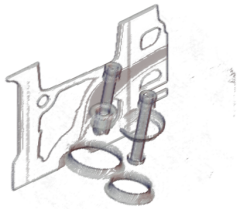
(c) Rendered with $\alpha(v, g)$ in (b)



(d) Using only regions 1, 2, and 3



(e) Using only regions 2 and 3



(f) Using only region 3

Figure 6.7: Analysis and renderings of engine block using two dimensional opacity functions. Renderings (d), (e), and (f) use only some regions of the $\alpha(v, g)$ in (b) and assign zero opacity outside the regions.

ure 6.7(c). For Figure 6.7(d), the regions 1, 2, and 3 of the opacity function were included, excluding the small area below region 1, thereby removing the small pieces of background which were incorrectly made opaque in the previous rendering. The outer surface in this rendering is noticeably cleaner and more consistent than that seen in the rendering from the one dimensional opacity function, Figure 6.6(j). Next, region one was removed from the opacity function to show the surface of the higher material components of the dataset (Figure 6.7(e)). Next, region two is removed from the opacity function, producing a rendering (Figure 6.7(f)) which shows only the boundary between the highest and lowest data values. Comparing the last two renderings, we learn structural information about the dataset which cannot be visualized with any one dimensional opacity function.

6.3 CT Head: Effect of changing histogram volume size

The CT scan of a female cadaver's head which Levoy used in his original paper has, like the "engine block", become a reference dataset often used to demonstrate volume rendering algorithms. This dataset is available with the Stanford VolPack library distribution, or it can be downloaded from the Stanford Computer Graphics Laboratory web page[4]. It is a $256 \times 256 \times 225$ volume of 8-bit values. Because the CT scan contains a lot of background (air), the dataset can be significantly cropped without impinging on the cadaver head itself[5]. We use the Hessian second derivative measure to produce a 256^3 histogram volume which includes first

[4] `ftp://www-graphics.stanford.edu/pub/volpack/data/head/`

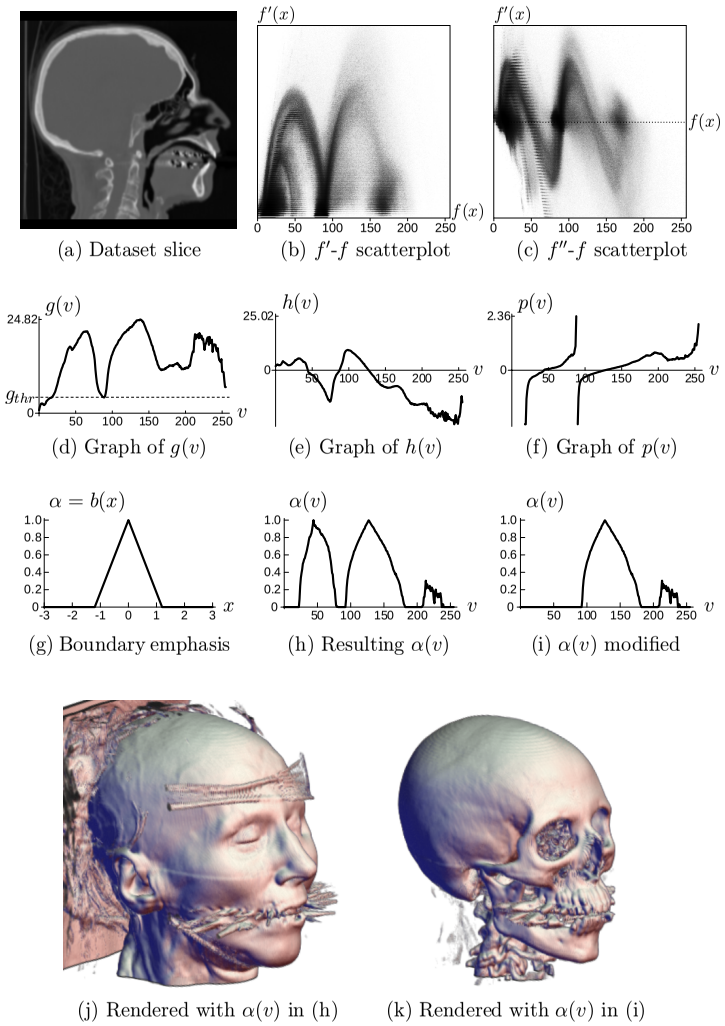
[5] Along the X axis, we use voxels 13 through 196 inclusively; along the Y axis, 2 through 248.

derivative values from zero to 66.0, and second derivative values from -43.0 to 43.0.

Figure 6.8 begins with a slice of the dataset and the two scatterplots. The two boundaries that are most evident from the scatterplots are those between air (around value 15) and soft tissue (value 90), and between soft tissue and bone (value 175). This is confirmed by looking at the plots of $g(v)$, $h(v)$, and $p(v)$ in Figures 6.8(d) through 6.8(f). The nearly ideal shape in the lower segment of $p(v)$'s graph (below value 90) indicates a clean boundary, while the upper segment of $p(v)$'s graph (above value 90) is somewhat distorted.

Still, using a simple boundary emphasis function (Figure 6.8(g)) results in an opacity function with two main peaks which successfully display the air-skin and tissue-bone boundaries. The rendering in Figure 6.8(j) was produced with the opacity function in Figure 6.8(h). Besides seeing the surface of the skin, we also can discern the gauze over the forehead and under the chin that was used to immobilize the head during scanning. The spikes projecting from the mouth are an artifact in the dataset due to dental fillings scattering the X-rays. By removing the lower of the two peaks (Figure 6.8(i)), the surface of the skull becomes visible (Figure 6.8(k)). The third smaller peak did not correspond to any significant boundary.

As was mentioned in Section 4.3, generation of opacity functions is possible from histogram volumes smaller that the 256^3 sizes used elsewhere in this thesis. Figure 6.9 shows the analysis of the CT head using a 64^3 histogram volume. The plots of $g(v)$, $h(v)$, and $p(v)$ in Figures 6.9(a) through 6.9(c) are similar to those for the 256^3 histogram volume (Figures 6.8(d) through 6.8(f)), but they are noticeably smoother. As a result, the generated opacity functions are also smoother, which



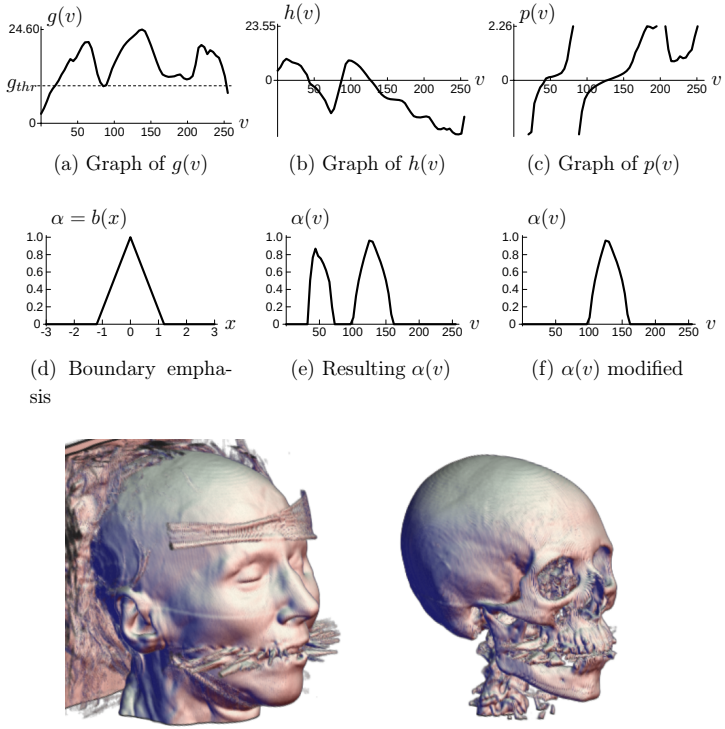
(a) Dataset slice (b) f'-f scatterplot (c) f''-f scatterplot

(d) Graph of $g(v)$ (e) Graph of $h(v)$ (f) Graph of $p(v)$

(g) Boundary emphasis (h) Resulting $\alpha(v)$ (i) $\alpha(v)$ modified

(j) Rendered with $\alpha(v)$ in (h) (k) Rendered with $\alpha(v)$ in (i)

Figure 6.8: CT head analysis and rendering



(a) Graph of $g(v)$ (b) Graph of $h(v)$ (c) Graph of $p(v)$

(d) Boundary emphasis (e) Resulting $\alpha(v)$ (f) $\alpha(v)$ modified

(g) Rendered with $\alpha(v)$ in (e) (h) Rendered with $\alpha(v)$ in (f)

Figure 6.9: CT head analysis and rendering using 64^3 histogram volume

is sometimes a benefit if the rendering algorithm produces artifacts as a result of sharp changes in the opacity function[6]. However, smoother $g(v)$, $h(v)$, and $p(v)$ functions are less accurate with respect to the original dataset, and this can be a very significant problem. For instance, the need for a higher g_{thresh} setting (Figure 6.9(a)) means that the new $p(v)$ function (Figure 6.8(f)) has a steeper slope, and thus the peaks in the new opacity function (Figure 6.9(e)) are narrower than before, even though we have used the exact same boundary emphasis function (Figure 6.9(d)), and nothing has changed about the underlying dataset. In spite of this, the new opacity functions still lead to acceptable renderings of the air-skin boundary (Figure 6.9(g)) and the tissue-bone boundary (Figure 6.9(h)).

6.3.1 Comparison with renderings using Levoy's method

Section 5.4 gave an overview of Levoy's two dimensional opacity functions and compared them to the ones presented in this thesis. We now compare the results of the two methods on the CT head dataset, as this is the only dataset we have renderings of using Levoy's technique. Levoy used the second of his two opacity functions, those designed for visualizing "region boundary surfaces" as described in Section 5.4.2. However, the exact settings he used for the opacity function (that is, the CT numbers he chose for each material), were not described, nor were any of the lighting or shading parameters for the rendering. The images which appeared in his paper, shown in Figure 6.10, will serve as the basis for the comparison.

Before showing renderings using our two dimensional opacity functions, we display in Figure 6.11 the results of the $p(v, g)$ calculation (from Equation 5.12),

[6]For instance, the popular shear-warp volume rendering algorithm suffers from this problem because it relies on *bi*-linear interpolation within the volume instead of *tri*-linear interpolation.

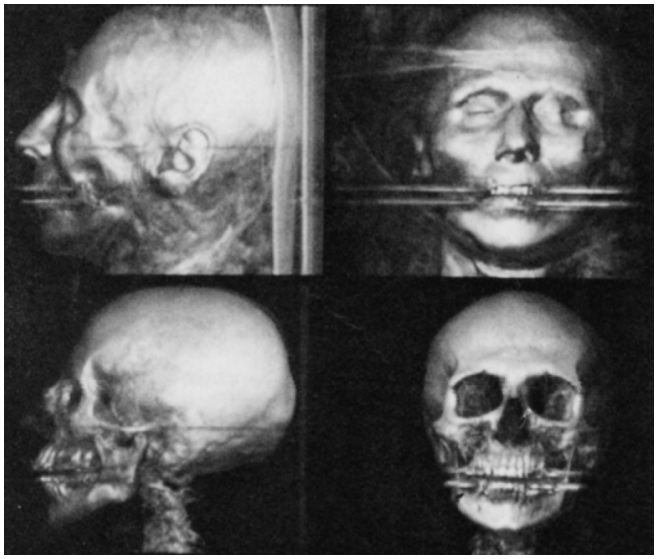


Figure 6.10: Renderings of the CT head from Levoy's paper

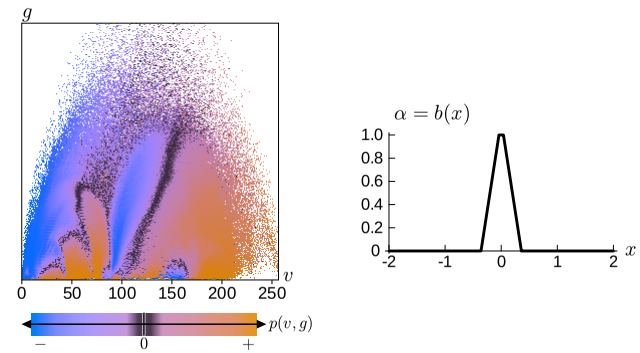


Figure 6.11: Precursors to $\alpha(v, g)$ calculation for CT head. The $p(v, g)$ calculated is shown in (a), and a new boundary emphasis function is shown in (b)

as well as a new boundary emphasis function $b(x)$. The new boundary emphasis function has a narrower peak so that we can better differentiate the regions in $\alpha(v, g)$ that correspond to the different boundaries. The $p(v, g)$ shown is based on a new histogram volume which includes more of the range of gradient values than the one used in the previous figures[7].

Just as inspection of $p(v)$ graphs provided information about the boundaries in a volume which could be displayed with one dimensional opacity functions, the graphical representation of $p(v, g)$ serves as a visual summary of the boundaries which can be rendered using a two dimensional opacity function. For instance, the long black streak near the center of $p(v, g)$ is the indication of the tissue-bone boundary. Bone takes on a range of values in the CT scan because its radio-opacity is not constant. The gradient magnitude at the midpoint of the tissue-

[7]The new histogram volume has a size of 256^3 and includes first derivative values zero through 85.8. The g_{thresh} used for the $p(v, g)$ calculation was zero.

bone boundary varies in proportion with the bone value, since the soft tissue radio-opacity is more constant. Thus the black streak, marking the middle of the boundary, slants towards higher data values as one moves upward along the gradient magnitude axis.

A great deal can be learned about the structure of a dataset through experimentation with the two dimensional opacity function $\alpha(v, g)$ generated from the histogram volume. Figure 6.12 demonstrates this for the main structural components of the CT head by showing opacity functions and the renderings which they produced. Figure 6.12(a) shows the initial $\alpha(v, g)$ as calculated from Equation 5.13. No adjustments were needed to produce the renderings. We can see the gauze strips very clearly, but the sides of the head are obscured. By removing some of the opaque regions (Figure 6.12(b)) from $\alpha(v, g)$ at low data values and low gradient magnitudes, the gauze can be completely removed from the rendering without disturbing the air-skin boundary. The precise editing of $\alpha(v, g)$ needed to achieve this effect was guided by the $p(v, g)$ image. In Figure 6.12(c), the air-skin boundary is removed, revealing the surface of the skull.

So far, we have shown that the semi-automatically generated two dimensional opacity functions succeed in capturing the boundaries in the dataset, and that minimal editing of the opacity function is needed to see the main boundaries. Thus, our method is at least as powerful as Levoy's, because we were able to visualize the same boundaries, but without needing to know the CT values for the materials beforehand. Furthermore, adjustments to the boundary emphasis function can be used for fine control of the rendered boundary appearance.

However, there is one last boundary in this dataset which was never visualized in Levoy's paper. Although disallowed by Levoy's own material adjacency restric-

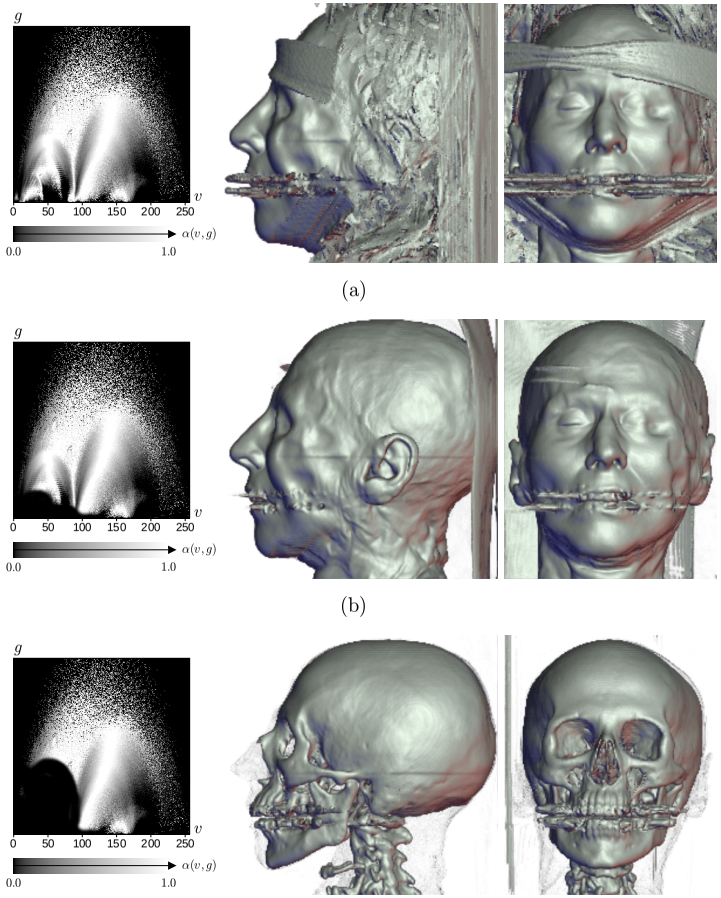
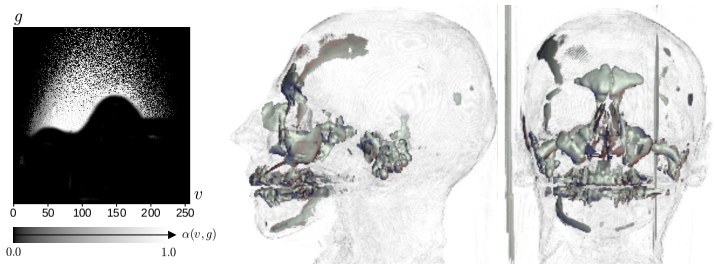


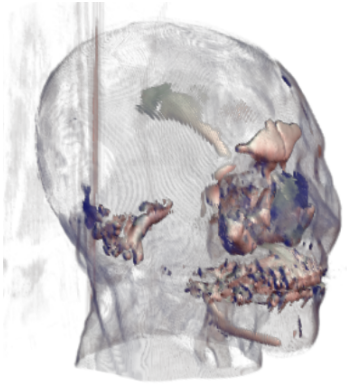
Figure 6.12: Renderings of CT head with two dimensional opacity functions. In (a), the initially generated $\alpha(v, g)$ shows the air-skin boundary. With minor editing (b), the gauze and surrounding material is cleared. Careful removal of the lowest boundary curve region reveals the tissue-bone boundary (c).

tions (Section 5.4.2), there are boundaries between bone and air in the dataset. More precisely, in the skull's various sinus cavities, there are boundaries between bone and air which contain only a thin intermediate layer of mucus membrane. Because of the resolution of this particular CT scan, the measured boundaries of the sinus regions effectively span the data values for air and bone directly, thus the boundary can be selected with the right two dimensional opacity function. The tissue-bone boundary can be removed from the rendering with a simple change in the opacity function (Figure 6.13(a)), revealing the previously unseen bone-air boundary within the skull. Figure 6.13(b) shows the same boundary from another viewpoint, and a transparent layer of the air-skin boundary has re-introduced for reference. The shape of the frontal nasal cavity (above the eyes) is especially easy to discern. Also visible is are the mastoid air cells, a spongy region of the temporal bone in the skull, near the ears[Moo85].

Because of where it lies in (v, g) space, the air-bone boundary can not be visualized with any one dimensional opacity function, since the soft tissue surrounding the skull would obscure it. Also, it can not be visualized with Levoy's method for region boundary display, because he is essentially using a one dimensional opacity function followed with a scaling by the gradient magnitude (recall Figure 5.12). The ability to see the air-bone boundary is more an indication of the potential power of two dimensional opacity functions in general, than it is a specific demonstration of our method's effectiveness. The information captured in the histogram volume, and made visible in $p(v, g)$, provided the guidance in editing the $\alpha(v, g)$ function needed to reveal the boundary.



(a)



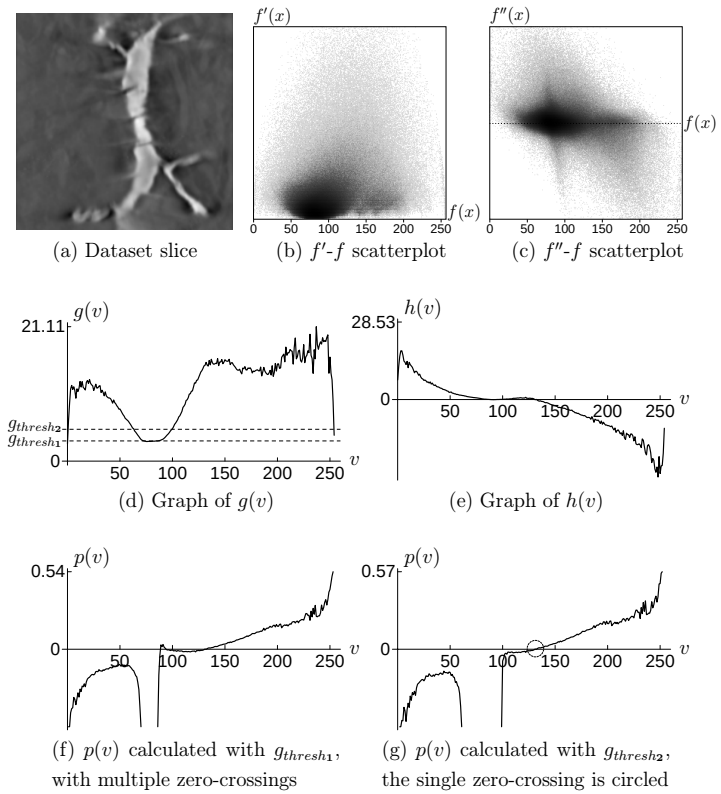
(b)

Figure 6.13: Rendering of CT head dataset displaying the air-bone boundary. (a) shows the opacity function and renderings, using the same viewpoint as in Figure 6.12. The rendering in (b) uses the same viewpoint as in Figures 6.8 and 6.9, and some of the air-skin boundary has been reintroduced.

6.4 Spiny Dendrite (hippocampus): Experimenting with g_{thresh} and $b(x)$ peak width for one dimensional opacity functions

We now analyze the neuron dataset first seen in the comparison of surface versus direct volume rendering (Figure 1.4). The dataset is a tomographic reconstruction of a spiny dendrite on a hippocampal pyramidal neuron from a rat. The specimen is courtesy of Prof. K. Hama of the National Institute for Physiological Sciences, Okazaki, Japan. The National Center for Microscopy and Imaging Research acquired a series of images of the two micron thick specimen with an intermediate high voltage electron microscope. The images were processed with single tilt axis tomography to create a volume of floating point values. This was quantized to eight bits by histogramming the raw data to determine the floating point range that best contained the important values. The histogram volume generated for this dataset has a resolution of 256^3, and was calculated using the Hessian second derivative measure.

As was mentioned in Section 4.4, the scatterplots for this dataset (Figures 6.14(b) and 6.14(c)) do not show clear evidence of any boundaries, as compared to the scatterplots for the other CT datasets analyzed in previous sections. This is also evident from the graphs of $g(v)$ and $h(v)$ (Figures 6.14(d) and 6.14(e)). Instead of having a noticeable maximum at the data value associated with the boundary, and being low elsewhere, the predominant feature of $g(v)$ is its *minimum* at the data value associated with the background value (around 80). The graph of $h(v)$ is also atypical— instead of having one distinct zero-crossing, it lies very close to the v axis along the range of values approximately between 75 to 140.



(a) Dataset slice (b) f'-f scatterplot (c) f''-f scatterplot

(d) Graph of $g(v)$ (e) Graph of $h(v)$

(f) $p(v)$ calculated with g_{thresh_1}, with multiple zero-crossings (g) $p(v)$ calculated with g_{thresh_2}, the single zero-crossing is circled

Figure 6.14: Analysis of hippocampal neuron dataset

We show next the results of two $p(v)$ calculations, based on different choices for the value of g_{thresh}. Figure 6.14(d) illustrates how g_{thresh_1} was chosen to coincide with the average gradient magnitude within the background value. The resulting $p(v)$ (Figure 6.14(f)) has *three* zero-crossings in the data value range from 75 to 140, theoretically indicating the presence of three boundaries in the volume. Yet the cross-section shows that there are basically just two types of material in the dataset, background and dendrite, and we are interested in the (single) boundary between the two. The problem is that the histogram volume did not succeed in properly measuring the soft boundary that exists between the dendrite and the background. As a result, straight-forward application of Equation 5.10 with what appears to be the correct g_{thresh} did not create a position function $p(v)$ which is suitable for further analysis.

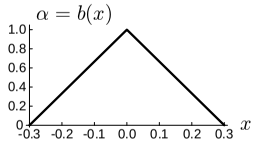
However, we can choose a g_{thresh} so as to create a position function $p(v)$ with a single zero-crossing. Raising g_{thresh} tends to stretch the graph of $p(v)$ away from the v axis, eliminating multiple zero-crossings. Using the g_{thresh_2} indicated in Figure 6.14(d) leads to the position function $p(v)$ seen in Figure 6.14(g). The new $p(v)$ has only one zero-crossing, which is circled for clarity.

This kind of experimentation with g_{thresh} should be explained, since it may appear to be somewhat ad hoc. As one can see from the scatterplot of data value and second derivative (Figure 6.14(c)), there is a general tendency for the second derivative to decrease as data value increases. This property is better represented by the fainter pattern of voxels with very high and very low second derivatives (located in the upper left and lower right portions of the scatterplot, respectively), than by the much larger number of voxels clustered along the region of near-zero second derivative values. Looking at the second derivative scatterplot, one can

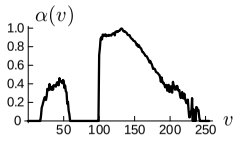
visually interpolate between the faint distribution of voxels with very high and very low second derivative values in order to find the approximate zero-crossing of this faint pattern. In our experience, the single zero-crossing in $p(v)$ that is revealed by increasing g_{thresh} is well-correlated with the zero-crossing in the scatterplot. We now proceed with the second stage of opacity function generation, by experimenting with different boundary emphasis functions.

For an initial $b(x)$ we start with a tent function centered at zero, spanning 0.3 voxels on either side, seen in Figure 6.15(a). The reason for the small scale in the domain of $b(x)$ is that the non-ideal shape of the $g(v)$ and $h(v)$ graphs led to a very inaccurate estimation of the boundary blurring parameter σ, calculated to be 0.45 with Equation 5.8. This implies a boundary thickness of $2\sigma = 0.9$, but inspection of dataset cross-sections would suggest the thickness is closer to 2.5. Still, the initial $b(x)$ is too wide, because the resulting opacity function (Figure 6.15(b)) is non-zero for too many different values, so the rendering (Figure 6.15(c)) shows no clear structure. This can be improved by narrowing the peak in $b(x)$ (Figure 6.15(d)), which similarly narrows the peak in $\alpha(v)$ (Figure 6.15(e)), and the new rendering (Figure 6.15(f)) is somewhat clearer.

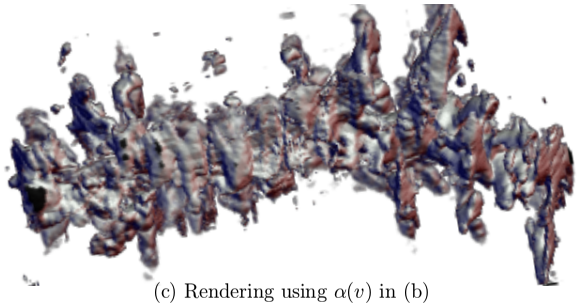
Continuing to make the $b(x)$ peak narrower (Figure 6.16(a)) leads to an even narrower peak in $\alpha(v)$ (Figure 6.16(b)), and finally the rendering (Figure 6.16(c)) shows the surface structure of the neuron. Again, this process was not as automatic as would be ideal, but to arrive at this rendering (starting with the $p(v)$ calculated previously) required empirically decreasing only one variable, the width of the triangular peak in $b(x)$. From here, we can further decrease the amount of "haze" surrounding the neuron by moving the peak in $b(x)$ slightly to the right, to slightly



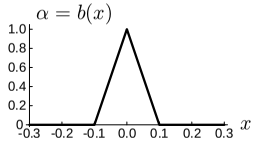
(a) Boundary Emphasis



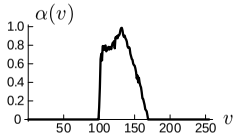
(b) Resulting $\alpha(v)$



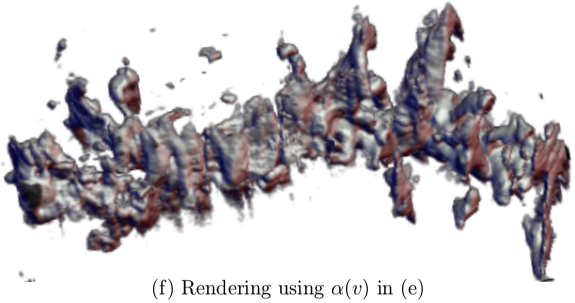
(c) Rendering using $\alpha(v)$ in (b)



(d) Boundary Emphasis

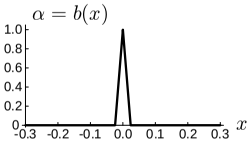


(e) Resulting $\alpha(v)$

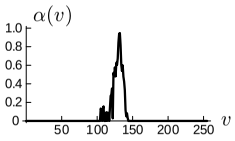


(f) Rendering using $\alpha(v)$ in (e)

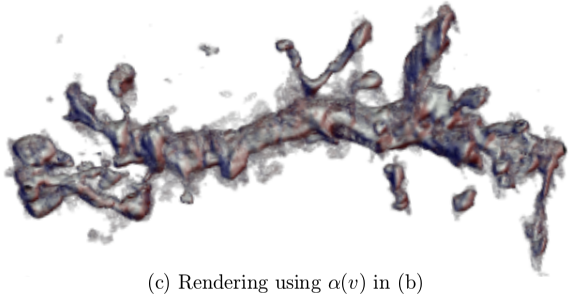
Figure 6.15: Renderings of hippocampal neuron dataset



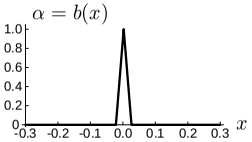
(a) Boundary Emphasis



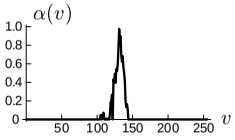
(b) Resulting $\alpha(v)$



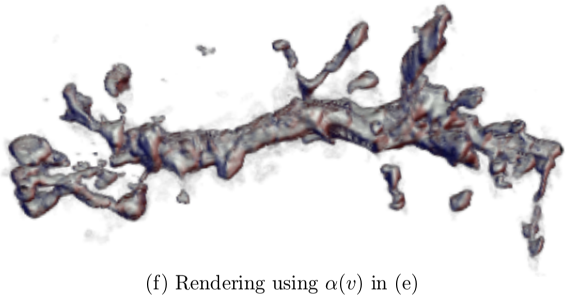
(c) Rendering using $\alpha(v)$ in (b)



(d) Boundary Emphasis



(e) Resulting $\alpha(v)$



(f) Rendering using $\alpha(v)$ in (e)

Figure 6.16: Renderings of hippocampal neuron dataset, continued

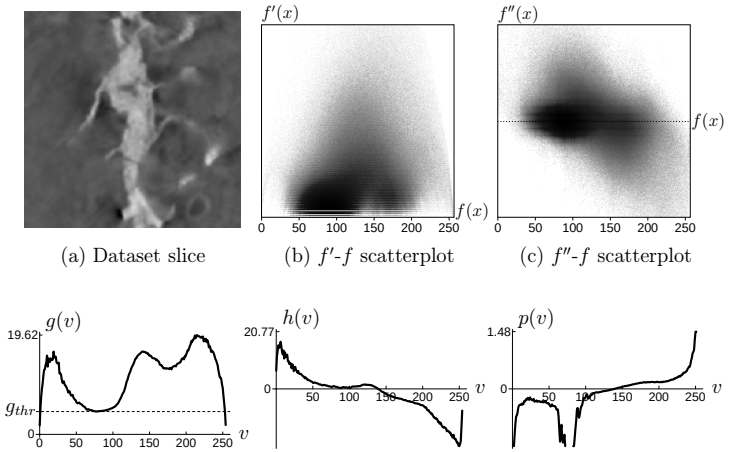
higher position values (Figure 6.16(d))[8]. The peak in the new opacity function moves in the same direction, putting greater emphasis on higher data values, so as to emphasize the boundary region closer to the interior of the neuron. The change in the resulting opacity function is slight (Figure 6.16(e)), but there is less semi-transparent material surrounding the neuron in the rendering (Figure 6.16(f)).

6.5 Spiny Dendrite (neostriatum): Comparison of one and two dimensional opacity functions, including Levoy's

We finish with a second spiny dendrite dataset from the National Center for Microscopy and Imaging Research, a tomographic reconstruction of part of a dendrite from the neostriatum of a rat's brain. The specimen was provided by Drs. C. Ingham and G. Arbuthnott (University of Edinburgh, Scotland) and C. Wilson (University of Tennessee, Memphis). The tomographic reconstruction and numerical conversions used with this dataset are the same as those used in Section 6.4

Figure 6.17 starts with a cross-section of the dataset and continues with the various analysis results. The cross-section (Figure 6.17(a)) indicates that this dataset is similar to the one analyzed and rendered in the previous section— there are significant variations in the data value within the background and dendrite caused by inconsistent radio-opaque dye absorption and tomographic artifacts. This is confirmed by inspection of the scatterplots (Figures 6.17(b) and 6.17(c)), which do not clearly contain the characteristic boundary curves. The graphs of $g(v)$, $h(v)$, and $p(v)$ (Figures 6.17(d), 6.17(e), 6.17(f)) are similar to those for

[8]The peak in $b(x)$ is centered at 0.0025 instead of 0.0.



(a) Dataset slice (b) f'-f scatterplot (c) f''-f scatterplot

(d) Graph of $g(v)$ (e) Graph of $h(v)$ (f) Graph of $p(v)$

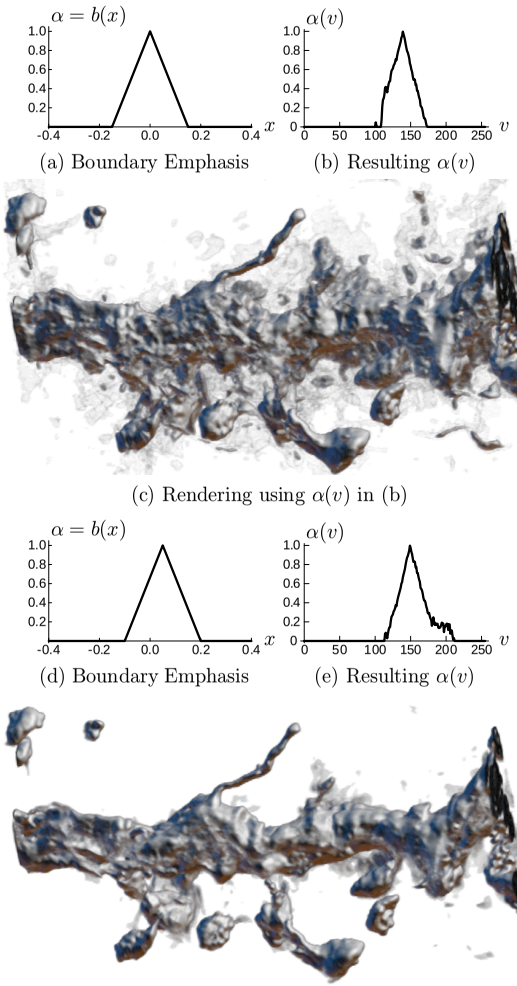
Figure 6.17: Analysis of neostriatum neuron dataset

the previous dendrite dataset, although with this dataset experimentation with g_{thresh} was not needed. Setting g_{thresh} to the average gradient magnitude within the background results in a position function $p(v)$ with only a single zero crossing, at data value 140.

Figure 6.18 show two renderings of the dataset based on slightly different boundary emphasis functions. Some experimentation produced the boundary emphasis function seen in Figure 6.18(a), which led to the opacity function in Figure 6.18(b) and the rendering in Figure 6.18(c). Although the rendering shows the surface of the dendrite fairly well, we can reduce the amount of surrounding haze by moving the spike in the boundary emphasis function to the right— to higher position values (Figure 6.18(d))— which in turn causes the spike in the opacity function (Figure 6.18(e)) to move to higher data values. The new rendering is clearer, although at the expense of having the thin features in the dendrite shrink in size.

This dataset can also be used to demonstrate the effectiveness of two dimensional opacity functions. This starts with the two dimensional position function $p(v, g)$, shown with a colormap in in Figure 6.19(a). The important feature in this image is the vertical gray band indicating an average second derivative near zero, signifying the middle of a boundary. The gray band being vertical indicates a consistent correlation with one data value, approximately 140, the same data value at which the one dimensional position function $p(v)$ had its zero-crossing.

The initial two dimensional opacity function $\alpha(v, g)$ shown in Figure 6.19(b) was generated from the same boundary emphasis function and g_{thresh} as used to generate the one dimensional opacity function in Figure 6.18(e). The rendering in Figure 6.19(d) indicates that this opacity function left some of the background near



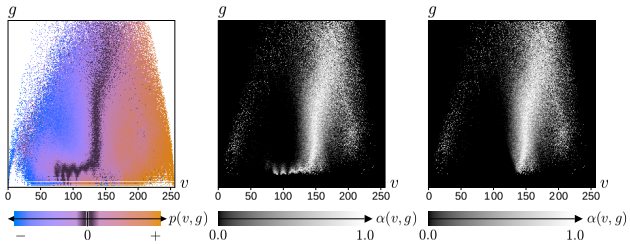
(a) Boundary Emphasis (b) Resulting $\alpha(v)$

(c) Rendering using $\alpha(v)$ in (b)

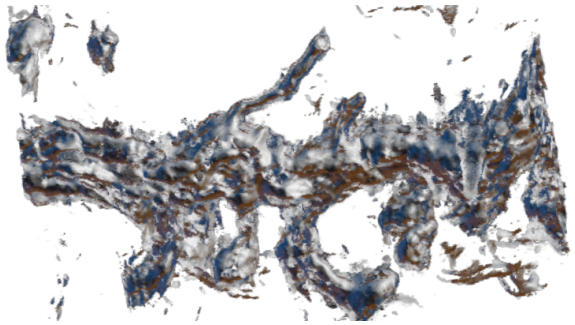
(d) Boundary Emphasis (e) Resulting $\alpha(v)$

(f) Rendering using $\alpha(v)$ in (e)

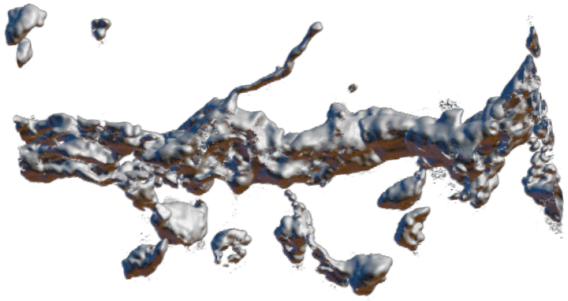
Figure 6.18: Renderings of neostriatum neuron dataset with one dimensional opacity functions



(a) Colormapped $p(v, g)$ (b) Initial $\alpha(v, g)$ (c) Edited $\alpha(v, g)$



(d) Rendering using $\alpha(v, g)$ in (a)



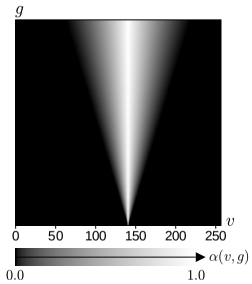
(e) Rendering using $\alpha(v, g)$ in (b)

Figure 6.19: Renderings of neostriatum neuron dataset with two dimensional opacity functions

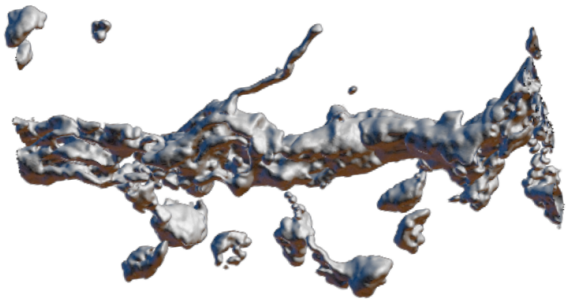
the dendrite opaque. Although the purpose of the g_{thresh} modification in the definition of the position function (Equation 5.12) was to eliminate the contribution of background values to the opacity function (so as to remove them from the rendering), this is difficult with two dimensional opacity functions because there can be a wide range of gradient magnitudes within the background. Simply increasing g_{thresh} (as was done in the previous section) to the highest background gradient magnitude is not helpful with the dendrite datasets. Doing so would eliminate significant portions of the dendrite boundary from the rendering, because where the boundary is very gradual, the gradient magnitude can be lower than g_{thresh}.

Instead, the flexibility of two dimensional opacity functions allows us to remove the contribution of the high gradients for only the data values associated with the background, leaving the dendrite boundary unaffected. This modification to $\alpha(v, g)$ is shown in Figure 6.19(c), and the resulting rendering is shown in Figure 6.19(e). The boundary is now very uniformly clear and distinct, compared to the best rendering obtained with a one dimensional opacity function. Whether or not the dendrite should have a more "fuzzy" appearance where its measured boundary is less distinct, a natural result of using a one dimensional opacity function, is a choice available to the user.

As a final comparison with Levoy's two dimensional opacity functions, we now render the same dataset with the first of Levoy's opacity functions, the one designed for visualizing the isovalue contours in smoothly varying data. Strictly speaking, this is a misapplication of the opacity function, because it was not intended for the display of material surfaces. However, the large variations in the data value and boundary characteristics in the neuron datasets make its use more appropriate. The two main parameters for the opacity function seen in Figure 6.20(a) —



(a) Levoy's $\alpha(v, g)$



(b) Resulting rendering

Figure 6.20: Rendering of neostriatum neuron dataset with Levoy's two dimensional opacity function

the width of the "V" in (v, g) space and the data value at which to center it — were set by trying to match the appearance of the semi-automatically two dimensional opacity function shown in Figure 6.19(c). In fact, the resulting rendering in Figure 6.20 looks very similar to Figure 6.19(e), although the rendered surface is slightly cleaner with Levoy's opacity function.

This effectively demonstrates that for datasets where there is a single boundary that has been imperfectly measured, Levoy's opacity function can be useful, provided that appropriate settings for its parameters are known. However, Levoy's method still has an element of trial and error, because nothing about the interface would indicate if there are multiple boundaries, and the single boundary rendered by this method may be obscuring other boundaries within. Normally, the user would still have to explore the parameter space of this opacity function in order to find the best rendering. Thus, even when the semi-automatically generated opacity functions are not actually used in the rendering process, they give information about all the boundaries present in the volume. This information can be used to guide the user towards appropriate settings in the use of other types of opacity functions, such as Levoy's.

Chapter 7

Conclusions and future work

In the volume rendering community, the problem of transfer function specification is acknowledged to be one of the remaining hard problems that needs to be solved before direct volume rendering gains widespread acceptance in a variety of disciplines. This thesis has presented a solution to this problem for those situations where the goal of the rendering is to visualize material boundaries in scalar volume datasets, with transfer functions which assign opacity only. This solution is based on the "histogram volume", a data structure which measures the position-independent relationship between the data value and its derivatives throughout the volume.

At a general level, the histogram volume allows opacity function generation to be separated into two steps— the first automatic, the second controlled by the user. After the histogram volume has been created, it contains information which permits the automatic calculation of the *position function*, mapping from data value to a signed distance to the boundary center. The user can then create the *boundary emphasis function* to map from this spatial domain to opacity. The composition of these two steps is an opacity function. Fortunately, the second

mapping is easy to create, since the user is controlling only the character of the rendered boundary, while the difficult portion— detecting and locating the boundaries in the data value domain— has been done automatically with the histogram volume. With the techniques presented here for semi-automatic opacity function generation, a large amount of guesswork has been removed from the process, and the interface has been simplified and made more intuitive.

This thesis has also demonstrated the usefulness of two dimensional opacity functions, first proposed by Levoy in 1988 [Lev88] but used only infrequently since then. Fortunately, the same interface for setting the boundary emphasis function can be used to control either one or two dimensional opacity functions with equal ease.

The histogram volume analysis which is used to generate the position function is based on a certain mathematical boundary model, but there are other possible ways of analyzing the histogram volume to extract the necessary information. We have shown that for every boundary that occurs in a dataset, there will be a characteristic curve in the histogram volume. Therefore, it would be fitting to use an object recognition technique from computer vision, such as the Hough transform, to find those curves wherever they appear in the histogram volume. The use of the Hough transform for this purpose was briefly explored, but concerns about its computational cost, mathematical complexity, and the correctness of its implementation, led to its being abandoned in favor of the much simpler techniques. Nonetheless, the Hough transform and related methods still warrant further exploration.

One important possible improvement to the presented technique is better modeling of the ideal boundary. Each imaging modality, such as CT and MRI, has a

characteristic frequency response, which in turn determines the nature of measured boundaries. We assumed a Gaussian frequency attenuation which made the computation of the position function extremely easy. However, a more accurate mapping from data value to position should be possible with more sophisticated models of frequency response, including those that model directional variation in frequency response, a characteristic of all standard scanners. This may necessitate measuring different kinds of derivatives, or recording them in other formats than a histogram volume, but the mapping from position to opacity could still be done with boundary emphasis functions.

There is also at least one interesting generalization of our method. MRI scanners can measure many different quantities (proton density, spin decay rate, diffusion, etc), and many MRI datasets contain vector information, with two, three, or more separate measurements at each voxel. Within each material, these separate data values will be roughly constant. Because this characteristic of *material uniformity* was the main driving assumption in the development of the techniques in the context of scalar data, it should be possible to extend them to non-scalar data which shares the characteristic. Though the data structures and methods of derivative measurement would be different, the algorithms and results would be largely the same: all the boundary regions in the volume would be made visible more easily and the user would again enjoy high-level control over the rendered boundaries. There would be the additional benefit of robustness in the boundary detection, because the spatially coincident boundaries in all the data channels would reinforce one another.

Besides these fundamental changes and generalizations of the algorithm, there are a number of smaller modifications which deserve exploration. There is room

for improvement in how the directional derivatives are measured. The histogram volume analysis stage relies on a method for collapsing either an entire slice or a scanline of the histogram volume down to one value. Currently an *average* value is used, but it might be advantageous to use another measure, such as mode or median. Finally, in its current form, the algorithm can not account for noise in the data besides establishing a simple threshold mechanism which allows a crude compensation for the ambient gradient magnitude caused by noise-induced fluctuations in data value. Better noise handling, either as a pre-process filtering or an integral part of the derivative measurement is certainly an area for further investigation.

Direct volume rendering has proven itself to be a potentially very valuable tool in a wide variety of applications. The power of the tool is currently being limited by the difficulties faced by its users in trying to set the transfer function. This thesis has offered a solution to the problem for rendering an extremely important class of scalar volume datasets— those in medical and biological applications— where the goal is visualizing the structure of material boundaries. In keeping with the spirit of direct volume rendering, whereby the data values map to the rendered image without any geometric pre-computation or algorithmic alteration, this method makes the boundaries visible based on a fundamental relationship measured in the dataset itself. The position-independent relationship between the data value and its first and second derivatives allows for two advancements: the automatic detection of the boundaries present in the volume, as well as an intuitive interface for controlling how those boundaries are rendered.

The traditional trial-and-error path to a useful transfer function has been replaced by a novel and powerful high-level interface for transfer function editing.

The core idea is to let the *data* help the *user* find the transfer function. Coupling the interface for transfer function specification to the underlying dataset guarantees some appropriateness of the transfer function for the task of visualizing the structure of material boundaries. It is the author's hope that this idea will have relevance to other visualization applications where the promise of direct volume rendering has yet to realized.

Appendix A

Bandlimiting and the "thickness" of boundaries

A.1 Bandlimiting and thickness in terms of σ

Our goal here is to show the relationship between the boundary's frequency content and its thickness, and how this relationship effects our ability to record the presence of the boundary in the histogram volume. The Fourier transform is an important part of this analysis, and because there are variety of possible definitions for the transform, we state the particular definitions being used here:

$$F(\omega) = \mathcal{F}(f)(\omega) \equiv \int_{-\infty}^{\infty} f(x) e^{-i\omega x}\, dx \tag{A.1}$$

$$f(x) = \mathcal{F}^{-1}(F)(x) \equiv \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega x}\, d\omega \tag{A.2}$$

We will use the following to model the data value as a function of position within a boundary region:

$$f(x) = \frac{1 + \mathrm{erf}(\frac{x}{\sigma\sqrt{2}})}{2} \tag{A.3}$$

Recall from Section 5.2 that the "thickness" of a boundary is defined to be 2σ. Setting $v_{min} = 0$ and $v_{max} = 1$ in the the formula for $f(x)$ given in Section 5.1 produces Equation A.3. Any other values for the variables v_{min} and v_{max} represent some scaling or shifting of the boundary, and the Fourier transform is not meaningfully changed by these operations.

To find the Fourier transform of the $f(x)$ given above, we could simply apply the definition of the Fourier transform to $f(x)$, and then reduce the resulting expression. We can avoid this labor, however, by exploiting a useful property of the transform under integration, expressed here with $\overset{\mathcal{F}}{\longleftrightarrow}$ pointing to the two expressions which are transforms of each other:

$$g(x) \overset{\mathcal{F}}{\longleftrightarrow} G(\omega) \;\Rightarrow\; \int_{-\infty}^{x} g(y)dy \overset{\mathcal{F}}{\longleftrightarrow} \frac{G(\omega)}{i\omega} + \pi G(0)\delta(\omega) \tag{A.4}$$

This states, in terms of a function $g(x)$ and its Fourier transform $G(\omega)$, how to find the transform of an integral if we already know the transform of the function being integrated. Aside from a "DC" term $\pi G(0)\delta(\omega)$ (which we will henceforth ignore[1]), when we integrate a function, the corresponding transform is divided by $i\omega$. We will use Equation A.4 to find the transform of $f(x)$ because doing so will bring us directly in contact with the attenuation in frequency space caused by the bandlimiting inherent in the measurement.

In order to exploit the property expressed in Equation A.4, we find the derivative of $f(x)$:

$$f'(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-x^2/2\sigma^2} \tag{A.5}$$

[1]The DC (direct current) component of the Fourier transform represents the average value of the function. Since all the relevant characteristics of the boundary's Fourier transform are in terms of non-zero frequencies, the DC component is not important for this analysis.

By our choice of $f(x)$, $f'(x)$ is a normalized Gaussian function with standard deviation σ. Its Fourier transform is[KK68]:

$$\mathcal{F}(f')(\omega) = e^{-\sigma^2\omega^2/2} = e^{-\omega^2/2(1/\sigma)^2} \tag{A.6}$$

Thus the transform of a Gaussian function is another Gaussian function, and there is an exact reciprocal relationship between their standard deviations. While the standard deviation in $f'(x)$ is σ, in $\mathcal{F}(f')(\omega)$ it is $1/\sigma$. Having $f'(x)$ and its transform $\mathcal{F}(f')(\omega)$, we can now use Equation A.4 to state the transform of the measured boundary function:

$$\mathcal{F}(f)(\omega) = \frac{1}{i\omega} e^{-\sigma^2\omega^2/2} \tag{A.7}$$

It is possible to utilize the same process to find the transform of the un-blurred boundary function, which represents the boundary prior to the measurement process. This is the so-called Heaviside step function $h(x)$:

$$h(x) = \begin{cases} 0 & x < 0 \\ 1/2 & x = 0 \\ 1 & x > 0 \end{cases} \tag{A.8}$$

By definition, the derivative of the step function is the delta function $\delta(x)$. The Fourier transform of the delta function is trivial to compute:

$$\delta(x) \overset{\mathcal{F}}{\longleftrightarrow} 1 \tag{A.9}$$

Then by again using Equation A.4 we can find the transform of the step function:

$$h(x) \overset{\mathcal{F}}{\longleftrightarrow} \frac{1}{i\omega} \tag{A.10}$$

Since we know from Fourier theory that the spectrum of the convolution of two functions is exactly the product of the two functions' spectra, we see from the expression for $\mathcal{F}(f)(\omega)$ (Equation A.7) that the spectrum of the measured boundary

is the product of the spectrum of the step function $(1/i\omega)$ times a Gaussian function $e^{-\sigma^2\omega^2/2}$. Thus, $e^{-\sigma^2\omega^2/2}$ is precisely the attenuation in frequency caused by the bandlimiting in the measurement process. We can now state the relationship between measured boundary thickness and bandlimiting:

> The boundary thickness will be 2σ when the bandlimiting due to measurement is equivalent to multiplication in frequency space by a Gaussian function with standard deviation $1/\sigma$.

A.2 Bandlimiting and thickness in the context of the Nyquist criterion

Because neither $1/i\omega$ nor $e^{-\sigma^2\omega^2/2}$ ever fall to zero as ω approaches infinity, the boundary spectrum $\mathcal{F}(f)(\omega)$ will always have non-zero magnitude . Also, due to the $1/i\omega$ factor, $\mathcal{F}(f)(\omega)$ is infinitely high near $\omega = 0$, and is thus not integrable. Thus, it is not possible to discuss what fraction of the boundary spectrum falls below the Nyquist frequency for a given Gaussian attenuation. Rather, it only makes sense to discuss the Nyquist criterion in relation to the Gaussian attenuation itself. The relevant attribute of the Gaussian function is its width, since this gives an indication of the range of frequencies which pass through the measurement process. As the Gaussian function is always positive, we must contrive some measure of its width. A common engineering practice is to measure the "Full Width Half Max"—the horizontal width of the function's graph at half of its maximum height. For a Gaussian with standard deviation η, the the FWHM is $2\eta\sqrt{2\ln 2}$ [KK68]. As the measurement attenuation Gaussian has standard deviation $1/\sigma$, its FWHM is $2\sqrt{2\ln 2}/\sigma$.

One plausible way relate the boundary thickness to the Nyquist rate is to set the FWHM of the Gaussian function in frequency space to equal the range of frequencies which satisfy the Nyquist criterion. If we specify that the distance between data sample points is one unit (that is, the length of a voxel's edge is one), then the Nyquist frequency is π radians per sample, and the range of frequencies which satisfy the Nyquist criterion are from $-\pi$ to π. Then we have:

$$\frac{2\sqrt{2\ln 2}}{\sigma} = 2\pi \tag{A.11}$$

Solving Equation A.11 for σ to find the boundary thickness 2σ yields:

$$\frac{\sqrt{2\ln 2}}{\sigma} = \pi \;\Rightarrow\; \sigma = \frac{\sqrt{2\ln 2}}{\pi} \;\Rightarrow\; \textit{thickness} = 2\sigma = \frac{2\sqrt{2\ln 2}}{\pi} \approx 0.74956 \tag{A.12}$$

Thus, the boundary thickness will be about 75% of the distance between sample points. For comparison, if we had set the FWHM of the Gaussian attenuation to fit in half the Nyquist range ($-\pi/2$ to $\pi/2$), then the corresponding boundary thickness would be twice as large, approximately 150% of the inter-sample distance.

It may seem that this boundary thickness will not be large enough to be detected by the histogram volume, if at any given location on the object surface only about one voxel can fit within the boundary region. However, we are helped by the fact that significant objects in a dataset tend to have large surface areas, and that the surfaces tend to take on a variety of orientations and positions relative to the sampling grid. To some extent this was illustrated with Figure 4.2 in the discussion of histogram volume formation.

At this point, however, we are also equipped to make a more quantitative probabilistic argument. If an object measured in a volume dataset has surface area A, the boundary region surrounding the object will have a *volume* approximately equal to $2A\sigma$, since the boundary thickness is 2σ. Assuming that the dataset

sample points are distributed evenly throughout the volume inside the boundary region, the amount of the boundary region volume gives the approximate number of sample points within the boundary region. Thus, if the Gaussian attenuation in frequency space has a standard deviation $1/\sigma$, we can compute an approximate number of hits along the the curves in the histogram volume:

$$\textit{approx. number of hits along histogram curves} = 2A\sigma \tag{A.13}$$

This relationship explains why the scatterplot curves for small surface area boundaries were lighter than those for larger surface area boundaries. We can also do a "back of the envelope" calculation to show that having the boundary thickness be only 1.0 is not undetectably thin. Suppose we have a 128^3 dataset which contains a sphere with a diameter 64 samples across. Its surface area is then $4\pi r^2$, and with $r = 32$ and thickness $= 1$, this implies about $12,868$ samples within the boundary region, representing sufficient hits along the boundary curve in the histogram volume.

Appendix B

Generating masks for volume derivative measurement

B.1 One dimensional masks from reconstruction filters

Figure 4.3 demonstrated the use of a simple mask which was used to measure the first derivative in one dimensional data. This appendix will describe a process which produces masks for measuring derivatives in one dimension, as well as how to generalize one dimensional masks to measure first and second partial derivatives in three dimensions[1].

It was stated in Chapter 4 that reconstruction of a continuous data value function between sample points in the volume dataset is not necessary for measuring

[1] In Chapter 3 we employed an abuse of terminology, using "first derivative" for "first directional derivative along the gradient direction". In this section we return to the strict usage; "first derivative" means the first derivative of a function of one variable (and likewise for a second derivative).

any of the quantities needed in the histogram volume creation. Instead, discrete masks can be used to measure the first and second partial derivatives which are needed for the directional derivative calculations. Although we will not need to explicitly reconstruct the data value anywhere, a simple way to derive the necessary masks is through consideration of reconstruction filters and convolution.

Given a sequence of sampled data points v_i, the creation of a continuous function which interpolates smoothly between them starts with a train of delta function spikes at unit intervals, scaled by the original data values:

$$g(x) = \cdots + v_{-1}\delta(x+1) + v_0\delta(x) + v_1\delta(x-1) + v_2\delta(x-2) + \cdots \tag{B.1}$$

A continuous reconstructed signal can then be obtained by convolving $g(x)$ with a continuous kernel $h(x)$:

$$(g \star h)(x) = \cdots + v_{-1}h(x+1) + v_0h(x) + v_1h(x-1) + v_2h(x-2)\cdots \tag{B.2}$$

Copies of the kernel are effectively placed at each sample point and scaled by the corresponding data value. The first derivative of the reconstructed function is easily computed, by the linearity of taking derivatives:

$$\begin{aligned}
&\frac{d(g \star h)}{dx}(x) \\
&= \cdots + v_{-1}\frac{dh}{dx}(x+1) + v_0\frac{dh}{dx}(x) + v_1\frac{dh}{dx}(x-1) + v_2\frac{dh}{dx}(x-2)\cdots \\
&= \cdots + v_{-1}h'(x+1) + v_0h'(x) + v_1h'(x-1) + v_2h'(x-2)\cdots \\
&= [\cdots\ v_{-1}\ v_0\ v_1\ v_2\ \cdots]^T[\cdots\ h'(x+1)\ h'(x)\ h'(x-1)\ h'(x-2)\ \cdots]
\end{aligned} \tag{B.3}$$

Equation B.3 defines the infinite sum as the dot product of two vectors, one listing the data values v_i, and one containing evaluations of the derivative of the kernel h. If the position x where we calculate the derivative of the reconstructed function

happens to be an integer k, we can express this as:

$$\begin{aligned}
&\frac{d(g \star h)}{dx}(k) = \\
&\quad [\cdots\ v_{k-2}\ v_{k-1}\ v_k\ v_{k+1}\ v_{k+2}\ \cdots]^T[\cdots\ h'(2)\ h'(1)\ h'(0)\ h'(-1)\ h'(-2)\ \cdots]
\end{aligned} \tag{B.4}$$

Assuming that the reconstruction kernel h has finite support, there can only be a finite number of non-zero values in the vector $[\cdots h'(2)\, h'(1)\, h'(0)\, h'(-1)\, h'(-2)\cdots]$. The finite vector containing all the non-zero values is a *mask* for measuring the first derivative of regularly sampled data. Since most reconstruction kernels are even functions ($h(-x) = h(x)$), the mask will have an odd number of values, with $h'(0)$ at the center position. Equation B.4 shows how a mask is used: after centering the mask at the data point of interest (v_k), the products between corresponding data and mask values are summed to produce the final measurement.

Given any continuous reconstruction kernel $h(x)$, we can evaluate its first derivative at integer positions to generate a first derivative mask. Of course, the first derivative has to exist at the integer positions in order for this to be valid. As an example, we take the Catmull-Rom cubic spline $c(x)$:

$$c(x) = \begin{cases} \frac{3}{2}|x|^3 - \frac{5}{2}|x|^2 + 1 & \text{for } |x| \le 1 \\ -\frac{1}{2}|x|^3 + \frac{5}{2}|x|^2 - 4|x| + 1 & \text{for } 1 < |x| \le 2 \\ 0 & \text{otherwise} \end{cases} \tag{B.5}$$

One can verify that $c'(x)$ is continuous; evaluating $c'(x)$ at the integers yields:

$$[\cdots\ \ c'(2)\ \ c'(1)\ \ c'(0)\ \ c'(-1)\ \ c'(-2)\ \ \cdots] = [\cdots\ \ 0\ \ -0.5\ \ 0\ \ 0.5\ \ 0\ \ \cdots] \tag{B.6}$$

Thus the Catmull-Rom kernel generates the first derivative mask $[-0.5\ \ 0\ \ 0.5]$.

In the same way that first derivative masks were derived above, masks for measuring the *second* derivative of regularly sampled data can be created from a

kernel $h(x)$:

$$[\cdots \quad h''(2) \quad h''(1) \quad h''(0) \quad h''(-1) \quad h''(-2) \quad \cdots]$$

In this case, the second derivative of the reconstruction kernel must exist at all integer locations for the mask to be valid. For instance, by differentiating $c(x)$ twice, one will find that the Catmull-Rom kernel does not generate a second derivative mask because its second derivative is discontinuous at ± 1 and ± 2.

B.2 Two common masks and a kernel which generates them

As was noted in Section 4.3, a common mask for measuring the first derivative is $[-0.5 \quad 0 \quad 0.5]$, called the "central difference" method. The name arises from the derivative approximation for a function $s(x)$:

$$\begin{aligned} s'(x) &\approx \frac{s(x+d) - s(x-d)}{2d} \\ &= \frac{1}{2d}s(x+d) - \frac{1}{2d}s(x-d) \\ &= [s(x-d) \quad s(x) \quad s(x+d)]^T \, [\frac{1}{2d} \quad 0 \quad -\frac{1}{2d}] \end{aligned} \tag{B.7}$$

Setting $d = 1$ gives the $[-0.5 \quad 0 \quad 0.5]$ mask. In the interests of implementation simplicity and computational speed, this thesis adopts central differences for the first derivative mask. In the context of volume rendering, central differences are often used to calculated the gradient magnitude as part of the shading calculations [LCN98]. There are more sophisticated derivative measurement techniques available [MMMY96], but a thorough exploration of them is tangential to the goal of this thesis.

There is another simple and common mask for our second derivative measurement, $[1 \quad -2 \quad 1]$, sometimes called the "second central difference" method. The name arises from the fact that the mask can be derived by twice applying the method of central differences (now with $d = 1/2$), as follows:

$$\begin{aligned} s''(x) &\approx s'(x + \frac{1}{2}) - s'(x - \frac{1}{2}) \\ &\approx s(x+1) - s(x) - (s(x) - s(x-1)) \\ &= s(x+1) - 2s(x) + s(x-1) \\ &= [s(x-1) \quad s(x) \quad s(x+1)]^T \, [1 \quad -2 \quad 1] \end{aligned} \tag{B.8}$$

This mask does not respond to sequences of constant or linearly changing values. Its result is non-zero only when the sample data point positioned at the middle of the mask is higher or lower than the average of its two neighbors. This thesis uses $[1 \quad -2 \quad 1]$ for the second derivative mask, again in the interests of implementation simplicity and computational speed.

There is an important theoretical question regarding the validity of using these two masks in combination for the histogram volume computation. Recall from Chapter 3 that we traced along an ideal boundary to reveal a characteristic relationship between the data value and its first and second derivatives, as illustrated in Figure 3.8. The point $(f(x), f'(x), f''(x))$ was plotted for each position x along the boundary in order to form the three dimensional curve which indicates the presence of a boundary. For such a case, the boundary function $f(x)$ studied was a continuous function with analytically known derivatives, thus reconstruction was not an issue.

However, when we are measuring derivatives of sampled data with masks, we are implicitly reconstructing the function. That is, the result of using a mask

is mathematically equivalent to convolving the sample points with a continuous kernel and analytically finding the derivative of the reconstructed function. Thus, there is a basic incompatibility if there does not exist a reconstruction kernel which can generate both the first and second derivative masks being used. For such a case, the first and second derivatives measures would be based on *different* reconstructions of the same data. There is no guarantee that the relationship observed in Figure 3.8 will hold if the first and second derivatives are taken from entirely different functions.

It is therefore imperative that we find a reconstruction kernel which generates $[-0.5 \;\; 0 \;\; 0.5]$ and $[1 \;\; -2 \;\; 1]$ as its first and second derivative masks, respectively. A piece-wise quintic polynomial kernel $q(x)$ was found which has all the necessary derivative characteristics[2]:

$$q(x) = \begin{cases} 1 - |x|^2 - \frac{9}{2}|x|^3 + \frac{15}{2}|x|^4 - 3|x|^5 & \text{for } |x| \leq 1 \\ -4 + 18|x| - 29|x|^2 + \frac{43}{2}|x|^3 - \frac{15}{2}|x|^4 + |x|^5 & \text{for } 1 < |x| \leq 2 \\ 0 & \text{otherwise} \end{cases} \tag{B.9}$$

The kernel, as well as its first and second derivatives, is plotted in Figure B.1. Although $q(x)$ looks very similar to the Catmull-Rom kernel $c(x)$ (also plotted) the differences become apparent upon taking derivatives, revealing that $q(x)$ is a higher degree polynomial than $c(x)$. Because $q(x)$, unlike $c(x)$, has continuous second derivatives, it can be used to generate a second derivative mask.

This quintic kernel was never implemented in the software for histogram volume generation, because it was never needed for any derivative measurements. Rather, its mere existence provides necessary justification for the approach taken in this

[2]The polynomial was found as the solution to a set of differential equations using the Mathematica symbolic math program from Wolfram Research, Champaign, Illinois.

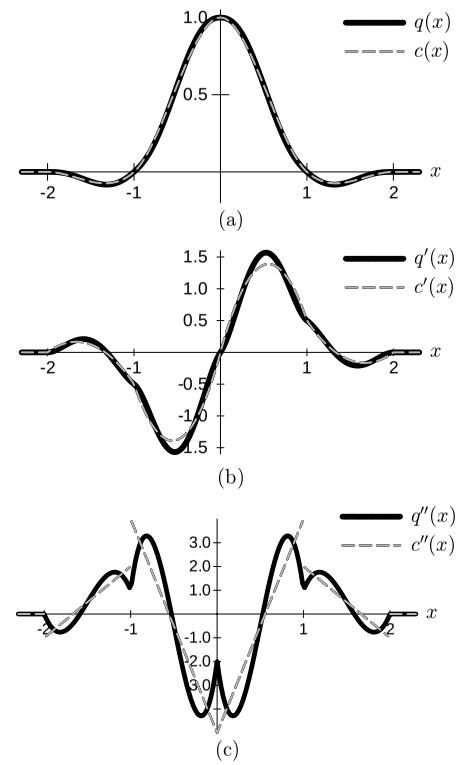


Figure B.1: The quintic kernel $q(x)$ and its derivatives. In (a) $q(x)$ is plotted along with the Catmull-Rom kernel $c(x)$ for comparison. The first derivatives of the two kernels are plotted in (b); both have continuous first derivatives, and both generate the $[-0.5 \;\; 0 \;\; 0.5]$ mask. The second derivatives are plotted in (c); the Catmull-Rom kernel has discontinuous second derivatives, unlike $q(x)$, which generates the $[1 \;\; -2 \;\; 1]$ second derivative mask.

thesis. It insures that there is some reconstruction of the sampled data which is consistent with the masks utilized for measuring derivatives. Equation B.3, and its analog for the second derivative, prove that the two masks we have chosen provide exact measures for the first and second derivatives of the data value as a continuous function of position, for the case where the data value was reconstructed with this quintic interpolation kernel.

B.3 Generalizing one dimensional masks

The two masks discussed thus far ($[-0.5 \;\; 0 \;\; 0.5]$ for first derivative and $[1 \;\; -2 \;\; 1]$ for second derivative) are only useful for measuring the derivatives of one dimensional data. These are now generalized to create masks for measuring first and second partial derivatives of sampled data in three dimensions.

Any reconstruction kernel $h(x)$ in one dimension can be trivially extended to a separable kernel $h_3(x, y, z)$ in three dimensions:

$$h_3(x, y, z) = h(x)h(y)h(z) \tag{B.10}$$

The function representing the sampled data in a volume is a three dimensional lattice of delta functions, scaled by the data values $v_{i,j,k}$, where i, j, k all range over the integers:

$$g_3(x, y, z) = \sum_{i,j,k} v_{i,j,k}\delta(x - i, y - j, z - k)$$

The reconstructed function $g_3 \star h_3$ is a lattice of kernels scaled by the data values:

$$\begin{aligned}(g_3 \star h_3)(x, y, z) &= \sum_{i,j,k} v_{i,j,k} h_3(x - i, y - j, z - k) \\ &= \sum_{i,j,k} v_{i,j,k} h(x - i)h(y - j)h(z - k)\end{aligned}$$

The partial derivatives of the reconstructed function are simple to express:

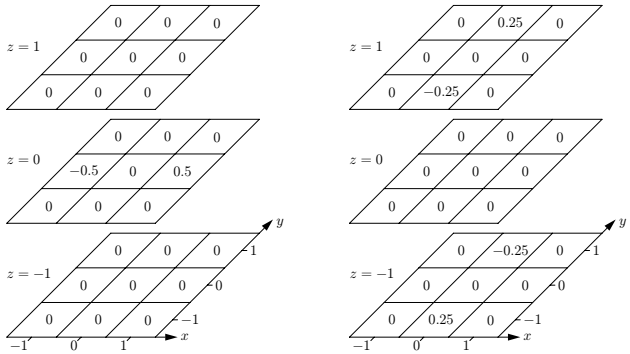
$$\begin{aligned}\frac{\partial(g_3 \star h_3)}{\partial x}(x_0, y_0, z_0) &= \sum_{i,j,k} v_{i,j,k} h'(x_0 - i)h(y_0 - j)h(z_0 - k) \\ \frac{\partial(g_3 \star h_3)}{\partial y}(x_0, y_0, z_0) &= \sum_{i,j,k} v_{i,j,k} h(x_0 - i)h'(y_0 - j)h(z_0 - k) \\ \frac{\partial(g_3 \star h_3)}{\partial z}(x_0, y_0, z_0) &= \sum_{i,j,k} v_{i,j,k} h(x_0 - i)h(y_0 - j)h'(z_0 - k)\end{aligned}$$

As in the one dimensional case, these expressions can be rewritten when the location at which the derivative is taken is a point (i_0, j_0, k_0) on the integer lattice:

$$\begin{aligned}\frac{\partial(g_3 \star h_3)}{\partial x}(i_0, j_0, k_0) &= \sum_{i,j,k} v_{i_0-i,j_0-j,k_0-k} h'(i)h(j)h(k) \\ \frac{\partial(g_3 \star h_3)}{\partial y}(i_0, j_0, k_0) &= \sum_{i,j,k} v_{i_0-i,j_0-j,k_0-k} h(i)h'(j)h(k) \\ \frac{\partial(g_3 \star h_3)}{\partial z}(i_0, j_0, k_0) &= \sum_{i,j,k} v_{i_0-i,j_0-j,k_0-k} h(i)h(j)h'(k)\end{aligned} \tag{B.11}$$

In this form, the expressions for the partial derivatives of the reconstructed function are a close analog to Equation B.4, the formulation of one dimensional derivative masks.

Whereas the one dimensional mask was a vector of evaluations of $h'(x)$, we now have a three dimensional lattice of evaluations of a partial derivative of h_3. These are the masks for derivative measurement in three dimensions. For example, the mask for calculating the first partial derivative along the x axis, $\frac{\partial}{\partial x}$, is the lattice of values $h'(i)h(j)h(k)$, where i, j, k all range over the integers. The mask for the second mixed partial derivative along the z and y axes, $\frac{\partial^2}{\partial y \partial z}$, is the lattice of values $h(i)h'(j)h'(k)$. As before, we assume the kernel h has finite support, thus there are only a finite number of non-zero values in the masks. Figure B.2 depicts the masks for $\frac{\partial}{\partial x}$ and $\frac{\partial^2}{\partial y \partial z}$. Even though these masks have an obvious three dimensional structure, the mathematics of their application is still that of a dot



(a) $\frac{\partial}{\partial x}$ mask (b) $\frac{\partial^2}{\partial y \partial z}$ mask

Figure B.2: Two examples of masks for volume derivative measurement

product: the mask is centered at the sample point of interest, and the products between corresponding data and mask values are summed to produce the final measurement. Having decided on the quintic kernel $q(x)$ for all mask generation, Equations B.11 can be used to generate $3\times3\times3$ masks for all the partial derivatives needed for any of the directional derivative measures.

Appendix C

Evaluating the performance of the derivative measures

Recall that the masks derived in Appendix B are not themselves the first and second directional derivative measures needed for histogram volume calculation. The directional derivative measures rely on the masks to provide *partial* derivative measurement, and the partial derivatives are composed into the final directional derivative measure according to the formulas in Section 4.3. We restate the formulas for the directional derivatives here, writing out the partial derivatives where necessary. There is only one measure for the first directional derivative, the gradient magnitude:

$$\mathbf{D}_{\widehat{\nabla f}} f = \|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2 + \left(\frac{\partial f}{\partial z}\right)^2} \tag{C.1}$$

For the second directional derivative, we have three measures, the first involving the gradient of the gradient magnitude:

$$\mathbf{D}^2_{\widehat{\nabla f}} f = \frac{1}{\|\nabla f\|} \nabla(\|\nabla f\|) \cdot \nabla f \tag{C.2}$$

The next measure is based on the Hessian:

$$\mathbf{D}^2_{\widehat{\nabla f}} f = \frac{1}{\|\nabla f\|^2} (\nabla f)^{\mathrm{T}} \mathbf{H} f \, \nabla f \tag{C.3}$$

$$= \frac{1}{\|\nabla f\|^2} \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \\ \frac{\partial f}{\partial z} \end{bmatrix} \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial x \partial z} & \frac{\partial^2 f}{\partial y \partial z} & \frac{\partial^2 f}{\partial z^2} \end{bmatrix} \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} & \frac{\partial f}{\partial z} \end{bmatrix} \tag{C.4}$$

The third and last measure, an approximation, is the Laplacian:

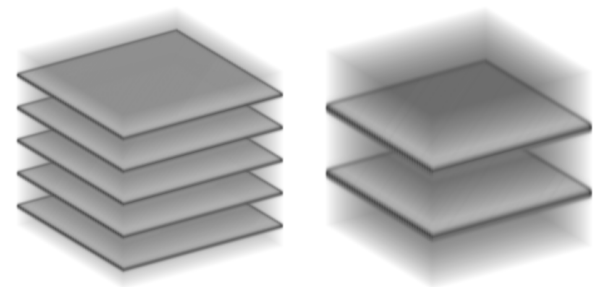
$$\mathbf{D}^2_{\widehat{\nabla f}} f \approx \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \tag{C.5}$$

It is relatively easy to perform an analytical Fourier analysis of an individual mask in order to find its frequency response, which is useful information for determining how the mask will respond to ideal boundaries of various thicknesses[BLM96]. But what is more relevant to the problem of histogram volume calculation is determining the frequency response of the directional derivative measure as a whole. Unfortunately, due to the complexity of the directional derivative formulas in Equations C.2, C.3, and C.5, the task is outside the scope of this thesis.

Instead, we evaluate the derivative measures by using them to analyze synthetic volumes containing a range of boundary thicknesses. Specifically, we create a synthetic volume for every boundary thickness between 0.33 and 8.0, at increments of 0.33. The synthetic volumes contain only planar boundaries, so we should keep in mind that the Laplacian second derivative measure will be uncharacteristically accurate, since it gives a correct directional derivative measure only for planar boundaries[1]. All the synthetic volumes were 128^3 in size. One subtle feature of the volumes was that the boundary planes were purposely not axis aligned. This

[1]To be completely thorough, we would need to create synthetic volumes containing boundaries at a wide variety of orientations and with a wide range of surface curvatures.

created more variation in the spatial relationship between the boundary and the sampling grid, which led to a better distribution of voxels along the curves in the scatterplots. Also, to maximize the number of voxels which accumulate along the curves in the scatterplots, multiple boundaries were created in each volume. By varying the spacing between boundaries in proportion to the boundary thickness, we maintain an approximately constant accumulation of hits along the scatterplot curves. To provide a sense of what these synthetic volumes look like in three



(a) (b)

Figure C.1: Renderings of two of the synthetic volumes used for derivative measure performance evaluation. The volume shown in (a) has a boundary thickness of 3.00, the one in (b) has boundary thickness 6.00. Both were rendered by giving the boundary high opacity and the material interiors a very low opacity.

dimensions, Figure C.1 gives renderings of two of them, one containing boundaries with thickness 3.00, and the other with boundaries of thickness 6.00.

Three histogram volumes are then calculated for each synthetic volume. While the gradient magnitude is always used for the first directional derivative measure, each of the three different second directional derivative measures are used once.

All histogram volumes were calculated at a resolution of 128^3. These histogram volumes are then projected to create scatterplots which we can visually compare in order to gain a qualitative understanding of how the derivative measures perform for a variety of boundary thicknesses. Recall from Section 4.3 that one important issue in histogram volume calculation is the range of derivative values that should be included on the first and second derivative axes of the histogram volume. In the analysis of these synthetic volumes, for each boundary thickness, the full range of the first and second derivative values which occur within the ideal boundary was calculated, and the range of derivative values to be included in the histogram volume was set to 120% of this. This provides the basis for qualitative evaluation of the derivative measures: by comparing the ideal versus actual sizes of the curves in the scatterplots, we get a sense of how the derivative measures respond to boundaries of a given thickness.

Figures C.3, C.4, and C.5 show the results of this analysis. Each line of the table corresponds to one boundary thickness (2σ), which is written in the first column. In the next column are slices of the datasets analyzed. After the slice, from left to right, each row contains the $f'(x)$ versus $f(x)$ scatterplot, followed by the three $f''(x)$ versus $f(x)$ scatterplots, using the different second derivative measures: gradient of gradient magnitude (from Equation C.2), Hessian (Equation C.3), and Laplacian (Equation C.5). The tops of the scatterplot columns are labeled to indicate which derivative has been measured and how. The scatterplots are turned on their sides (rotated 90 degrees clockwise from their usual presentation) to facilitate comparison of the peak amplitude in the curves across different thicknesses. Also, a thin dashed line indicates where the peak of the scatterplot curve would be in an ideal measurement. To serve as a basis for comparison, Figure C.2 shows the

two ideal scatterplots in the new orientation.

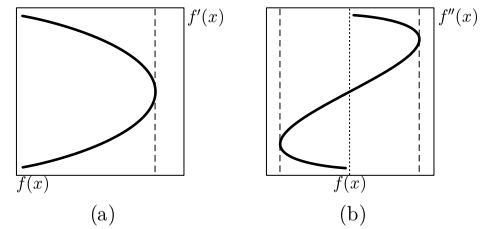


Figure C.2: Rotated ideal scatterplot curves. (a) shows the curve for first directional derivative versus data value; (b) the curve for second directional derivative versus data value. As in the scatterplots in Figures C.3 through C.5, there are dotted lines to indicate the location of the peaks of the ideal curves.

The first thing to notice about the scatterplots in Figures C.3, C.4, and C.5 is that for small thicknesses (around 1.00), none of the derivative measures' responses were very high. Besides being distorted in shape compared to the ideal, the scatterplot curves are compressed towards the $f(x)$ axis. This is an important consideration when analyzing volumes with sharp transitions between material regions. Because the histogram volume will contain a somewhat distorted measure of the boundaries, the $g(v)$, $h(v)$, and $p(v)$ calculations will also be distorted, and there will be a mismatch between the user-specified boundary function $b(x)$ and the actual opacity assigned to the boundaries in the volume by the opacity function created from our algorithm. As was done with the turbine blade dataset (Section 6.1), it may be beneficial to slightly blur the volume as a pre-process, if the resulting slight loss of detail is tolerable.

The row for thickness 2.66 at the bottom of Figure C.3 shows the typical relationship between the three second derivative measures. The measure based on

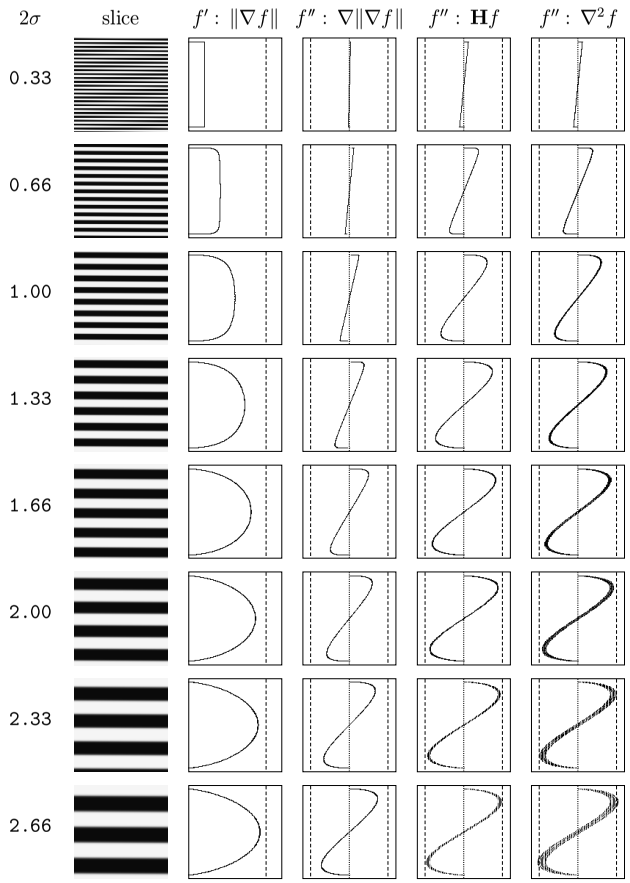


Figure C.3: Derivative measure comparisons for thicknesses 0.33 to 2.66

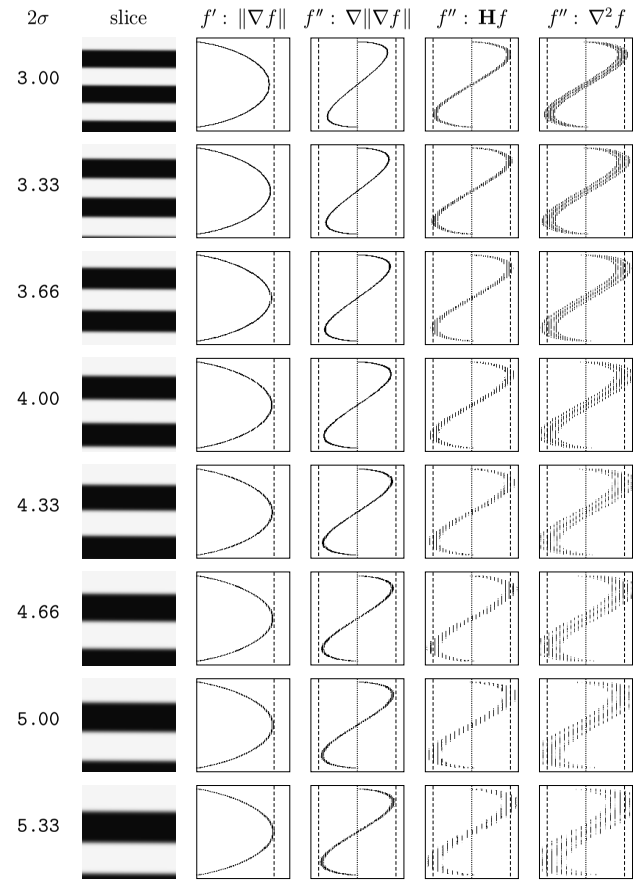


Figure C.4: Derivative measure comparisons for thicknesses 3.00 to 5.33

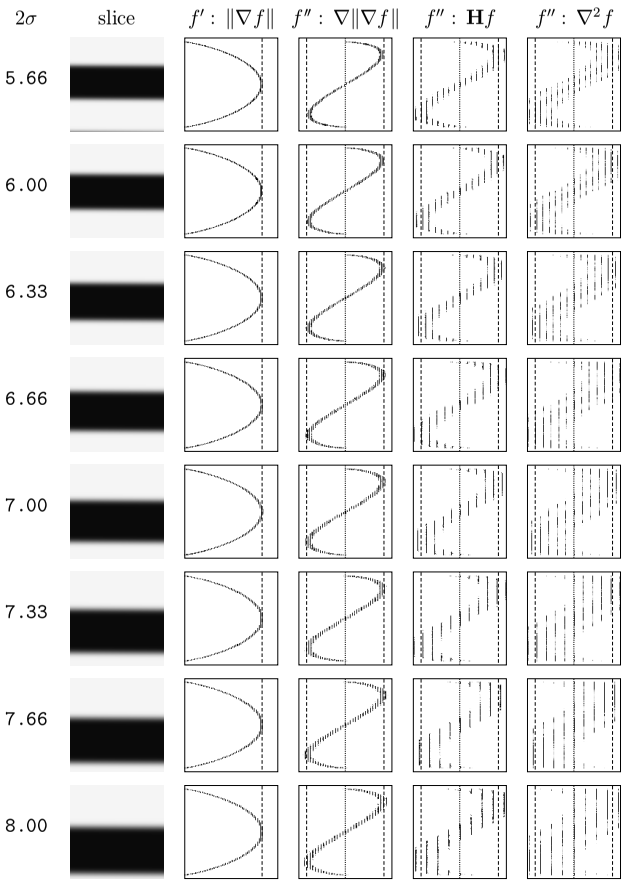


Figure C.5: Derivative measure comparisons for thicknesses 5.66 to 8.00

$\nabla\|\nabla f\|$ gives the smallest amplitude response to the boundary, while the measures based on $\mathbf{H}f$ and $\nabla^2 f$ are closer to the ideal, and are about equal. The more noticeable difference between the three measures is the amount of spread in the range of derivative values at a single data value. The curve from the $\nabla\|\nabla f\|$ measure is very coherent, but the $\mathbf{H}f$ curve is more dispersed, and the $\nabla^2 f$ curve is even more so.

The dispersion of the scatterplot curves is a result of *quantization noise*, the error introduced by representing a signal's values with fixed point instead of floating point precision. All the synthetic volumes generated for this analysis contained only 8-bit quantities. As a result, the error in the signal value (as stored in the synthetic dataset) is on average one half of 1/256, or 1/512. This amount is small relative to the changes in data value within sharp boundaries, but it is a greater percentage of the change in data value within more blurred boundaries. Thus, as the boundaries become more blurred, the quantization noise will become more and more prominent in a derivative measurement. Also, since second derivative measures are by nature more sensitive to noise than first derivative measures, the effect of quantization is always more evident in the $f''(x)$ scatterplots than in the one for $f'(x)$.

The collection of scatterplots as a whole hints at a fundamental relationship between the bit depth of the data, the boundary thickness, and the derivative measures' robustness against quantization noise, as governed by the mathematical properties of the derivative measures. A thorough exploration of this relationship would be vital to the design of higher quality derivative measures, but for the time being we choose amongst the ones already implemented. As can be seen in Figures C.4 and C.5, the second derivative measures can be ordered by their

sensitivity to quantization noise: $\nabla\|\nabla f\|$, $\mathbf{H}f$, and then $\nabla^2 f$. By the time the thickness has reached 6.00 or higher, the $\nabla^2 f$ curve has become so dispersed as to be unrecognizable, while even at thickness 8.00 the $\nabla\|\nabla f\|$ curve is still reasonably coherent. Despite this, the $\nabla\|\nabla f\|$ measure is less accurate than the other two measures, since its curve is always further from the ideal size than the others. Given the trade-offs between the three measures in terms of quantization noise, accuracy, and computational cost, we have generally adopted the Hessian-based second derivative measure as the standard.

Bibliography

[BLM96] Mark J. Bentum, Barthold B.A. Lichtenbelt, and Tom Malzbender. Frequency Analysis of Gradient Estimators in Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 2(3):242–254, 1996.

[BPS97] Chandrajit L. Bajaj, Valerio Pascucci, and Daniel R. Schikore. The Contour Spectrum. In *Proceedings Visualization '97*, pages 167–173, 1997.

[BRT95] Lawrence D. Bergman, Bernice E. Rogowitz, and Lloyd A. Treinish. A Rule-based Tool for Assisting Colormap Selection. In *Proceedings Visualization '95*, pages 118–125, 1995.

[Can86] J. F. Canny. A computational approach to edge detection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 8(6):679–698, November 1986.

[DCH88] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume Rendering. *ACM Computer Graphics (SIGGRAPH '88 Proceedings)*, pages 65–74, August 1988.

[EGLea96] M. H. Ellisman, D. Greenberg, S. Lamont, and S. Young et al. Implementing a Collaboratory for Microscopic Digital Anatomy. *Supercomputer Applications and High Performance Computing*, 10(2/3):170–181, 1996.

[Fau93] Olivier Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*, chapter 4: Edge Detection. MIT Press, Cambridge, MA, 1993.

[Fru96] Thomas Fruhauf. Raycasting Vector Fields. In *Proceedings Visualization '96*, pages 115–120, 1996.

[GK96] A. Van Gelder and K. Kim. Direct Volume Rendering with Shading via Three-Dimensional Textures. In *1996 Symposium on Volume Visualization*, pages 23–30, October 1996.

[GW93] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*, chapter 4.4. Addison-Wesley Publishing Company, Reading, MA, 1993.

[HHKP96] Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister. Generation of Transfer Functions with Stochastic Search Techniques. In *Proceedings Visualization '96*, pages 227–234, 1996.

[IYK96] T. Itoh, Y. Yamaguchi, and K. Koyamada. Volume Thinning for Automatic Isosurface Propagation. In *Proceedings Visualization '96*, pages 303–310, 1996.

[KK68] Granino A. Korn and Theresa M. Korn. *Mathematical Handbook for Scientists and Engineers*, page 820. McGraw-Hill, New York, 1968.

[Lai95] David H. Laidlaw. *Geometric Model Extraction from Magnetic Resonance Volume Data*. PhD thesis, Department of Computer Science, California Institute of Technology, Pasadena, CA, May 1995.

[LCN98] Barthold Lichtenbelt, Randy Crane, and Shaz Naqvi. *Introduction to Volume Rendering*, chapter 4. Prentice-Hall, New Jersey, 1998.

[Lev88] Marc Levoy. Display of surfaces from volume data. *IEEE Computer Graphics & Applications*, 8(5):29–37, 1988.

[LL94] Philip Lacroute and Marc Levoy. Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transform. In *ACM Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 451–458, July 1994.

[MABea97] J. Marks, B. Andalman, P.A. Beardsley, and H. Pfister et al. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. In *ACM Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 389–400, August 1997.

[Mao96] Xiaoyong Mao. Splatting of Non Rectilinear Volumes Through Stochastic Resampling. *IEEE Transactions on Visualization and Computer Graphics*, 2(2):156–170, 1996.

[Max95] Nelson Max. Optical Models for Direct Volume Rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995.

[MH80] D. Marr and E. C. Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London*, B 207:187–217, 1980.

[MMMY96] Torsten Möller, Raghu Machiraju, Klaus Mueller, and Roni Yagel. Classification and Local Error Estimation of Interpolation and Derivative Filters for Volume Rendering. In *1996 Symposium on Volume Visualization*, pages 71–77, 1996.

[Moo85] Keith L. Moore. *Clinically Oriented Anatomy*, page 882. Williams and Wilkins, Baltimore, MD, 1985.

[MT96] Jerrold E. Marsden and Anthony J. Tromba. *Vector Calculus*, chapter 2.6. W.H. Freeman and Company, New York, 1996.

[Mur95] Shigeru Muraki. Multiscale Volume Representation by a DoG Wavelet. *IEEE Trans. Visualization and Computer Graphics*, 1(2):109–116, 1995.

[Nov94] Kevin L. Novins. *Towards Accurate and Efficient Volume Rendering*. PhD thesis, Department of Computer Science, Cornell University, Ithaca, NY, January 1994.

[Ous94] John K. Ousterhout. *Tcl and the Tk Toolkit*. Addison-Wesley Publishing Company, Reading, MA, 1994.

[PBL96] James S. Painter, Hans-Peter Bunge, and Yarden Livnat. Mantle Convection Visualization on the Cray T3D. In *Proceedings Visualization '96*, pages 409–412, 1996.

[Pho75] Bui-Tong Phong. Illumination for computer-generated pictures. *Communications of the ACM*, 18(6):311–317, June 1975.

[PMea96] Vishram Pandit, Robert McDermott, and Gianluca Lazza et al. Electrical Energy Absorption in the Human Head From a Cellular Telephone. In *Proceedings Visualization '96*, pages 371–374, 1996.

[PR78] Durga P. Panda and Azriel Rosenfeld. Image Segmentation by Pixel Classification in (Grey Level, Edge Value) Space. *IEEE Trans. on Computers*, 27(9):875–879, September 1978.

[SHLJ96] Han-Wei Shen, Charles D. Hansen, Yarden Livnat, and Christopher R. Johnson. Isosurfacing in span space with utmost efficiency (ISSUE). In *Proceedings Visualization '96*, pages 287–294, 1996.

[SJM95] J.A. Schmidt, C.R. Johnson, and R.S. MacLeod. An interactive computer model for defibrillation device design. In *International Congress on Electrocardiology*, pages 160–161, 1995.

[Wes90] L. Westover. Footprint Evaluation for Volume Rendering. In *ACM Computer Graphics (SIGGRAPH '90 Proceedings)*, pages 367–376, August 1990.

[WGKJ96] W.M. III Wells, W.E.L. Grimson, R. Kikinis, and F.A. Jolesz. Adaptive segmentation of MRI data. *IEEE Transactions on Medical Imaging*, 15(4):429–442, 1996.

[YESK95] Roni Yagel, David S. Ebert, James N. Scott, and Yair Kurzion. Grouping Volume Renderers for Enhanced Visualization in Computational Fluid Dynamics. *IEEE Trans. Visualization and Computer Graphics*, 1(2):117–132, 1995.