

**Surface Wrapping: A Deformable Mesh Approach to  
Semi-Automatic 3D Volume Segmentation**

by

**James A. Carlson**

B.A., Hampshire College, 1997

A thesis submitted to the  
Faculty of the Graduate School of the  
University of Colorado in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
Department of Computer Science

2010

UMI Number: 3404043

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI 3404043

Copyright 2010 by ProQuest LLC.

All rights reserved. This edition of the work is protected against unauthorized copying under Title 17, United States Code.



ProQuest LLC  
789 East Eisenhower Parkway  
P.O. Box 1346  
Ann Arbor, MI 48106-1346

This thesis entitled:  
Surface Wrapping: A Deformable Mesh Approach to Semi-Automatic 3D Volume  
Segmentation  
written by James A. Carlson  
has been approved for the Department of Computer Science

---

Clayton Lewis

---

Geoffrey Dorn

Date \_\_\_\_\_

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

Carlson, James A. (Ph.D., Computer Science)

Surface Wrapping: A Deformable Mesh Approach to Semi-Automatic 3D Volume Segmentation

Thesis directed by Clayton Lewis

Surface Wrapping, a new approach to semi-automatic three-dimensional segmentation of volumetric data, can be used to rapidly and accurately extract object boundaries from volumes in which noise or poor imaging quality would otherwise make this process difficult. Current semi-automatic techniques often fail to perform effectively when the region of interest is both complex and poorly imaged, either requiring extensive editing of the resulting boundary or forcing the user to resort to manual segmentation. Surface Wrapping attempts to address these problems by combining a simple but robust interface for defining an arbitrarily complex approximate bounding surface with an interactive mesh deformation procedure, providing the best features of both manual and semi-automatic segmentation methods with few of the drawbacks of either.

This dissertation describes the design and implementation of Surface Wrapping, and evaluates the effectiveness of the approach using examples from both seismic and medical datasets. The technique is shown to be viable across a wide range of applications, with the potential to become a powerful addition to the existing array of 3D segmentation tools.

## **Dedication**

To my parents.

## Acknowledgements

I would first like to thank my friends and colleagues at TerraSpark Geosciences—in particular, Gwen Pech, Mick Coady, Jon Marbach, Stan Hammon, and Ben Kadlec—for their support, encouragement, and technical assistance throughout one of the most challenging periods of my life. On the subject of emotional support, I must also thank Carol Ryan and Ann Kloos, without whose aid I would certainly never have completed my doctorate.

I am also grateful to my committee for all their time spent on my behalf. Richard Han deserves special thanks for being a fantastic advisor to me for over three years, who handled my decision to completely switch thesis topics in midstream with the utmost grace. As my committee chair, Clayton Lewis's help has been invaluable to me as I have wrestled with completing this work.

I owe a tremendous debt of gratitude to Geoffrey Dorn, who not only provided the original inspiration for my research on Surface Wrapping, but who has acted as a project collaborator, advisor, boss (in the best possible sense of the term), resource for technical knowledge, and friend since I joined his research organization in late 2002.

I would also like to thank Lee Spector, my undergraduate advisor at Hampshire College. It is largely thanks to Lee's enthusiasm for the subject of computer science, and artificial intelligence in particular, that I have found what I consider to be my calling in life.

Finally, I must thank my parents for their constant love and support throughout my life. They have always encouraged me to follow my own ambitions, and have made it clear that they wish for me to find happiness, even if that comes at the expense of having a respectable career. I think I may have achieved both anyway.

## Contents

### Chapter

<b>1</b>	Introduction	1
	1.1 Overview . . . . .	2
<b>2</b>	Background	3
	2.1 Overview of Medical and Geological Volume Data . . . . .	3
	2.1.1 Medical Volumes . . . . .	4
	2.1.2 Geological Volumes . . . . .	5
	2.2 Project Origins . . . . .	9
	2.3 Related Work . . . . .	11
	2.3.1 Manual Segmentation . . . . .	11
	2.3.2 Automatic Segmentation . . . . .	12
	2.3.3 Semi-Automatic Segmentation . . . . .	14
	2.3.4 Other Related Work . . . . .	17
<b>3</b>	Surface Wrapping	19
	3.1 A Brief History of Surface Wrapping . . . . .	19
	3.2 Overview of the Surface Wrapping Technique . . . . .	21
	3.3 The Insight Earth Application Framework . . . . .	21
	3.4 Voxel Classification . . . . .	22
	3.5 Constructing the Basis Mesh . . . . .	25



3.5.1	Basis Mesh Construction for Surface Wrapping . . . . .	26
3.5.2	Bounding the Painted Slab . . . . .	29
3.5.3	Basis Mesh Construction for Surface Draping . . . . .	33
3.6	Mesh Deformation . . . . .	34
3.6.1	Surface Permeability . . . . .	39
3.6.2	Snapping to Peaks and Troughs . . . . .	42
3.6.3	Preventing Self-Intersections . . . . .	44
3.6.4	Key Algorithm Design Motivations . . . . .	47
4	Evaluation . . . . .	49
4.1	Effectiveness . . . . .	49
4.1.1	Seismic Data . . . . .	50
4.1.2	Medical Data . . . . .	61
4.1.3	Stanford Bunny . . . . .	63
4.2	Usability . . . . .	65
4.2.1	The Geoscience Interpretation Visualization Consortium . . . . .	66
4.2.2	Responding to User Feedback . . . . .	66
4.2.3	Ease of Learning . . . . .	67
4.2.4	Use In The Field . . . . .	68
5	Conclusions . . . . .	70
5.1	Contributions . . . . .	70
5.2	Future Directions . . . . .	72

<b>Bibliography</b>	75
<b>Appendix</b>	
<b>A Glossary</b>	79
<b>B Research And Development History</b>	82
B.1 Surface Wrapping . . . . .	82
B.1.1 Surface Wrapping, Version 1 (2D Prototype) . . . . .	82
B.1.2 Surface Wrapping, Version 2 (3D With Minimal GUI) . . . . .	84
B.1.3 Surface Wrapping, Version 3 (Flexible Tube Approach) . . . . .	86
B.1.4 Surface Wrapping, Version 4 (Mesh Painting, Bounded by a Fixed Range of Slices) . . . . .	89
B.1.5 Surface Wrapping, Version 5 (C++ Port, Range Slider Introduced)	91
B.2 Surface Draping . . . . .	94
B.2.1 Surface Draping, Version 1 (Early Prototype) . . . . .	94
B.2.2 Surface Draping, Version 2 (Bezier Patch-Based Surfaces) . . . . .	95
B.2.3 Surface Draping, Version 3 (Point Set Infill) . . . . .	98
<b>C User Interface Implementation</b>	99
C.1 The Insight Earth User Interface . . . . .	99
C.2 The Surface Wrapping and Surface Draping Processes . . . . .	101
C.3 Surface Wrapping Process . . . . .	101
C.3.1 Mesh Painter . . . . .	105
C.3.2 Toolbar . . . . .	110
C.3.3 Control Panel . . . . .	113
C.4 Surface Draping Process . . . . .	117
C.4.1 Setting the Target Voxel Range . . . . .	122

C.4.2	Creating a Basis Mesh for Surface Draping . . . . .	123
C.5	Common Process Parameters . . . . .	124
C.5.1	Elasticity . . . . .	124
C.5.2	Surface Permeability . . . . .	124
C.5.3	Prevent Intersections . . . . .	125
C.5.4	Deformation Constraints . . . . .	125
C.5.5	Z Deformation Scale . . . . .	126
C.5.6	Wrapping/Draping Steps . . . . .	126
C.5.7	Tighten Mesh . . . . .	127
C.6	Related Mesh Display Options . . . . .	127
C.6.1	Show Intersections . . . . .	127
C.6.2	Highlight Fixed Vertices . . . . .	127

## Figures

### Figure

2.1	Individual slices from an MR scan. . . . .	4
2.2	Slices from an MR scan stacked to form a volume. . . . .	5
2.3	Marine seismic acquisition. . . . .	6
2.4	Characteristics of an acoustic wavelet. . . . .	7
2.5	Inline slice showing a horizon, a fault, and salt dome. . . . .	8
2.6	Stratal slice showing a channel. . . . .	8
2.7	Illustration of Surface Draping. . . . .	11
3.1	Shrink wrapping is used for objects of all shapes and sizes. . . . .	20
3.2	The Mesh Painter user interface. . . . .	28
3.3	The range slider control. . . . .	30
3.4	Visual cues indicating when the current slice is within the brush range. . . . .	31
3.5	A coarse initial interpretation defined as a point set. . . . .	34
3.6	The output of the Point Set Infill process. . . . .	35
3.7	A Z-grid created from the infilled point set. . . . .	36
3.8	Mesh deformation: The projected location of a vertex. . . . .	37
3.9	Mesh deformation: The centered location of a vertex. . . . .	37
3.10	Mesh deformation: The final location of a vertex. . . . .	38
3.11	Surface permeability illustrated: deforming an elastic surface with a blunt object. . . . .	40

3.12 Surface permeability illustrated: deforming an elastic surface with a sharp object. . . . .	40
3.13 Calculating the vertex sharpness value for surface permeability. . . . .	41
3.14 Comparing results with surface permeability disabled and enabled. . .	42
3.15 A horizon interpreted with peak snapping enabled. . . . .	43
3.16 Mesh self-intersection highlights. . . . .	45
3.17 Preventing mesh self-intersections. . . . .	46
4.1 Steps in the mesh deformation process for K12CD base of salt. . . . .	52
4.2 Graph showing the percent of vertices in the draped K12CD surface that differ from the original interpretation. . . . .	53
4.3 Overly-aggressive smoothing of the K12CD basis mesh. . . . .	54
4.4 Shaded, orthographic view of the draped surface for the K12CD base of salt. . . . .	54
4.5 Comparison of manually-picked and draped surfaces for K12CD on one slice. . . . .	55
4.6 Comparison between the original EI175 amplitude volume and the processed volume. . . . .	56
4.7 EI175 salt dome: manual interpretation compared to Surface Wrapping results. . . . .	57
4.8 EI175 salt dome: wrapped surface showing 3D distance to manual picks.	58
4.9 Cracked mud. . . . .	59
4.10 Part of an inline slice from Quad 15, comparing raw amplitude and processed volumes. . . . .	60
4.11 Depth-shaded rendering of the draped Quad15 base of salt surface. . .	61
4.12 Surface Wrapping applied to an MR volume. . . . .	62
4.13 Stanford Bunny, surface-wrapped. . . . .	64

4.14 Stanford Bunny, interior and exterior surfaces. . . . .	65
B.1 Test results from the first 2D Surface Wrapping prototype. . . . .	83
B.2 Results from the first 3D implementation of Surface Wrapping. . . . .	85
B.3 The first 3D implementation of Surface Wrapping applied to the “peanut” test volume. . . . .	85
B.4 GUI for defining a tube-like basis mesh. . . . .	86
B.5 The “peanut” volume wrapped with a flexible tube. . . . .	88
B.6 Wireframe rendering of the flexible tube. . . . .	89
B.7 First implementation of the Mesh Painter tool. . . . .	90
B.8 Wrapping the “donut” test volume. . . . .	91
B.9 The Mesh Painter implemented within Insight Earth. . . . .	93
B.10 Results from the first Insight Earth-based implementation of Surface Wrapping. . . . .	94
B.11 Defining a simple contoured 3D surface using spline patches. . . . .	96
B.12 Highlighting the intersection of the spline surface and the current slice.	97
C.1 The main window of Insight Earth. . . . .	100
C.2 The process node for Surface Wrapping. . . . .	101
C.3 The Mesh Painter window. . . . .	106
C.4 The Target Voxel Range histogram palette. . . . .	106
C.5 The range slider control. . . . .	107
C.6 Showing different display and target volumes in the Mesh Painter. . . .	108
C.7 The histogram palette for the display volume. . . . .	109
C.8 Onion skinning. . . . .	110
C.9 The toolbar of the Mesh Painter window. . . . .	110
C.10 The brush cursor. . . . .	111
C.11 The eraser cursor. . . . .	112

C.12 Using the Flood Fill tool to fill a closed painted region. . . . .	112
C.13 The Control Panel for the Mesh Painter. . . . .	114
C.14 The Target Voxel Range histogram. . . . .	114
C.15 The Change Mesh Resolution dialog. . . . .	115
C.16 The Display Volume Histogram window. . . . .	116
C.17 The scale editor for the Mesh Painter. . . . .	116
C.18 Hiding the Mesh Painter's control panel. . . . .	118
C.19 The process node for Surface Draping. . . . .	119
C.20 Setting the target voxel range for Surface Draping. . . . .	123
C.21 Comparison of elasticity values. . . . .	124
C.22 A self-intersecting mesh. . . . .	126
C.23 The Highlight Fixed Vertices mesh display option. . . . .	128

## **Chapter 1**

### **Introduction**

Three-dimensional (3D) segmentation of volumetric image data is critical to many fields, particularly the medical and geological sciences. Computed tomography (CT) and magnetic resonance (MR) scans are used in the diagnosis and treatment of many types of medical conditions, as well as in education and research. The analysis of geophysical volumes often requires 3D interpretation of structural features such as faults, and stratigraphic features such as channels and salt bodies.

Rapid advances in data storage capacity, processor speed, and the inclusion of dedicated 3D graphics processing units in commodity hardware make working with volumetric data increasingly accessible [24], yet the segmentation process itself remains both time-consuming and error-prone. Many automatic and semi-automatic segmentation techniques are currently in use, but when working with volumes in which the boundaries of interest are poorly imaged or unclear because of excessive noise, or where human judgement is needed to correctly pick a boundary, these approaches often fail to produce complete surfaces, or require such extensive editing to fix inaccuracies that much of the time saved relative to manual segmentation is lost [3].

Surface Wrapping is a novel semi-automatic volume segmentation technique that finds a middle ground between fully manual segmentation and current semi-automatic methods, allowing a high level of control in regions where human judgement is necessary while automatically picking boundaries in areas where there is less



ambiguity in the data. A unique graphical user interface has been developed that allows fast and easy construction of a highly complex approximate 3D bounding surface. This surface is converted to a closed, triangulated mesh, which is then deformed, via a user-guided process, to fit the region of interest. Surface Wrapping is computationally inexpensive and scales well to large volumes, and is capable of producing good results when applied to volumetric images that would be difficult or impractical to segment using other approaches.

## **1.1 Overview**

Chapter 2 provides background information about current approaches to segmentation, discusses technologies similar to Surface Wrapping that are in use in other problem domains, and briefly explains the conceptual origins of this research. Chapter 3 gives an overview of Surface Wrapping and describes the technical details of the current implementation. Chapter 4 evaluates the success of the technique from the standpoint of both its effectiveness as a segmentation tool and the usability of the software, and Chapter 5 summarizes the contributions of this work and suggests opportunities for future research.

## **Chapter 2**

### **Background**

This chapter begins with an overview of the characteristics of volumetric datasets used in the medical and geological sciences. The origins of the Surface Wrapping approach are briefly examined, followed by a review of other manual, automatic, and semi-automatic 3D segmentation techniques. Finally, a selection of work is described that is not directly related to segmentation, but in which physical shrink wrapping is used as a model for surface deformation.

#### **2.1 Overview of Medical and Geophysical Volume Data**

This dissertation addresses 3D segmentation problems relating to both medical and geological sciences; however, greater emphasis has been placed on the application of Surface Wrapping to datasets from the latter domain. There are three significant reasons for this preference. First, seismic (geophysical) volumes are more prone to noise and poor imaging quality than medical volumes as a result of the means by which the data are acquired, and therefore present a greater challenge for segmentation techniques that involve automated assistance. Second, studies involving seismic data are relatively uncommon in the related literature in the field of computer science, with most such research focusing instead upon applications in medicine. Third, while medical volumes have been used in the development of Surface Wrapping from the inception of the technique, the majority of recent research has focused on its use for

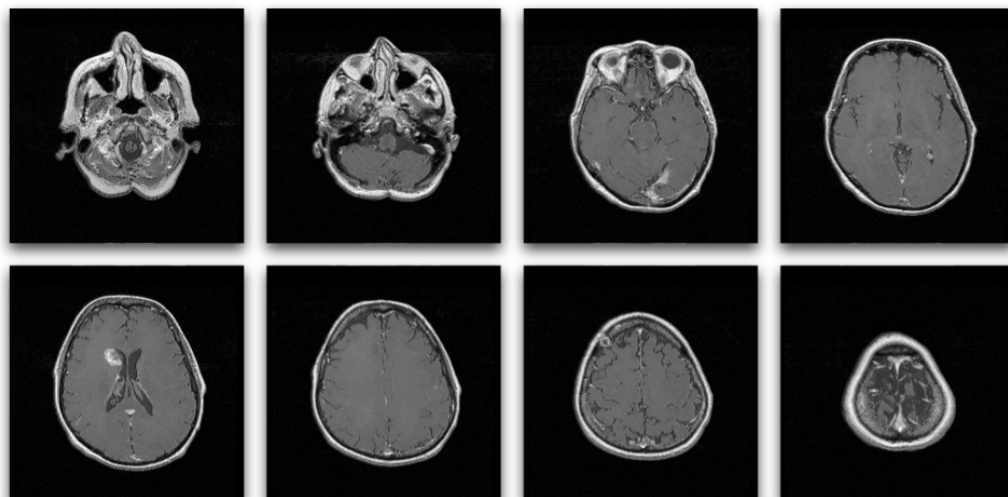


Figure 2.1: Individual slices from an MR scan.

seismic interpretation.

This section gives an overview of the methods used to acquire medical and seismic volumes, and discusses some of the ways in which segmentation is typically used.

### 2.1.1 Medical Volumes

Segmentation of medical volumes is necessary for many applications, including visualization for diagnosis and education, and analysis for dosimetry and other forms of treatment planning. Volumetric images of live subjects are primarily acquired using magnetic resonance (MR), computed tomography (CT), or positron emission tomography (PET) scanners, which produce a series of 2D slices (Figure 2.1) from which a 3D volume can be assembled (Figure 2.2). Different types of tissues in the body produce different voxel values within the volume, based on opacity to X-rays (in CT scans) or response to magnetic fields (in MR scans), resulting in 2D slices that bear a visual resemblance to traditional X-ray photographs.

Efforts such as the Visible Human Project [52] have also produced highly de-

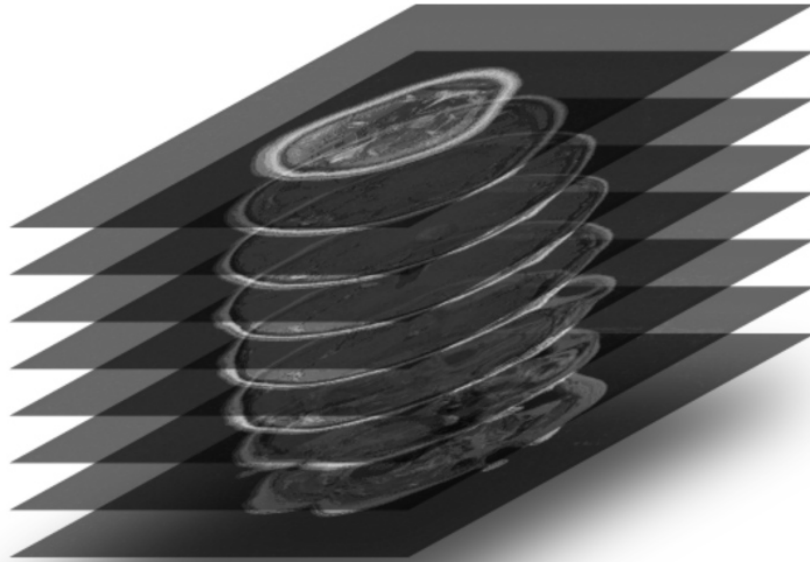
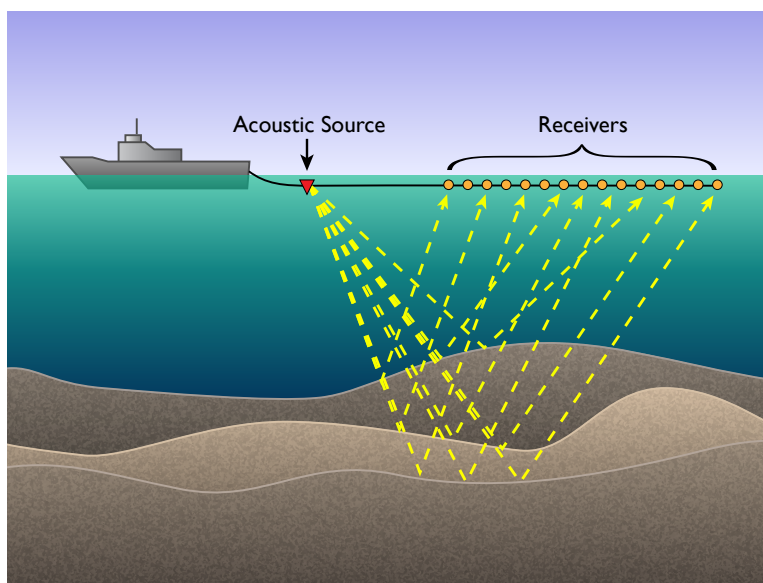


Figure 2.2: Slices from an MR scan stacked to form a volume.

tailed volumetric datasets for the purposes of education and research, using cadavers instead of live subjects. Slices of these volumes are created using direct photography instead of non-invasive scans: the subjects are frozen to stabilize soft tissues, then planed down at fine (1 mm or less) intervals, allowing the resulting surface to be photographed. Unlike MR or CT volumes, the data generated by this process is full-color, but lacks information about tissue density, which can require different approaches for segmentation [35].

### 2.1.2 Geophysical Volumes

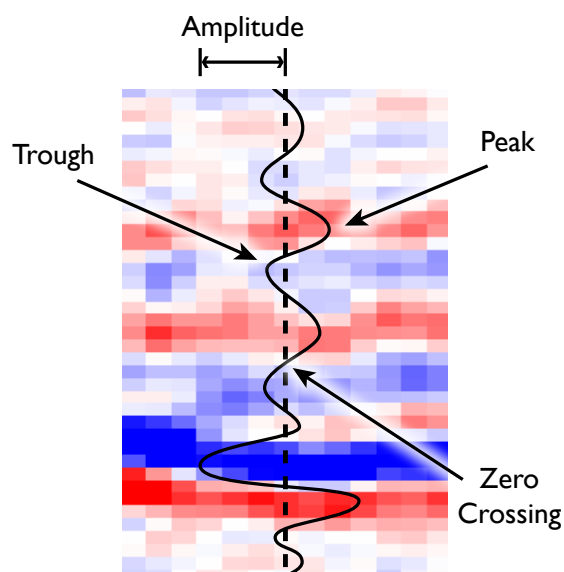
The geophysical volumes of interest to this study are created via the process of seismic acquisition. Seismic data are acquired by propagating an acoustic wave through the ground, which is partially reflected back to the surface at the interfaces be-



**Figure 2.3:** Marine seismic acquisition. A ship drags acoustic sources followed by long streamers of receivers that record the signal as it is reflected by the strata in the ground.

tween different layers of rock or sediment. An array of receivers on the surface records the reflected acoustic wave, generated from sources such as dynamite blasts, shots of compressed air, or vibrator machines.. The recorded reflections are then moved to their true location in X-Y-Time space—a process known as migration—to create a 3D seismic volume. (The Time component of the X-Y-Time coordinate space refers to the two-way travel time of the reflected acoustic signal, which does not map directly to depth.) Seismic data may be acquired on land or at sea (Figure 2.3).

Figure 2.4 shows an illustration of an acoustic wavelet as represented in a time-migrated seismic volume. A trace is indicated by the dotted line (with time increasing in the downward direction), with the corresponding waveform drawn as a curved solid line. The amplitude of the reflected signal is indicated by the position of the waveform on the left or the right of the centerline, with negative amplitude increasing to the left and positive amplitude increasing to the right. Peak and trough amplitude refer to the maximum and minimum amplitude values, respectively, at a point on the wavelet



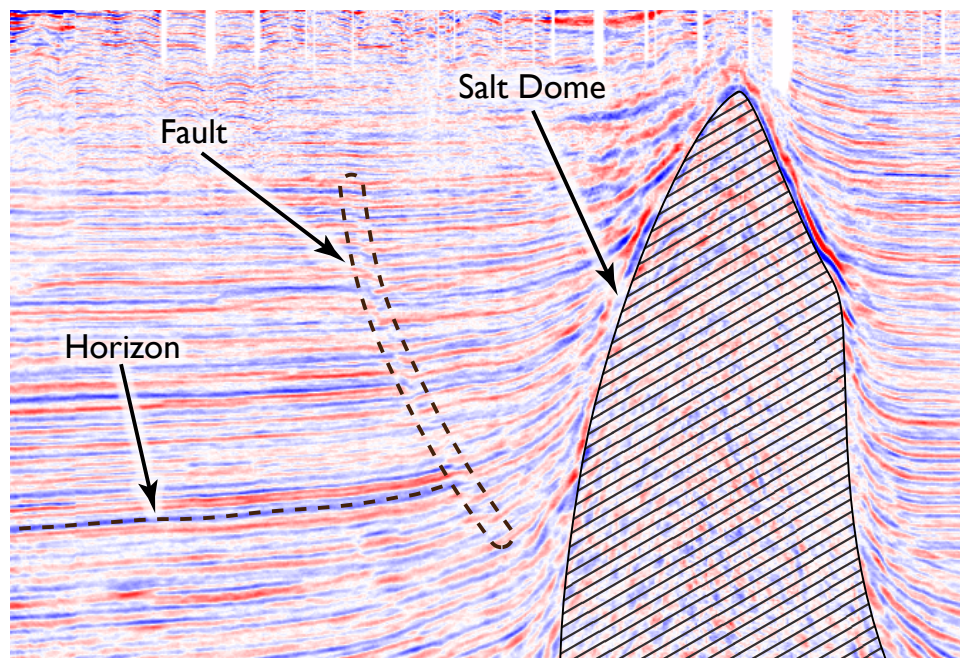
**Figure 2.4:** Characteristics of an acoustic wavelet, overlaid on an image from an actual amplitude volume.

where the values at the times before and after are nearer the centerline. In an 8-bit amplitude volume, voxel values typically range from -127 to 127.

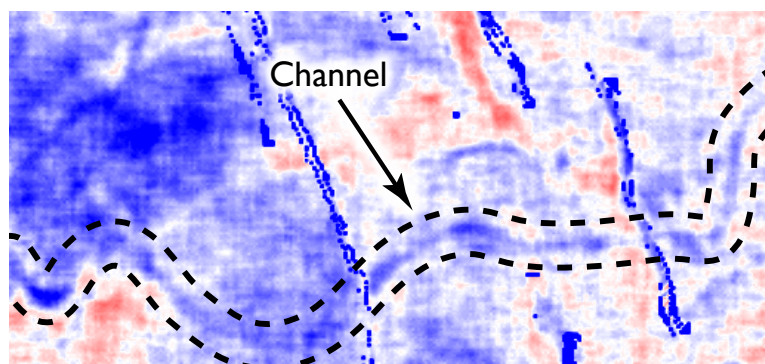
Some common geological features that will be discussed in this dissertation are illustrated in Figures 2.5 and 2.6. Figure 2.5 shows a horizon, which is an interface between two layers of rock which causes a reflection of the acoustic signal; a fault, which is a fracture of rock causing displacement along a planar surface; and a salt dome, which is a formation of salt that has pushed upward through the rock. Figure 2.6 shows a slice from a stratal volume (Section 3.4) that reveals a channel, which is a concavity formed by the flow of water (i.e., a stream bed).

While there are hard problems relating to the segmentation of both medical and geophysical volumes, some challenges that are unique to seismic interpretation should be mentioned in particular.

The first difficulty is that the relatively long wavelength of an acoustic signal used in seismic acquisition greatly limits the vertical resolution of the volume: for



**Figure 2.5:** Inline slice from an amplitude volume with a horizon, fault, and salt dome highlighted.



**Figure 2.6:** A slice from a stratal volume with a channel highlighted.

example, a 33 Hz signal can have a wavelength of 75 meters, with a limitation on the detection of separate, adjacent surfaces to one-quarter the wavelength [5, 14]. By contrast, biological features are captured in a CT scan using X-rays, which, having a wavelength of 10 to 0.010 nanometers, allow correspondingly finer resolution.

Further complicating seismic interpretation is the necessity of reconstructing a spatial volume from reflected signals that are generated from, and detected at, approximately the same surface plane. Migration is a complex undertaking that is subject to both errors in human judgement and the limitations on acoustic imaging already discussed. When seismic data must be acquired on land, it is often impossible to achieve the kind of regular, grid-like spacing of emitters and receivers that is used for acquisition on sea, and these irregularities make migration even more problematic. Although there are some circumstances in which transmissive methods can be used to acquire geological data—borehole to borehole, for example—such techniques are uncommon as compared to reflection seismology. While this is not meant to imply that constructing medical volumes is simple, transmissive imaging is at least comparatively reliable when a subject can be surrounded by emitters and receivers from 360 degrees.

## 2.2 Project Origins

Surface Wrapping is an extension of Surface Draping, a semi-automatic horizon interpretation technique first described by Geoffrey Dorn in 1999 [12]. Surface Draping was conceived as a means to simplify the picking of horizons which are too complex to interpret efficiently using conventional manual or automated techniques (e.g. autotracking).

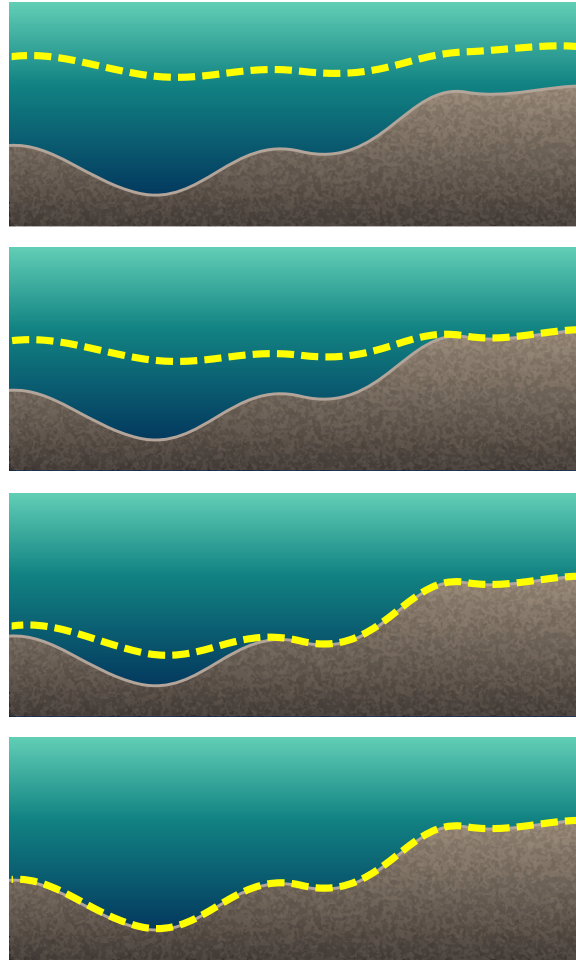
Like many deformable surface methods, Surface Draping relies on the user to specify an approximate initial boundary which is then automatically conformed to the surface imaged in the data. The initial boundary is defined by the user on successive vertical, axis-aligned slices of the volume, but rather than picking directly on the hori-



zon of interest, the user picks the boundary slightly above (or below) the event. A 2.5D surface is created by repeating this procedure for each slice in the volume and merging the collection of manually-defined 2D lines; or, to save time, the user can define the lines on alternate slices only (or every 5th slice, 10th slice, etc.), in which case the complete initial surface is created by infilling the gaps between slices using linear interpolation.

After the initial surface has been fully constructed, the deformation algorithm shifts each point in the surface vertically by searching downward along the trace from the point's initial Z position until a peak (or trough) is encountered, then snapping the point to the center of that event. In this manner, a complete, laterally-extensive surface can be generated that automatically captures the horizon of interest in one pass without the need for subsequent manual editing. Metaphorically, this approach is similar to draping an elastic sheet over a physical surface and allowing gravity to pull the sheet downward (hence the name Surface Draping).

As originally envisioned, Surface Draping is limited in that it can only be used to segment boundaries that can be represented as Z-grids. Surfaces that overlap in Z cannot be represented, which eliminates the possibility of using the technique to segment objects with closed boundaries: e.g. channels, canyons, and salt bodies in seismic surveys, or virtually any objects of interest in CT or MR scans. Surface Wrapping is intended to address these and other shortcomings, generalizing the concept of Surface Draping so that it can be applied in the segmentation of arbitrary 3D bodies in any domain.



**Figure 2.7:** Surface Draping: A simulated elastic sheet, shown here as a dotted line, is placed above the surface of interest (top image), then iteratively lowered until it fits the surface.

## 2.3 Related Work

### 2.3.1 Manual Segmentation

Manual segmentation refers to any procedure in which the user identifies the boundaries of the region of interest by hand, without automated assistance. Manual segmentation is often, if not always, performed on a slice-by-slice basis, and is therefore a very time-consuming process that requires a high level of attention to detail. The two most common approaches to manual segmentation are tracing an outline (using ei-

ther raster drawing tools or parametric curves) [39] or painting a filled region [30, 41].

Despite the amount of time required for manual segmentation, it is still considered the only practical approach in many situations, particularly when working with volumes in which the boundaries of interest are poorly-imaged or obscured by noise. This can force a compromise between the need for accurate results and the amount of time that can be devoted to the task. In seismic interpretation, where a salt body may extend across hundreds (or even thousands) of inline and crossline slices, it is common practice to interpret the surface on every tenth slice.

Whether or not a segmentation technique is regarded as fully manual is somewhat open to debate. Horizon picking tools in most seismic interpretation software, for example, offer the ability to snap a picked point to the nearest peak or trough on a trace, yet such tools are typically considered manual rather than semi-automatic because the initial pick must be made very close to the peak or trough of interest for the snapping to work as intended. Similarly, interpolating a surface across picked lines (when slices have been skipped during manual interpretation) can be considered a semi-automatic technique [11], even though the results are not guided by the data.

### 2.3.2 Automatic Segmentation

Automatic methods use global parameters to create a segmentation of image data without requiring user-supplied positional input.

The simplest form of automatic segmentation is binary thresholding (also known as binarization): classifying all voxels with a value above (or below) a user-defined threshold as belonging to the region of interest [53]. Multi-thresholding is a variation in which one or more ranges of foreground values are defined, which is often used to exclude noise (or other undesired characteristics of the image) with values above or below the range that defines the target [46]. While it is usually difficult to obtain clean results that exclude all unwanted features using thresholding alone, the technique is

still frequently used for volume visualization when a precise boundary is not required, and the computational efficiency of thresholding allows uses in applications where real-time interaction is important.

An isosurface (also known as a level surface or implicit surface) can be constructed to provide a lightweight representation of the boundaries of the foreground voxels in a binarized volume, an approach that can be coupled with thresholding for use in real-time 3D visualization. The marching cubes algorithm [33] is commonly used to generate an isosurface as a triangulated mesh, but there are other methods, such as marching tetrahedrons [37], that achieve similar results. Using on-the-fly mesh simplification to reduce total polygon count, extracting isosurfaces can be an efficient means of rendering very large volumetric features [13].

Considerable research has been devoted to automatic segmentation of both 2D and 3D image data using artificial intelligence-based approaches [50]. Brandel et al. [4] demonstrated a knowledge-based technique that generates an automatically-interpreted model of a seismic volume using a “geological pilot,” a computational supervisor that guides the model-building process by ensuring that automatically interpreted surfaces are geologically correct. Knowledge-based approaches have also been used in the segmentation of medical volumes, often attempting to isolate specific features (such as the lungs [1] or vasculature [28]) by identifying regions that resemble the known characteristics of the features of interest. A more generalized automated technique described by Olowoyeye et al. [40] uses Gabor filtration to classify regions of a volume based on their textural properties. The works cited above do not represent an exhaustive survey of automatic segmentation algorithms, but are intended to illustrate the scope of this area of research.

While fully automated segmentation algorithms are useful in situations where speed is the main requirement, it is widely acknowledged that such approaches are usually insufficient when the more important factors are the accuracy and complete-

ness of results [20, 40].

### 2.3.3 Semi-Automatic Segmentation

Semi-automatic methods attempt to provide a middle ground between fully manual and fully automatic segmentation, ideally picking most of the significant boundaries automatically with little manual input, while allowing the user to provide guidance when ambiguity in the data prevents the software from correctly tracking the target region. In some domains, semi-automatic segmentation has been shown to produce accurate results much more rapidly than manual segmentation [21].

#### 2.3.3.1 Region-Growing (and Autotracking)

Seeded region-growing algorithms segment an image by accepting one or more user-provided seed points as input, then searching outward for neighboring pixels or voxels that are part of the same object based on some measure of connectedness or similarity [53]. The simplest such predicate accepts voxels with values that fall within a set range of the value at the seed point; essentially, the region-growing equivalent of histogram thresholding. Fan et al. [15] survey many techniques that employ different criteria for pixel/voxel connectivity or similarity. Another variation, described by Kadlec [26], provides an additional level of control by allowing the user to interactively modify the parameters that constrain the growth of the region, making it easier to achieve the correct fit to the feature boundaries. Region-growing algorithms are commonly employed in medical data analysis, where they are useful in the segmentation of bone, cartilage, blood vessels, or other anatomical structures with consistent tissue density [23].

Volume-filling algorithms are less frequently used for seismic interpretation, but region-growing techniques are commonly employed in automated horizon tracking tools, also known as autotracker. Like other seeded region-growing methods, an auto-

tracker is used by manually picking a seed point on or near a horizon, from which a 2D curve or 3D surface is propagated across connected voxels [5]. It should be noted that the term “autotracker” can refer to any semi-automatic seeded approach for generating horizon surfaces (e.g. the ant-tracking algorithm described by Pedersen et al. [42]), not strictly those based on region-growing algorithms.

There are two major drawbacks to region-growing techniques that can hinder their usefulness: the inability to close gaps in the data when a surface is incompletely imaged, and the risk of expanding to fill undesired portions of the volume due to the presence of similar voxels that do not belong to the region of interest. The former problem can sometimes be addressed with tools that allow the user to identify and automatically fill gaps in the output surface, although most (if not all) such tools are only applicable for infilling holes in regular grids, not for connecting arbitrary edges of separate closed meshes. Region-growing algorithms can, to some extent, be prevented from growing past the bounds of the target by restricting the distance to which a surface will expand past the seed point, but this approach can require significant trial and error on the part of the user to determine the optimal range limit with respect to the location of the seed point.

Even when range limits are employed, extensive manual editing of the output of region-growing tools is often necessary. In discussions with seismic interpreters who use autotrackers, a common complaint is that although autotrackers can accurately pick 90% of the surface area, 90% of the time required to interpret a complete, accurate surface is spent manually cleaning up and picking the remaining 10%. Perhaps a better indication of the practical limitations of such tools is that, as discussed in Section 2.3.1 it is often considered preferable to manually pick a large salt body on every tenth slice—essentially discarding 90% of the relevant data in the volume—than to use an autotracker.

### 2.3.3.2 Active Contours (a.k.a. Snakes)

The use of active contour models to segment 2D images was first described in 1988 by Kass et al. [27]. The active contour approach is used by first manually defining a spline-based approximate boundary near the region of interest. An energy minimizing function is used to attract the boundary to nearby edges in the image, and simulated elastic and rigid forces are employed to cause a preference for smooth boundaries. The technique is also known as “snakes” because the contour appears to wriggle snake-like as it searches for a configuration that minimizes its global energy. Active contour models have also been used to rapidly segment volume data by applying the technique on each consecutive slice [34, 44] (or a subset of the slices [32]) containing part of the region of interest, then combining the results from all slices. A variant of active contours that uses a 3D initial bounding surface was described by Chan et al. in 2002 [7].

An important enhancement to this approach was introduced by Cohen in 1991 [9] as “balloons,” which resolves some of the problems observed in the original technique. An inflation force is employed to help push the curve past isolated edge points and onto stronger edges, and variations on the image forces and edge detection algorithm were employed to yield better stability. A three-dimensional version of balloons is described in 1992 by Cohen and Cohen [8] for segmentation of medical volumes.

One drawback to snakes is that after the initial boundary is defined, the energy minimization function is applied globally to the entire curve (or surface), which can lead to unexpected and undesirable results that require the user to either revise the initial boundary or manually edit the final contour. Another frequently-cited problem is the difficulty of such techniques to fill large concavities, as the energy-minimizing function tends to cause the contour to snap straight across gaps.

### 2.3.3.3 Live-Wire

The Live-Wire technique introduced by Mortensen and Barrett in 1995 [36] addresses the issues associated with snakes by snapping the contour to the strong edges in the image at interactive speeds while the user is drawing the coarse boundary, allowing immediate correction of poor results. An “edge cooling” feature prevents local changes in the energy of the contour from affecting earlier portions of the boundary that are considered optimal, and a local training operation alters the parameters of the edge-detection algorithm interactively as the gradients in the image change. Farin [16] improved upon this approach by allowing the user to paint a “corridor” surrounding the boundary of interest that restricts the area covered by the energy minimization function, which can prevent the boundary from snapping to undesired edges and also improves the performance of the algorithm. Live-Wire is one of the most commonly-used techniques for medical volume segmentation, typically using an approach similar to that described by Loncaric et al. [32] by combining the results of contours traced on a subset of the slices [47].

### 2.3.4 Other Related Work

Surface Wrapping is not the first deformable surface technology for which physical shrink wrapping has served as a model, though it does appear to be the first instance in which the concept has been directly applied to volume segmentation.

Kobbelt et al. [29] describe a remeshing technique that uses shrink wrapping as a physical model to generate a mesh with subdivision connectivity based on a mesh with arbitrary connectivity. The method used to deform the generated mesh is similar to that described in this dissertation, iteratively projecting the vertices along their normals, then relaxing the mesh to maintain regular spatial vertex distribution and prevent folding.



A shrink wrapping approach is proposed by Jeong and Kim [25] for constructing a mesh based on an arbitrarily-structured point cloud, also using a mesh deformation algorithm similar to that of Kobbelt et al. [29]. An improvement to this technique was presented by Koo et al. [31], in which a better fit is achieved by creating an approximate bounding mesh defined as a collection of cubic cells in a 3D grid, with cells that form the boundary specified by the presence of points from the input data falling within the corresponding cell in the grid. This method of creating a coarse initial mesh of arbitrary topological complexity from a set of axis-aligned cells was one of the inspirations for the manual “mesh painting” tool used in Surface Wrapping (Section C.3.1). A similar concept has also been employed by Celniker et al. [6] for remeshing isosurface representations of geophysical data.

Tools also exist for 3D model creation that use shrink wrapping approaches to approximately fit one mesh to the surface of another mesh under user control. Poser Tool Box [10], a commercial add-on to the Poser 3D modeling product [49], includes a tool for creating form-fitting labels or articles of clothing by using a cylindrical mesh as a starting point, which the user is able to progressively contract toward a centerline until some portion of the cylinder connects with the target mesh. The *shrinkwrap* modifier in Blender [43] is a skinning/retopology tool that fits an arbitrary, user-defined mesh to the surface of an underlying target mesh by moving the vertices of the bounding mesh along their normal vectors until they reach the target.

## Chapter 3

### Surface Wrapping

Surface Wrapping is a novel semi-automatic approach to volume segmentation that is intended to be most effective when applied to datasets in which noise, acquisition artifacts, poor imaging, or other sources of ambiguity prevent the effective use of alternative semi-automatic techniques. Surface Wrapping is essentially based on the concept of physical shrink wrapping (Figure 3.1), in which a loose-fitting plastic material surrounding an object is shrunk, using either heat or suction, until it conforms to the enclosed item. A rather literal interpretation of this physical model has been developed (as compared to other segmentation techniques which reference this sort of metaphor, e.g., balloons [8]) which couples a graphical user interface for creating a complex 3D initial bounding surface with an interactive deformation scheme that enables manual improvement of the fit as the surface is conformed to the region of interest.

#### 3.1 A Brief History of Surface Wrapping

Unlike most semi-automatic volume segmentation techniques currently in use [20], Surface Wrapping has been developed as a 3D solution from the outset, a distinction which has had a significant impact on both the user interface and the underlying algorithms. As discussed in Section 2.2, the original goal of the project was simply to determine whether or not a Surface Draping-like approach could be effective at all



**Figure 3.1:** Plastic shrink wrap used as a protective covering for swiss cheese, little glass vials, electrical connections, and heavy industrial equipment.

for the segmentation of bounded 3D objects, beginning with simple, roughly spherical bodies such as brain tumors. When early experiments showed promise, the project's goals expanded as the capabilities of the system became increasingly apparent, a trend which has brought about major—and frequent—changes in most aspects of the software.

Following early trials with spheroid objects (for which no user interface was provided apart from a simple 3D viewer window, no manual control of the surface deformation process was allowed, and a spherical geodesic mesh served as the initial bounding surface), the tool was redesigned based on a new goal of segmenting sinuous, channel-like features. In response to feedback criticizing the inability of software to handle bifurcations, another complete redesign was conducted, after which it was possible to segment bodies having limited vertical extent but arbitrarily complex horizontal boundaries. A further improvement allowed Surface Wrapping to be used to segment features that extend across any number of slices in Z, but the vertical borders of which should roughly orthogonal to the horizontal plane (e.g., canyons and

salt diapirs) for most efficient use of the tool.

A more thorough overview of the project's history is provided in Appendix B.

### 3.2 Overview of the Surface Wrapping Technique

At a high level, Surface Wrapping is comprised of the following three steps:

- (1) *Voxel classification*, in which the user isolates—as much as is practical—the voxels belonging to the borders of the region of interest via data filtering and thresholding.
- (2) Construction of a *basis mesh*: the initial surface that approximately bounds the region of interest.
- (3) An interactive *mesh deformation* process in which the basis mesh is conformed to the region of interest.

The remainder of this chapter discusses each of these steps in detail, preceded by a brief description of the application framework within which the software has been developed.

### 3.3 The Insight Earth Application Framework

Surface Wrapping has been implemented as a process module within the Insight Earth application framework. Insight Earth provides facilities necessary for volume interpretation, such as data import and export, an interactive 3D viewer, and a range of processes that can be used to prepare volumes for the initial step of voxel classification. The application (and thus Surface Wrapping itself) is written in C++, using the wxWidgets toolkit [48] for the user interface infrastructure, and OpenGL for 3D (and some 2D) display functionality. A more detailed explanation of the Insight Earth user interface is provided in Section C.1.

### 3.4 Voxel Classification

“Voxel classification,” as most commonly used in the literature relating to 3D segmentation, is defined as the process of determining which voxels belong to the regions of interest in a volume [54]. This term is often used in the context of automatic voxel classification algorithms, such as the approach described by Olowoye et al. [40], but it can refer to manual or semi-automatic methods as well. As the first step of the Surface Wrapping method, voxel classification simply refers to the binary (or ternary, as explained in Section 3.6.2) separation of the image data into foreground (or “target”) and background voxels.

There are two objectives of this step. The first is to isolate, as much as possible, the borders of the region of interest from the surrounding data; revisiting the metaphor of physical shrink wrapping, the target voxels represent the solid object onto which the elastic surface is contracted. The second goal is to make it easier for the user to create a basis mesh.

Double-thresholding is used to binarize the volume into target and non-target voxels, as

$$M = \begin{cases} \text{true,} & \text{if } t_1 \leq v \leq t_2 \\ \text{false,} & \text{else} \end{cases} \quad (3.1)$$

where  $v$  is a voxel value and  $t_1$  and  $t_2$  are threshold values specified by the user (via the mechanism described in Section C.3.1.1), with  $M = \text{true}$  classifying a target voxel.

While conceptual simplicity and low computational cost were certainly factors in the decision to use thresholding as the basis for determining the borders of the region of interest, these were not the only motivations. (Even the original Surface Draping implementation used a more sophisticated approach of searching along traces for

peaks or troughs, then determining the ultimate position of each point in the surface with sub-voxel precision by fitting a curve to those voxels.)

The main reason for using thresholding is that this arguably makes Surface Wrapping as generalized as possible by limiting the assumptions that are made about the characteristics of the data. By exclusively targeting peaks and troughs, it would not be possible to use Surface Draping to segment features in medical volumes, for example. Similarly, algorithms that make assumptions about the relative proximity of strong edges may be well-suited to finding the bounds of anatomical features, but would not be viable for finding the bounds of geobodies because of the alternating pattern of peaks and troughs that are usually the strongest edges found in a seismic volume. Keeping such assumptions to a minimum allows Surface Wrapping to be equally applicable to many different domains.

For some types of data, thresholding may be all that is necessary to use Surface Wrapping to segment the region of interest—for instance, when segmenting bones in a CT scan. For other datasets, a simple form of preprocessing, such as the application of an edge-detection filter, will be necessary. In some situations, however, this process can become much more involved.

As an extreme example, consider the domain transformation workflow described by Hammon [19]. This process was developed, in part, to make channels easier to identify and interpret, especially in steeply-dipping intervals in which channels are broken up by extensive faulting. Domain transformation is particularly useful when used in conjunction with Surface Wrapping because flattening the stratigraphy in a volume can allow the bounds of a channel to be interpreted in a single pass using the Mesh Painter to define the basis mesh, but achieving that result requires multiple steps. First, unconformities such as reefs, canyons, and salt bodies must be segmented, a task which might itself be undertaken using Surface Wrapping. Next, faults are interpreted, a process which may include the use of multiple attributes to enhance the imaging of

the faults, plus the use of manual, automatic, and semi-automatic interpretation techniques to determine the actual fault surfaces. Major horizons within and surrounding the interval in which the channel is found are then interpreted, another process which may involve the use of attributes to improve the effectiveness of semi-automatic interpretation tools, such as autotrackers or Surface Draping. All of these interpreted surfaces are then used by the domain transformation process to produce a volume in which the depositional layers in the interval are flattened, which makes most or all parts of the channel visible on a single stratal (Z-axis orthogonal) slice. Finally, additional processing may be applied to the volume to further enhance the boundaries of the channel, after which the user can finally set the threshold range that specifies the target voxel values for Surface Wrapping.

It is true that, with the exception of the final filtering procedure and setting the target voxel range, the domain transformation process described above could be equally useful preparation for many semi-automatic segmentation techniques in addition to Surface Wrapping; therefore, should all the preceding steps truly be considered aspects of voxel classification? This remains an open question.

It could be argued that even if binary voxel classification does perform equally well for different types of data, the phrase “equally well” could also be read as “equally poorly.” One approach to mitigating such potential disadvantages is discussed in Section 3.6.2, but this issue has not yet been investigated in detail.

What voxel classification cannot do, as Surface Wrapping is presently implemented, is provide voxel-level guidance for the behavior of the mesh deformation algorithm. For techniques such as active contour models, the image data is preprocessed to calculate the “energy” at each voxel, creating a gradient field that provides information at every position about the proximity to an edge [27]. In Live-Wire and related techniques, this energy calculation is updated dynamically as the user interacts with the software, allowing the edge-detection algorithm to adapt as needed to changing

characteristics of the image at different locations [36]. Whether or not the binarization of the data in the voxel classification step of Surface Wrapping necessarily results in the loss of information that could improve the results of the mesh deformation step is another open question.

### 3.5 Constructing the Basis Mesh

The next step following voxel classification is the creation of the *basis mesh*: a 3D surface that approximates the boundaries of the region of interest in the volume. This is the mesh that will be conformed to the segmented feature in the final mesh deformation step of Surface Wrapping, and like the elastic surface used to shrink-wrap a physical object, the basis mesh must be defined entirely on one side of the borders of the region of interest. When segmenting a bounded feature (e.g. an organ in an anatomical volume, or a geobody in a seismic volume), the basis mesh can be either contracted from the exterior of the object, or expanded to fill the interior. When segmenting a non-bounded, approximately planar surface (e.g. a horizon or fault), the basis mesh can be defined on either side, but should not overlap with the target surface.

In this text, a *mesh* is specifically defined as an unstructured, triangulated mesh in which each triangle stores an ordered vector of its component vertices, and each vertex maintains an unordered set of adjacent vertices (sometimes classified as a *Face-Vertex* mesh).

The *basis mesh* must also possess the following four characteristics. First, the spatial distribution of vertices must be approximately uniform across the entire surface. Second, the density of vertices must be high enough to capture the desired level of detail in the region of interest (a requirement which is explained in Section 3.6). Third, the number of neighboring vertices must be approximately the same for each vertex, as explained in Section 3.6. Fourth, the winding order of the vertices must be



consistent for all triangles, such that the face normals (as calculated relative to a fixed vertex ordering) point in the same relative direction across the surface of the mesh.

This section describes two approaches for constructing a suitable basis mesh, one of which is used to create meshes with closed boundaries, and the other for creating open-bordered meshes suitable for draping onto laterally-extensive surfaces. These are not the only means by which a basis mesh can be created, however; any tool can be used, provided that the resulting mesh has the required structural characteristics.

### 3.5.1 Basis Mesh Construction for Surface Wrapping

3D modeling software is widely acknowledged as challenging to design so that it is easy to use, particularly given the limitations of the normal target machine: a personal computer with a 2D display, a 2D pointing device (plus a standard keyboard), and system-level software interfaces that are all but exclusively biased towards 2D interaction metaphors [20]. Fortunately, the scope of the requirements for Surface Wrapping's mesh construction tool is limited compared to a full 3D modeling package, but many of the same difficulties still apply.

As explained in Section 3.1, the current interface was primarily designed to simplify the segmentation of channel- or canyon-like objects, with capabilities such as onion skinning (Section C.3.1.2) added later to accommodate objects with more complex vertical boundaries. Beyond working with arbitrary 3D shapes, the software needed to make clear which parts of the volume are covered by the basis mesh, and where the bounds of the mesh are located in relation to the target voxels. In areas of the volume where the edges of the region of interest are ambiguous, it had to be possible for the user to essentially revert to manual segmentation, while still making it fast to lay out a rough border surrounding those edges which are clearly-defined. It was also desired that, ideally, the interface should not be so completely foreign to users of existing segmentation or interpretation software that acceptance would be hindered, unless

a radically different design would bring equally great improvements to the capabilities of the system.

Koo et al. [31] demonstrated that an initial surface constructed from the faces of a set of cubic cells could be conformed to the bounds of a point cloud using a deformable surface scheme similar to that which had been implemented for Surface Wrapping, allowing for shapes of arbitrary complexity to be represented without exhibiting the blockiness of the initial mesh in the final result. While this process automatically generates the initial surface based on which cells contained points from the cloud, an approach which would be ineffective if applied to volumes in which noise or other undesired features are present, the concept of building a basis mesh from the exterior faces of discrete cells was used as the foundation of a tool in which such a surface can be defined by hand.

The *Mesh Painter*, as the tool was eventually named, bears a superficial resemblance to raster-based painting applications (e.g., MacPaint or MS Paint), and even more similarity to manual segmentation tools in which a region of an image is delineated by densely filling its interior (rather than tracing an outline). The Mesh Painter presents a greyscale slice image from the volume in the X, Y, or Z plane (the orientation is selectable by the user) in which target voxels are highlighted in bright red, with a round “paint brush” used to fill in the region covered by the basis mesh (Figure 3.2).

The major difference between the Mesh Painter and outwardly similar 2D segmentation tools is that the paint brush operates over a range of slices simultaneously, creating a full 3D mesh in a single pass. This range of slices is specified by the user: for example, when interpreting a channel, the user would first determine which slices contain the upper and lower bounds of the channel and set the range of the brush accordingly. More complex meshes can be built up in layers by painting a layer in one range of slices, then moving to a different range and painting a second layer, and so on.

This “3D painting” interface is made practical by two factors: the behavior of the

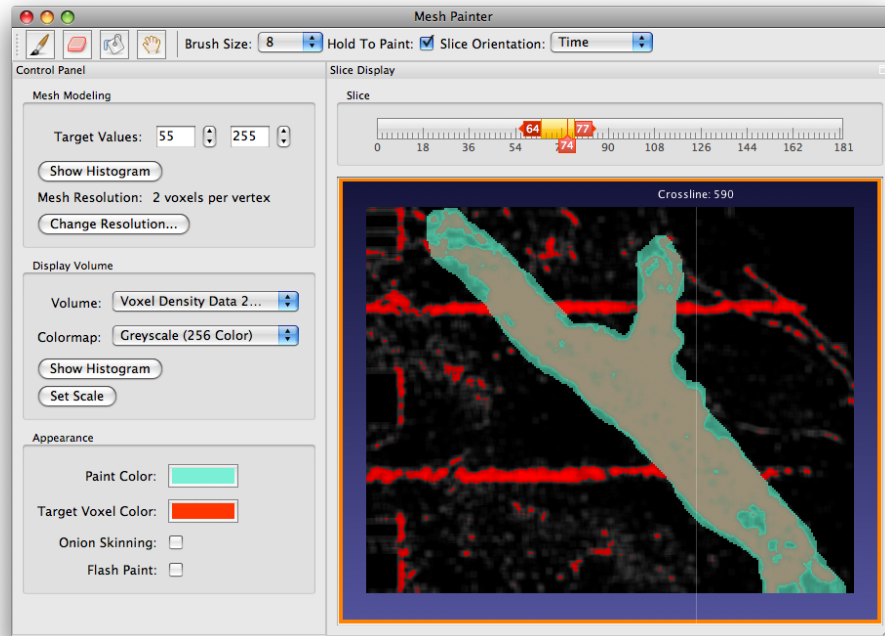


Figure 3.2: The Mesh Painter user interface.

mesh deformation algorithm, and the representation of the painted region as a collection of discrete cells, rather than 2D parametric curves. Because the mesh deformation algorithm does not require the boundary of the basis mesh to be placed close to the bounds of the region of interest—only that it be placed on either the inside or outside of the bounds—the user can be quite imprecise without negatively affecting the results. The rasterized representation of the painted region enables the user to fine-tune the border in small areas of the volume without requiring duplication of effort: for example, an initial “rough draft” of the basis mesh covering all relevant slices can be painted in one pass first, after which the user can narrow the range of the brush and perform detailed cleanup, on a slice-by-slice basis if necessary. This representation also allows the user to switch arbitrarily between different slice orientations, hollow out spaces in the interior of the painted region to simultaneously interpret inclusions, and easily merge portions of the painted region that were previously defined as separate bodies.

### 3.5.2 Bounding the Painted Slab

In the first version of Surface Wrapping in which 3D painting was used to define the basis mesh, the depth of the brush was specified as a constant number of slices on either side of the current slice. While this approach seemed obvious and straightforward enough at first, in practice it was all but completely unusable. Often the boundaries of the region of interest have a vertical slant, making it important to be able to position the painted region relative to either the top or bottom slice in a slab, whereas painting with respect to the image shown in the middle slice would result in the basis mesh intersecting the upper or lower boundary. The other major disadvantage is that it makes the process of building up a multiple-layer region more difficult by relying on the user to correctly calculate the center slice of the new layer without causing gaps between layers or undesired overlapping regions.

#### 3.5.2.1 The range slider control

The solution was to allow the user to explicitly set the first and last slices of the slab, thus separating the viewing of slices from the range over which the brush operates.

The simplest way (from a development perspective) to provide this feature would be to introduce two number-entry text boxes in which to enter the upper and lower bounds of the brush. A slightly better approach might be to use a pair of sliders instead of the text boxes, which would at least visibly restrict the range of acceptable values.

However, there are several unintuitive rules that must be enforced about how the brush range can be specified. First, the lower slice must always be less than the upper slice. Second, the upper and lower bounds of the brush must be aligned with the painted cell grid; for example, if the cell size is 4, the bounds of the brush can only

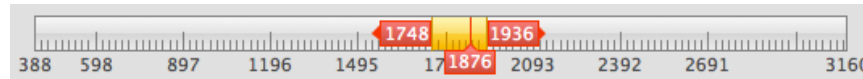


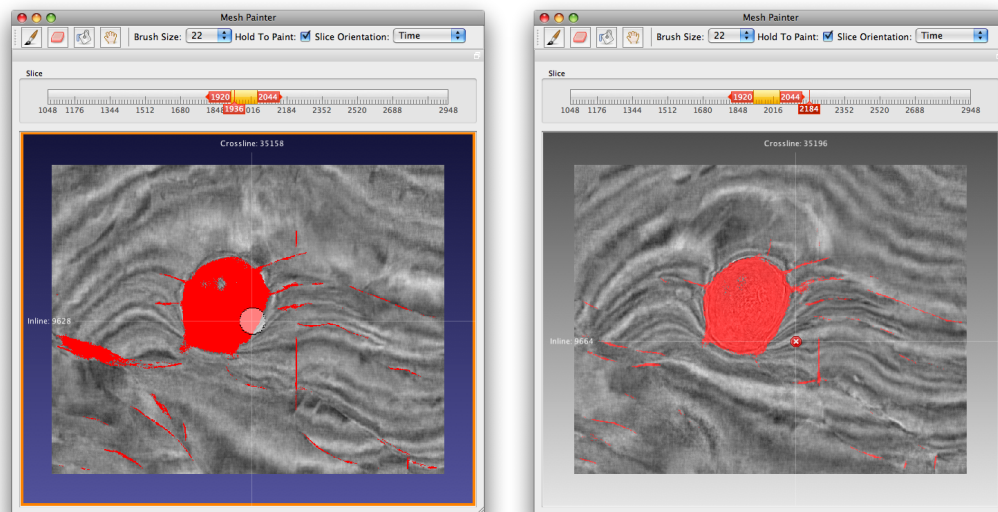
Figure 3.3: The range slider control.

fall on slices that are multiples of 4—except at the upper limit of the volume, if the number of slices in the volume is not itself a multiple of 4 for the current orientation. Third, the slice numbers are typically not the array indices of the slices in the volume, but instead map to the coordinate space of the survey from which the seismic volume was obtained, which may require using floating-point numbering for the slices. A final consideration, though not strictly a requirement, is that it would be helpful to provide some obvious way for the user to see at a glance when the current slice falls within the range covered by the brush.

These concerns were addressed by the development of a custom GUI widget, referred to as a *range slider* control (Figure 3.3). The range slider is not a unique concept (in fact, the appearance of the control was partly inspired by a similar control in Apple Inc’s GarageBand application [2]) but there has been no general standardization of such a control. The implementation used in the Mesh Painter allows the user to set the range of the brush by dragging the thumbs to the left and right of the orange region, or by dragging the orange region itself to reposition both the upper and lower limits simultaneously. The thumb at the bottom is used to set the current slice, and displays a red vertical line in the center that indicates the slice’s position relative to the brush range. All thumbs display the exact number of the corresponding slice, and can be manipulated by either dragging with the mouse, or by single-clicking to select a thumb and using the left and right arrow keys to change the position.

### 3.5.2.2 Indicating when the current slice is “in range”

Despite the presence of the red line indicating the position of the current slice relative to the range of slices affected by the brush, it was clear from both the author’s own experiences with the software and casual observation of other users that this feature alone is insufficient to indicate whether or not the current slice is “in range.” Particularly when using the arrow keys to move between slices, one often finds oneself attempting to paint on a slice that falls outside of the selected range, and at least momentarily caught off-guard when no paint is applied to the current slice. It seemed reasonable to speculate that because the range slider requires close examination to determine if the current slice is in range during the actual act of painting, one or more additional cues were needed to indicate the in range/out of range conditions.



**Figure 3.4:** The Mesh Painter window, showing the visual state when the current slice falls within the selected range (left), and outside of the selected range (right).

Several different visual cues were added to increase the user’s awareness of whether the current slice is in or out of range. First, a background gradient was added to the slice viewer, which changes from greyscale to color when the current slice moves

into the brush range. While this is useful for smaller datasets—the marked change from grayscale to color in the periphery is noticeable even when the user’s attention is focused on the center of the slice—this background is often completely obscured by slices from larger volumes, so a thick, bright-orange border was also added to the foreground around the slice display which only appears when the brush is in range. These cues proved to be helpful, but even though they are very large visual indicators, they were still found to be too easy to ignore in practice when the user’s attention is mainly fixed on the cursor.

A third visual cue was added, attempting to take the user’s locus of attention [45] into consideration: when the current slice is out of range, the cursor is changed from a circle (indicating the size of the paint brush) to a bright red stop sign-shaped symbol with an X through the center. This improved the situation significantly, but perplexingly, it was still not completely sufficient: when using Surface Wrapping, users continued to occasionally miss this cue.

After further consideration of the root of this problem and its uncanny persistence, it was hypothesized that when painting, one’s attention is more often focused on the target voxels than on the brush itself, and that changing the appearance of the target voxels when the slice is out of range might be another good visual cue. Target voxels still need to be highlighted so that the user can determine where to set the bounds of the brush, so instead of completely hiding the target voxels, the highlighting was set to 50% transparent when the slice is out of range (Figure 3.4). This mechanism, finally, seems to have ended the problem of figuring out when the slice is in range: since implementing the semi-transparent target voxel cue, informal observation of users has indicated a great reduction, if not complete elimination, of this source of confusion.

### 3.5.3 Basis Mesh Construction for Surface Draping

During the research and development of the Surface Wrapping concept, the original Surface Draping horizon interpretation technique has also been revisited. Surface Draping differs from Surface Wrapping primarily in the assumption that the initial coarse interpretation results in a non-closed, laterally extensive surface, as opposed to a closed bounding mesh. It was felt that the new mesh deformation functionality—such as simulated mesh elasticity (Section 3.6) and surface permeability (Section 3.6.1)—that had been developed for the segmentation of bounded objects would be equally applicable to a “draping” mode of operation, with no algorithmic changes required; therefore, the most significant research topic related to Surface Draping is the means by which the user defines the basis mesh in relation to the volume data.

Section B.2 discusses past approaches to creating the basis mesh for Surface Draping. The method that is currently being used did not involve the construction of a new user interface (or any other software development specific to this thesis), but instead takes advantage of a built-in Insight Earth process called Point Set Infill. This process is used to generate a smoothly-contoured surface based on an arbitrary, sparse point set, assuming that no areas of the surface overlap in Z. The surface is generated by first fitting a series of 2D splines to the points along a user-specified axis, creating a set of smooth “ribs.” A second set of densely-spaced 2D splines is then fit across these ribs in the orthogonal direction, and new points based on these splines are added to the point set, filling the gaps in the original point set with these new points at voxel-unit spacing.

In conjunction with Surface Draping, this process is used as follows. Working on either inline or crossline slices, the user manually creates a new, sparse point set, drawing a series of points slightly above the horizon of interest on multiple slices (Figure 3.5). When this initial interpretation is complete, the user then applies the Point



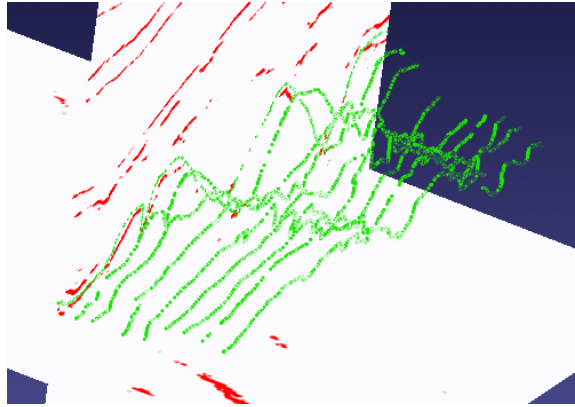


Figure 3.5: A coarse initial interpretation defined as a point set.

Set Infill process to the manually-defined point set, increasing the density of the points (Figure 3.6). A Z-grid is then created (using another built-in process) by mapping the Z position of each point to a corresponding cell on a horizontal, voxel-spaced grid (Figure 3.7), which is then converted to a triangulated mesh that serves as the basis mesh for Surface Draping.

This is unlikely to be the final technique for creating a drapable basis mesh, but it is clearly a step in the right direction. Initial results have been promising: this approach allows Surface Draping to be applied to more complex horizons than was possible using an earlier spline-based drawing interface (Section B.2.2), yet it can still be used rapidly enough to offer significant improvements in both speed and detail over manual and semi-automated techniques for some types of horizons. Specific results are provided in Section 4.1.1.1.

### 3.6 Mesh Deformation

The final step of Surface Wrapping is mesh deformation, the goal of which is to accurately conform the basis mesh to the bounds of the region of interest. The mesh deformation algorithm treats the mesh as a connected, elastic surface and the target

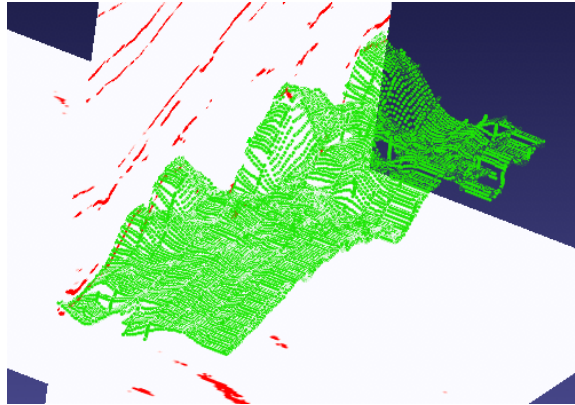


Figure 3.6: The output of the Point Set Infill process.

voxels of the volume data as a solid object through which the surface cannot pass.

The deformation process is user-guided: several parameters are exposed in the GUI that control the performance of the algorithm, and the global stopping condition is interactively supervised, employing a preprocessing scheme that caches results to improve responsiveness.

The mesh deformation algorithm is iterative, performing a series of calculations for each vertex that establishes the next location and determines when the vertex has reached a target voxel:

- (1) A *projection* calculation is used to move the vertices in the direction of the region of interest.
- (2) A *centering* calculation is used to help regularize the spacing between vertices and to approximately simulate surface elasticity.
- (3) The updated vertex locations are determined by taking a weighted average of the results of the previous two calculations.
- (4) The voxel value at each updated vertex location is checked to determine which vertices have reached the bounds of the region of interest.

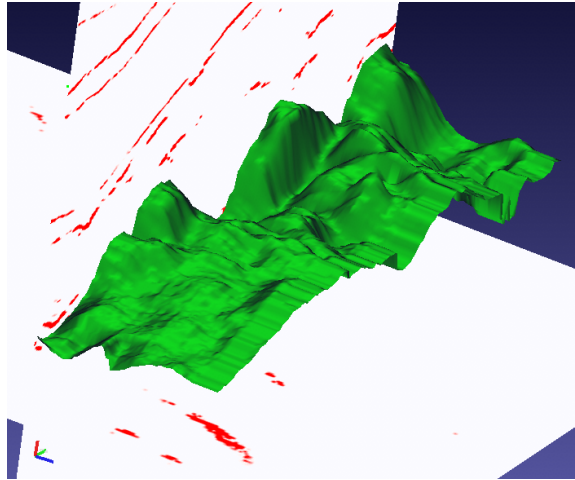


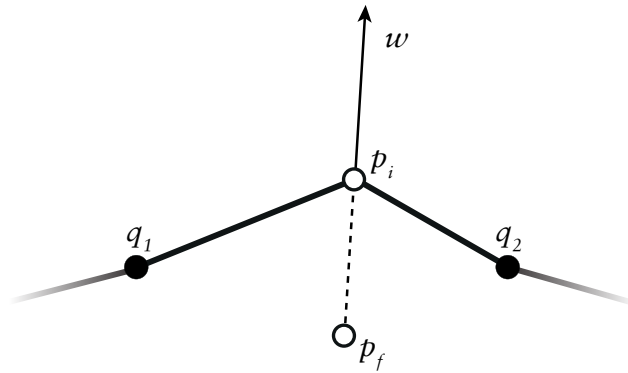
Figure 3.7: A Z-grid created from the infilled point set.

For a mesh  $M$ , the complete state of the mesh is cached at each iteration of the algorithm, with  $M_0, \dots, M_n$  as the set of mesh states for  $n$  iterations, where  $M_0$  represents the state of the basis mesh. State information is maintained at the vertex level, and consists of the position in Cartesian coordinates, and a boolean flag indicating whether or not the vertex is fixed in place (with an initial value of false for all vertices); neighbor connectivity is always constant. Each mesh state  $M_{i+1}$  is derived from the previous state  $M_i$  by application of the series of calculations outlined above to each vertex  $v$ .

In the *projection* calculation, a position  $p_f$  for each vertex  $v$  is determined as

$$p_f = p_i + 0.5ws \quad (3.2)$$

where  $p_i$  is the location of vertex  $v$  in  $M_i$ ,  $w$  is the normal vector of  $v$ , and  $s$  is either 1 or -1 (Figure 3.8). Unit length is the length of the side of a voxel, assuming cubic voxels for the purposes of the calculation of  $w$ . The constant factor 0.5 was arrived at through experimentation, as better results were observed when the distance between  $p_i$  and  $p_f$  was limited to half the voxel length. The factor  $s$  controls whether



**Figure 3.8:** Mesh deformation: The projected vertex location  $p_f$  is calculated relative to the normal vector  $w$ .

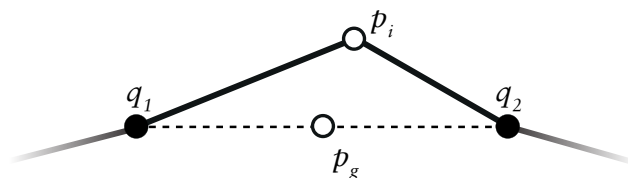
$p_f$  is calculated by projection in the direction of the normal or opposite the normal, the practical effect of which is that a positive value of  $s$  causes the mesh to expand, while a negative value results in contraction.

In the *centering* calculation, a position  $p_g$  for each vertex  $v$  is determined as

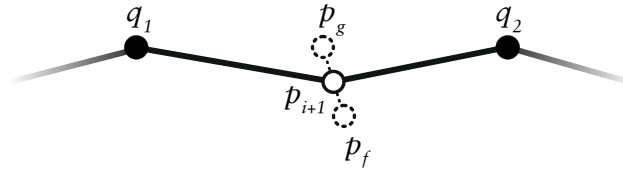
$$p_g = \frac{|\sum_{i=1}^n q_i|}{n} \quad (3.3)$$

where  $q_1, \dots, q_n$  are the positions of all immediate neighboring vertices of  $v$  in  $M_i$  (Figure 3.9). By finding a centered position for each vertex relative to its neighbors, this calculation has the effect of smoothing and “tightening” the mesh, providing a simple model for surface elasticity.

To determine the position  $p_{i+1}$  for each vertex  $v$  in the mesh state  $M_{i+1}$ , a vector



**Figure 3.9:** Mesh deformation: The centered vertex location  $p_g$  is calculated relative to the neighboring vertices  $q_1$  and  $q_2$ .



**Figure 3.10:** Mesh deformation: The final vertex location  $p_{i+1}$  is calculated relative to the projected location  $p_f$  and the centered location  $p_g$ . In this illustration,  $p_{i+1}$  is shown halfway between  $p_f$  and  $p_g$ , implying an elasticity factor  $e = 0.5$ .

of motion  $r$  is first calculated:

$$r = p_i - (p_f(1 - e) + p_g e) \quad (3.4)$$

where  $e$  is a user-specified elasticity factor between 0 and 1 which governs the relative weighting of the projection and centering calculations (Figure 3.10). Higher values of  $e$  result in a stiffer, smoother surface, while lower values simulate a more flexible surface that conforms more accurately to fine details in the region of interest.

The position  $p_{i+1}$  is then calculated according to

$$p_{i+1} = \begin{cases} p_i + r, & |r| \leq 1 \\ p_i + (r/|r|), & |r| > 1 \end{cases} \quad (3.5)$$

which restricts the updated vertex position to falling within one unit length of its previous location in  $M_i$ . Finally, the voxel value at  $p_{i+1}$  is checked against the user-defined target voxel range to determine whether or not  $v$  should be fixed in place in  $M_{i+1}$ . For a voxel value  $u$  and upper and lower thresholds  $t_{\text{lower}}$  and  $t_{\text{upper}}$ ,  $v$  will be flagged as fixed if  $t_{\text{lower}} \leq u \leq t_{\text{upper}}$ .

This algorithm scales well to large volumes, as the performance is dependent only upon the number of vertices in the mesh and the number of iterations, not the dimensions of the volume. The algorithm takes  $O(mn)$  time for  $m$  iterations and a

mesh of  $n$  vertices, and as  $n$  dominates for a mesh of any practical size, the complexity is therefore  $O(n)$ .

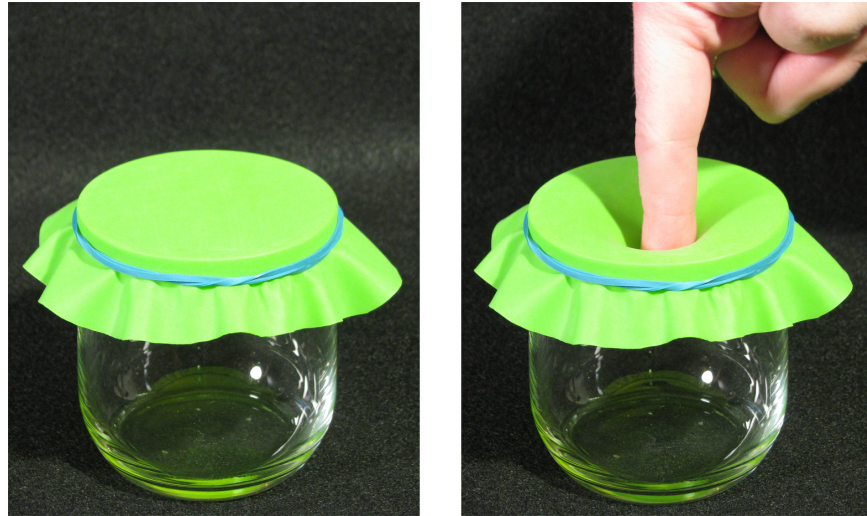
### 3.6.1 Surface Permeability

Noise in volumetric data is a common problem for automatic and semi-automatic segmentation techniques, particularly when scattered outlier voxels fall near the region of interest [9]. Shortly after it became practical to apply Surface Wrapping to actual seismic and medical volumes, it was found that noise in the data frequently caused “spikes” in the final mesh: points in between the basis mesh and the target boundary where one or two vertices would become prematurely fixed on an outlier voxel. Filtering the image to remove isolated foreground voxels (e.g. using a mean or median filter, or by rejecting any voxels that do not pass a minimum neighbor connectivity threshold) is often not desirable, because such techniques may also filter out isolated voxels that are actually part of a weakly-imaged boundary that should be preserved as much as possible.

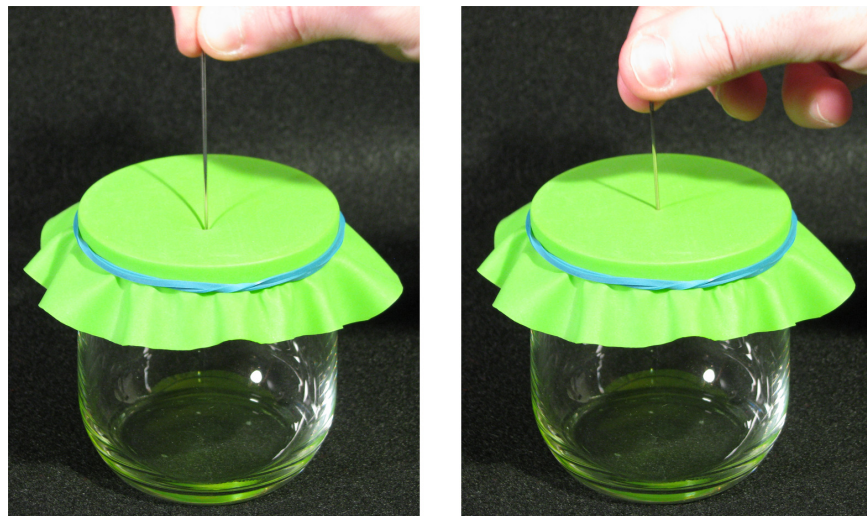
In order to address the problem of noise-related spikes while still allowing poorly-imaged boundaries to contribute to the shape of the final mesh, an improvement to the mesh deformation algorithm so that the mesh is allowed to pass through outlying foreground voxels. This technique, referred to here as “surface permeability,” is conceptually simple: if a fixed vertex is determined to be too “sharp,” its designation is reverted to unfixed, and it is allowed to move freely during the next iteration of the algorithm.

The effect of this property is easily illustrated with physical materials. A blunt object will tend to deform an elastic surface without breaking it (Figure 3.11). A sharp object can also deform the elastic without breaking it, but less force is required to cause it to penetrate the surface (Figure 3.12).

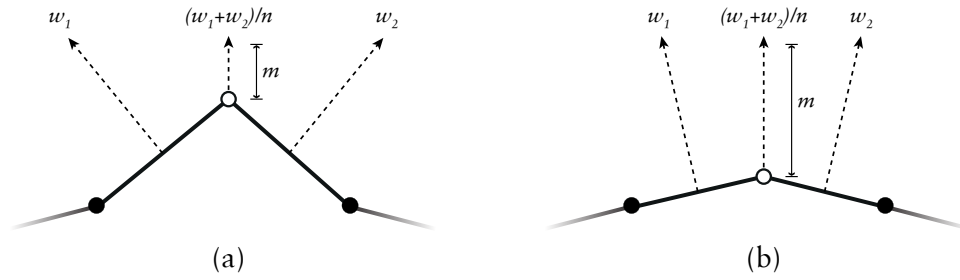
As the last step of each iteration of the mesh deformation algorithm, a sharpness



**Figure 3.11:** An elastic surface (left), and the elastic surface deformed by a blunt object (right).



**Figure 3.12:** An elastic surface can be deformed by a sharp object (left), but is easily punctured (right).



**Figure 3.13:** (a) The mean of the surface normal vectors  $w_1$  and  $w_2$  has a magnitude  $m$  that is closer to 0 for a “sharp” vertex. (b) For a “blunt” vertex,  $m$  is closer to unit-length.

measure  $m$  is calculated for each fixed vertex  $v$  as

$$m = \frac{|\sum_{i=1}^n w_i|}{n} \quad (3.6)$$

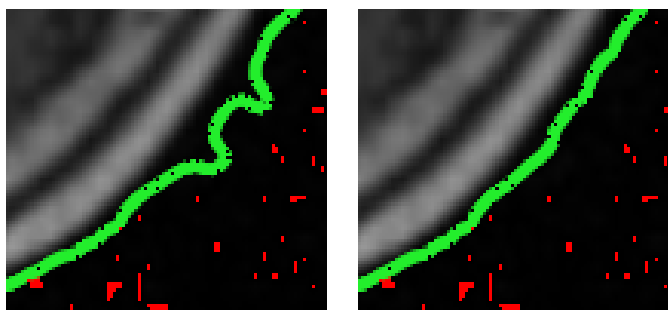
where  $w_1, \dots, w_n$  are the surface normal vectors for all triangles adjacent to  $v$ . This calculation produces a value between 0 and 1, where lower values indicate a sharper vertex (Figure 3.13). This value is then tested against a user-defined threshold value using the predicate

$$f(m, t) = \begin{cases} \text{true,} & \text{if } m \leq t \\ \text{false,} & \text{else} \end{cases} \quad (3.7)$$

with a result of true indicating that the vertex should be flagged as unfixed, and therefore allowed to move past the obstruction in the following iteration of the deformation algorithm.

It was originally expected that a more complex implementation of surface permeability would be required—e.g., calculating the sharpness of a cluster of two or more vertices simultaneously—but in practice, the simplistic approach described above performs very well. Figure 3.14 shows a cross section of a noisy MR volume to which Surface Wrapping has been applied, with surface permeability disabled (left), and en-





**Figure 3.14:** Surface Wrapping with surface permeability disabled (left), and enabled (right).

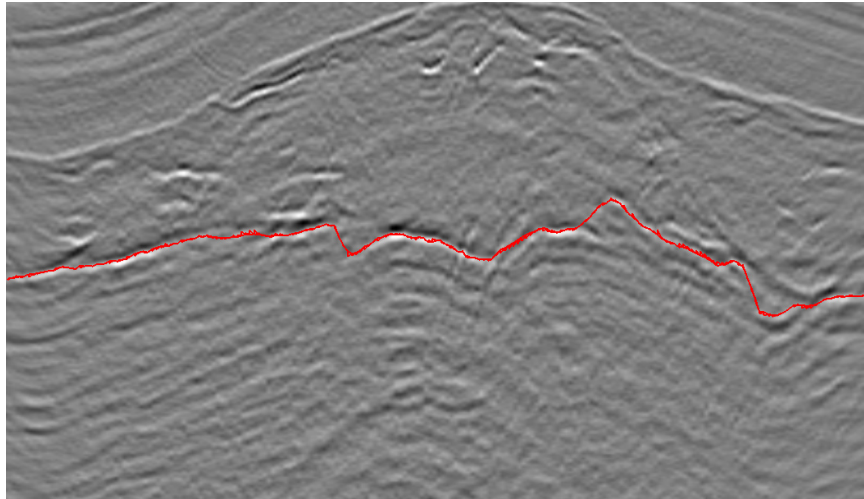
abled (right) using a permeability threshold of 0.7.

### 3.6.2 Snapping to Peaks and Troughs

As discussed in Section 3.4, one of the motivations for using a simple thresholding scheme to classify boundary voxels was to make Surface Wrapping as generalized as possible, relying on the user to choose the most appropriate preprocessing methods for the data domain and the type of object being segmented. Thresholding alone is not sufficient in every situation, however, and the shortcomings of this scheme are easily demonstrated by examining the problem of horizon interpretation.

Horizons in an amplitude volume are interpreted on either peaks or troughs, and an interpreted surface can be placed with sub-voxel precision by modeling the curve of the acoustic wave of each trace and snapping each point in the surface to the peak or trough of the waveform. Even if a volume were filtered such that voxels at all peaks and troughs were set to constant values regardless of their original amplitudes, Surface Wrapping would still be unable to achieve the desired level of sub-voxel accuracy.

The strategy used to address this particular problem was to add a post-processing step to each iteration of the mesh deformation algorithm, in which any vertex that encounters a target voxel in the current iteration is snapped to the nearest peak (or trough, as specified by the user). The vertex snapping modification leverages a built-



**Figure 3.15:** A horizon interpreted with peak snapping enabled. High values are shown here as light, and low values as dark.

in function of Insight Earth (also used by Insight Earth's horizon autotracker) which accepts a list of voxel values as input, and returns the index of the optimal voxel plus a floating-point offset indicating the precise peak or trough within that voxel's range. By fine-tuning the final position in this manner prior to flagging a vertex as fixed, results are greatly improved when using Surface Wrapping for horizon interpretation (Figure 3.15).

This situation is further complicated when interpreting salt bodies, because frequently the top surfaces of the salt need to be picked on the peaks, while the base surfaces must be picked on troughs (or vice versa). This not only eliminates the possibility of indiscriminately snapping to peaks or troughs across the entire mesh, it also implies that a single target value range for voxel classification is insufficient for providing the initial vertex stopping condition.

As large salt body interpretation has been an increasingly important focus for this research, another set of enhancements was introduced that builds upon the vertex snapping functionality described above. An optional second target voxel range was

added so that separate ranges can be chosen for peaks and troughs, using different highlight colors for the corresponding voxels in the Mesh Painter. The peak/trough snapping feature was also modified such that the user can assign different snapping behavior based on whether the surface is facing upward or downward. This was accomplished by changing the mesh deformation algorithm such that the sign of the Z component of a vertex's normal is checked prior to determining whether or not the vertex should become fixed, and choosing to snap to peaks or troughs accordingly.

Although this approach to vertex snapping is specifically designed to take advantage of the characteristics of seismic amplitude volumes, a more generalized solution, such as the image snapping technique introduced by Gleicher [18], could be incorporated to improve results when segmenting data from other domains.

### 3.6.3 Preventing Self-Intersections

An undesired artifact of the current mesh deformation algorithm is the occurrence of self-intersections: that is, places where the mesh passes through itself, causing undesired "folds" in the surface. This is generally encountered in areas where the basis mesh is too far from the surface of the region of interest, or where a gap in the data allows opposite sides of the mesh to converge. Self-intersections are highly undesirable, as they can cause problems for downstream applications when the resulting mesh is imported, and also make it impossible to correctly calculate the volume enclosed by the mesh.

Two approaches to solving this problem have been attempted. The first was to implement a triangle-triangle intersection test which was applied to every triangle in the mesh following each iteration of the deformation algorithm. However, this proved to be impractical for a number of reasons. First, the process of testing each triangle was too slow, even after extensive efforts to improve the performance, making it unsuitable for meshes with more than a few thousand triangles. Second, while the test is

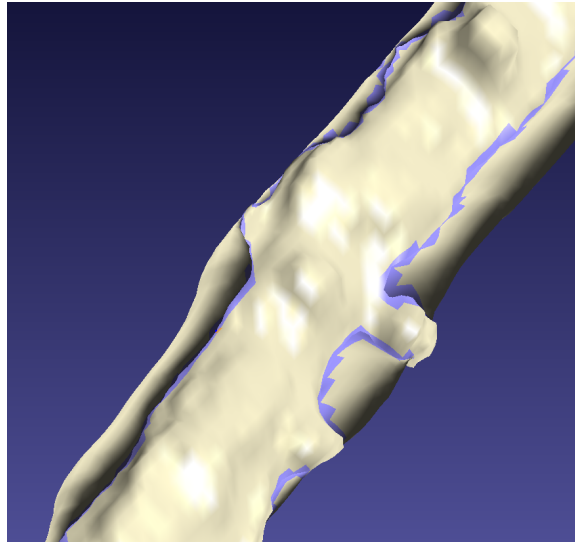
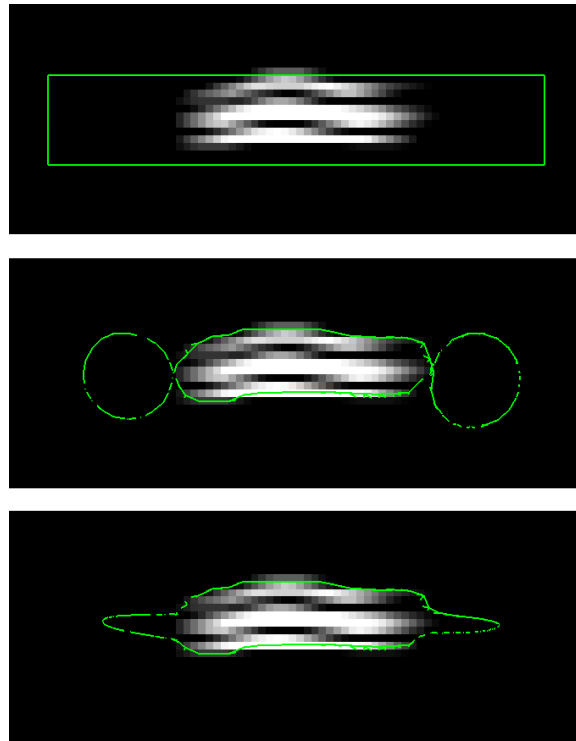


Figure 3.16: A self-intersecting mesh, with intersecting triangles highlighted in blue.

guaranteed to find all intersections, simply identifying intersections is not sufficient: these situations must also be resolved, such that at the end of each iteration, the mesh is left in a state where there are no intersecting triangles. A number of different mechanisms for resolving intersections were tried, but as all required multiple additional intersection tests for each iteration, the whole approach was deemed too inefficient to be practical. The triangle-triangle intersection test was, however, retained for diagnostic purposes: a display option for meshes was added which, when enabled, highlights all intersecting triangles (Figure 3.16).

The second approach that was implemented derives from a technique commonly used to prevent intersections in 3D animation of simulated cloth. Following each iteration of the mesh deformation algorithm, a spherical region around each vertex is checked for the presence of other vertices (excluding those vertices that share an edge), where the radius of the sphere is slightly greater than the initial distance between vertices. If another vertex is found within this sphere, the vertex being examined is flagged. After every vertex has been checked, the positions of those which were flagged



**Figure 3.17:** Cross-section of a bounding mesh for a simple channel. Top: The initial bounding mesh. Middle: The mesh after 30 steps of the standard deformation algorithm. Bottom: The mesh after 30 steps with the option to prevent intersections enabled.

are reset to their positions in the previous iteration of the algorithm.

This technique proved to be effective for use in Surface Wrapping. It is reasonably efficient, in practice slowing down processing at a linear rate of roughly four times the normal speed. That the basis meshes for both Surface Wrapping and Surface Draping are created with roughly even spacing between all vertices addresses one of the prerequisites for using this technique: namely, that triangles in the mesh cannot vary too greatly in size. The other prerequisite is that vertices cannot be allowed to move farther than the radius of the proximity testing spheres at each iteration. The mesh deformation algorithm fulfills this requirement as well, because vertices are prevented from moving a distance greater than one voxel unit length per iteration.

Figure 3.17 shows the result of applying this technique to a poorly-constructed

basis mesh for a channel. The top image shows a cross-section of the basis mesh at a crossline slice in which the mesh extends far beyond the bounds of the channel. The middle image shows the mesh after 30 iterations of the standard deformation algorithm, exhibiting folds on the left and right where the mesh has intersected with itself. The bottom image shows the mesh after 30 iterations when the option to prevent self-intersections is enabled. While the mesh does not yet meet the left and right edges of the channel at this step, the problem of folding has been eliminated.

#### 3.6.4 Key Algorithm Design Motivations

The design of the current mesh deformation algorithm was arrived at over the course of several years, but it has remained quite simplistic from the beginning. Certain aspects of the algorithm have been directly influenced by existing techniques—arguably, the core of the algorithm is a mass-spring model [51] with zero mass and infinitely compressible springs—but the unique requirements of Surface Wrapping have led to some decisions that are not necessarily obvious.

The most important factor motivating the design of the algorithm has been that Surface Wrapping must be usable in practice to achieve an acceptable fit to the region of interest based on the guidance of the user. The restriction that the software must be “usable in practice” was the main cause for rejecting a rigorous, physically accurate elastic surface simulation; such a computationally-intensive approach would limit the tool’s interactivity, and it seemed unlikely that an accurate elastic model would necessarily produce the best results. Keeping the algorithm simple allows the user to freely experiment with different parameter settings and repeatedly alter the basis mesh without raising concerns about the cost, in computation time, of such adjustments.

Deformable models based on webs of connected particles often take advantage of the concept of a “rest state” of a surface: the normal shape of the surface when it is not being acted upon by outside forces. In the case of Surface Wrapping, there is no

logical rest state, because the desired shape of the mesh is not known until it has been deformed to fit the target data. Having a known rest state can be used to help prevent self-intersections and excessive drooping into concavities by, for example, limiting the maximum angle that is allowed for an edge relative to its rest state, but again, such techniques cannot be directly applied to Surface Wrapping.

In general, there has been a compromise between speed and accuracy, resulting in a deformable surface model that achieves good results quickly and predictably.

## Chapter 4

### Evaluation

Surface Wrapping is intended to be most beneficial when applied to data that are too noisy, too ambiguous, or too poorly imaged to be segmented using other automated or semi-automated techniques. There are two main criteria for assessing the extent to which the current implementation fulfills that objective: quantifiable accuracy of the results and time saved relative to other segmentation techniques, and the less quantifiable examination of the software’s usability in real-world scenarios. This chapter first examines the technical effectiveness of Surface Wrapping, then assesses the usability of the tool in its current implementation.

#### 4.1 Effectiveness

Quantifying the accuracy of any segmentation technique—whether manual, automatic, or semi-automatic—is widely acknowledged to be a difficult task [39]. Arguably, the core problem is that for non-synthetic datasets, it is generally not possible to find a “ground truth” against which results can be compared, as the only means of generating such baseline readings is by using some form of segmentation, and all forms of segmentation have known potential sources of inaccuracy.

Since one of the claims of this thesis is that Surface Wrapping can dramatically reduce the time required to accurately segment regions of interest in a volume, it would be ideal to simply measure how long it takes to achieve results that are com-



parable in accuracy to those produced by other methods, or to report how much the results diverged if similar accuracy could not be reached. For the reasons stated above, this type of evaluation is not possible, and therefore some compromises have been made relative to the ideal testing procedure. Instead of trying to reach target accuracy or speed objectives, Surface Wrapping and Surface Draping have been used in a manner typical of how they would be used in real-world interpretation scenarios. The results were then compared to previously interpreted surfaces when possible, measuring the vertical offset of the two surfaces at each vertex in the case of Surface Draping, and the 3D distance between each vertex of the wrapped mesh and the nearest point on the previously interpreted surface in the case of Surface Wrapping.

While such measures may be of some use, it is arguably more important (given the limitations of these sorts of comparisons) to show where the interpreted surfaces diverge and classify the reasons why these differences occur. In all tests that were performed, it is easy to find discrepancies in places where Surface Wrapping yielded undesired results, and just as easy to find instances where Surface Wrapping clearly produced better results, or where the correctness of either interpretation is debatable. These distinctions are illustrated in the following sections, which also discuss some of the more subtle strengths and weaknesses of Surface Wrapping and Surface Draping.

#### **4.1.1 Seismic Data**

##### **4.1.1.1 K12CD Base of Salt**

The K12CD volume was taken from a survey of a gas basin in the southern North Sea. The dimensions are 205 inline slices by 335 crossline slices, covering a physical region 12.3 km long and 8.375 km wide. A salt body extends laterally across the entire volume, and while the top of salt is smooth and fairly contiguous, the base of salt is highly fragmented and poorly imaged in some areas, making the base a good candidate

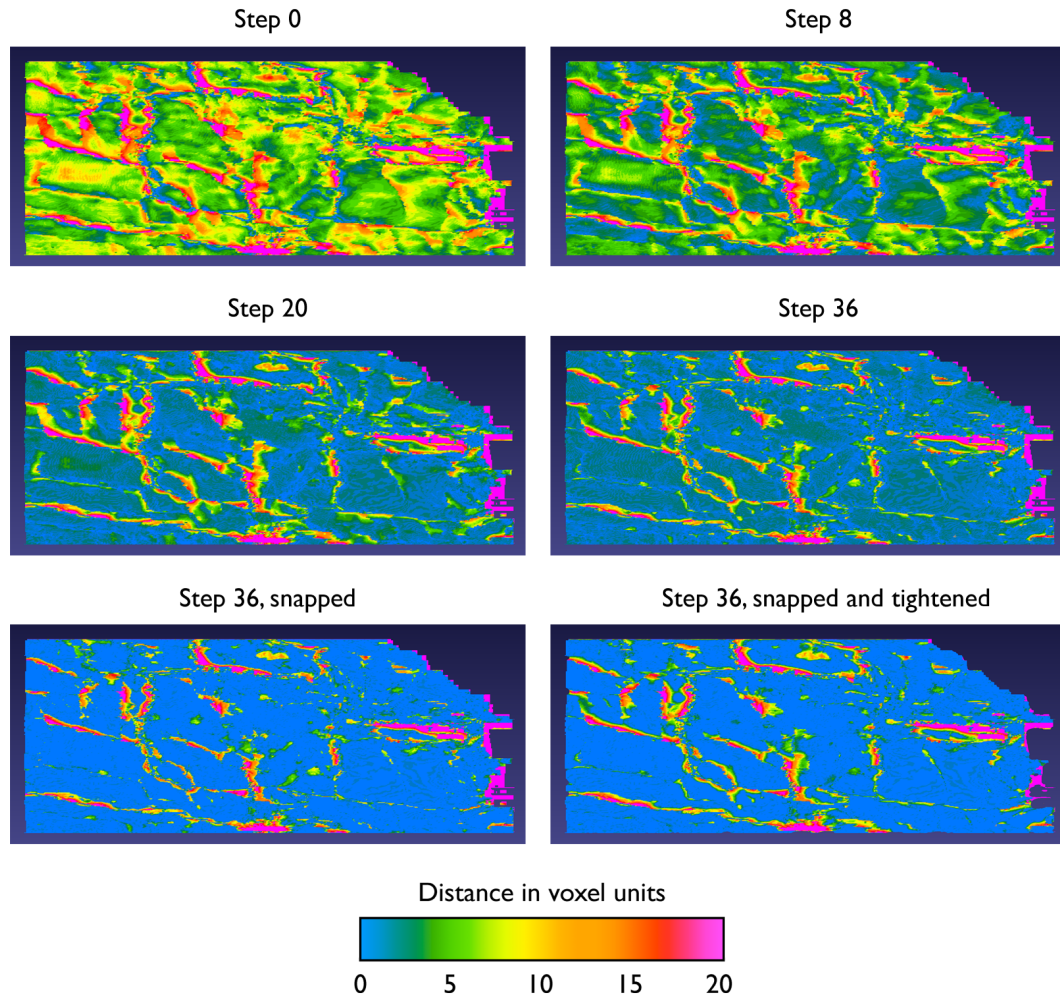
for Surface Draping.

An existing interpreted base of salt surface was available for K12CD, allowing direct comparison of results. The original interpretation was accomplished over the course of two months (approximately 320 hours in total) using a combination of manual picking and the use of a 3D autotracker, the output of which reportedly needed extensive manual cleanup.

To make use of Surface Draping most efficiently, the target volume was filtered to isolate strongly-reflecting troughs, a step which took roughly 45 minutes using a standard processing workflow. A basis mesh was then created in 18 minutes, picking above the region of interest using a graphics tablet as a pointing device. Various combinations of mesh deformation parameters were tested, and the final result was snapped to troughs and then tightened. Approximately 2.5 hours were required from start to finish.

Since both the original interpreted surface and the draped surface are Z-grids of the same resolution (one vertex per voxel in the X-Y plane), a quantitative comparison can be made by measuring the vertical separation between the two surfaces at each vertex location. Figure 4.1 shows an overhead view of the mesh at various steps in the deformation process, colored according to the vertical separation from the original surface at each X-Y coordinate. In Step 0, which shows the basis mesh, there are some places where the light blue color indicates zero offset; in most cases, this indicates a crossing of the two surfaces, where a different judgement call was made while creating the basis mesh relative to the original pick. Deformation was stopped at Step 36, after which the mesh was snapped to peaks, then tightened.

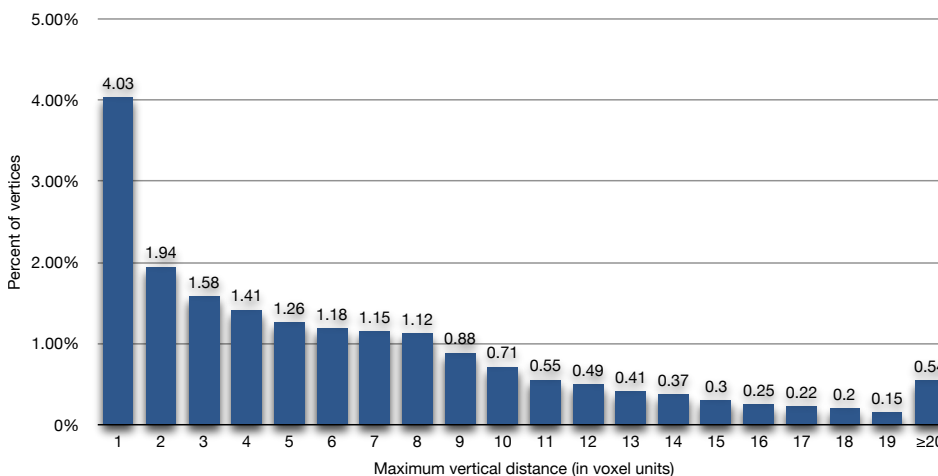
81.27% of the vertices in the draped mesh fall within 1 voxel unit of the original interpretation. Figure 4.2 gives the distribution of vertices where the vertical offset is greater than 1 voxel unit, indicating that, while the new surface is consistent with the original surface in most locations, it is far off the mark in a significant percentage of



**Figure 4.1:** Overhead view showing steps in the mesh deformation process as an approximately-picked initial mesh is draped onto the base of salt in the K12CD volume (Steps 0–36), then snapped to peaks (bottom left) and tightened (bottom right). The mesh is colored to indicate the vertical distance between the draped mesh and the manually picked surface.

locations. This can be partly explained by inconsistent human interpretation of the data in the original interpretation, but another reason was found that accounts for many, if not most, of the discrepancies.

During analysis of the results, an issue with the Point Set Infill process was found that either had not been encountered before, or had not previously caused noticeable problems for Surface Draping. While Point Set Infill is intended to fit a surface to all



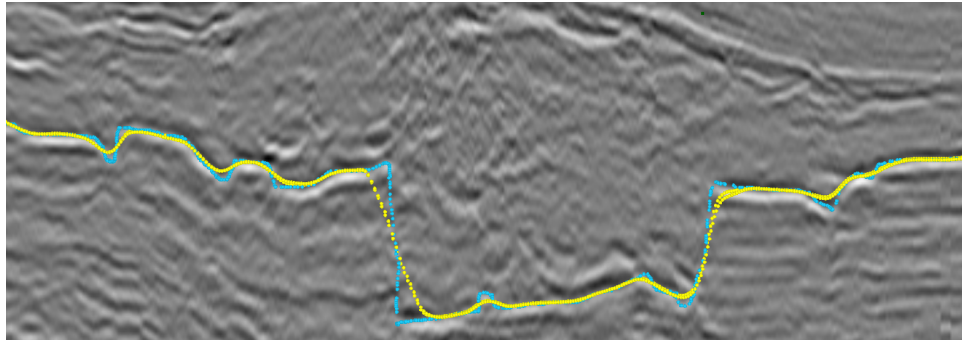
**Figure 4.2:** The percent of vertices in the draped K12CD surface that differ from the original interpretation. 81.27% of the vertices (not represented in this graph) fell within 1 voxel unit of the original interpretation.

points in its input data, in places where there was a sharp vertical drop from one line of points to the next, the process generated a smooth curve that undercut the upper ridge or smoothed over the base (Figure 4.3). Because the sharp ridges in the user's picks were not being honored, the corresponding regions of the final surface do not fit the data. The result of this problem is shown in Figure 4.4, in which the final surface is angled to show the fault faces where most of the inaccuracies were found.

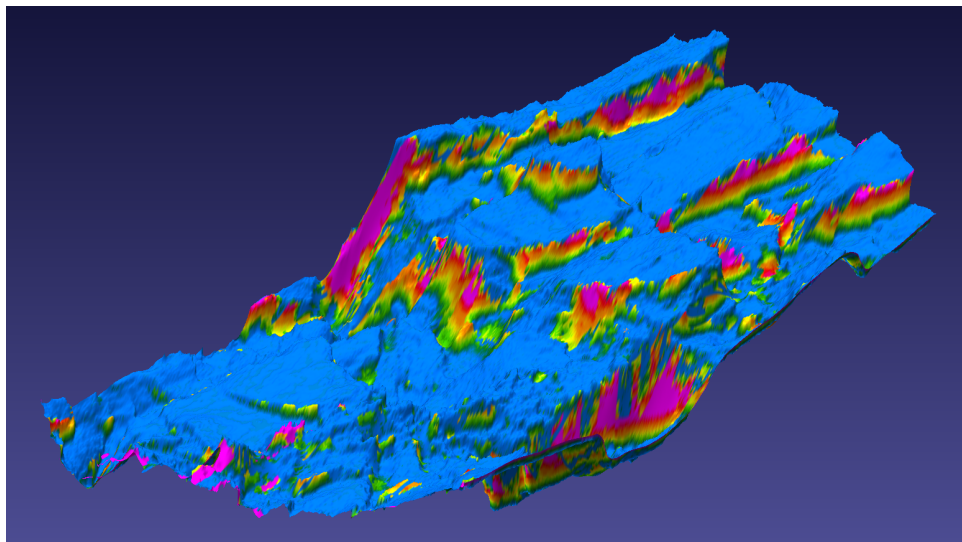
Despite differences in human judgement and problems related to overly-aggressive smoothing of the infilled point set, in most regions there was strong agreement between the draped surface and the original surface. A representative inline slice is shown in Figure 4.5, which exhibits areas of strong correlation, differences caused by disagreement of human interpreters, and differences caused by excessive smoothing.

#### 4.1.1.2 EI175 Salt Dome

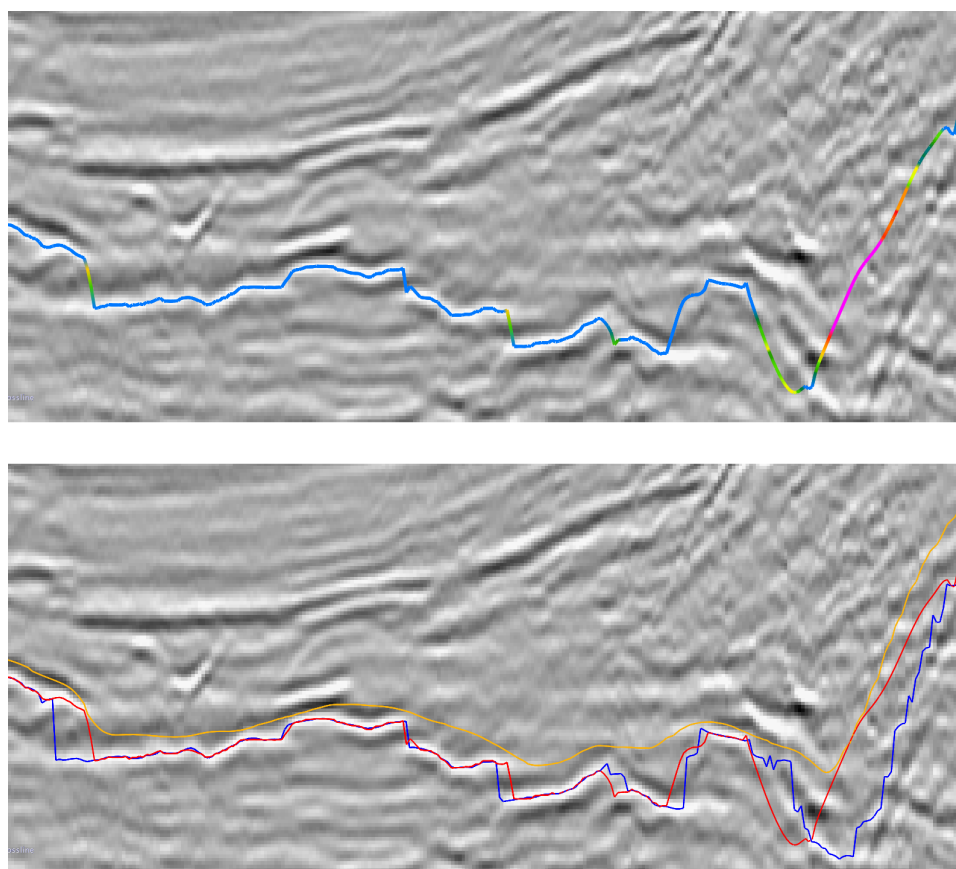
The EI175 volume was taken from a survey in the Gulf of Mexico, which contains a salt diapir in the center. The dimensions of the full volume (in samples) are



**Figure 4.3:** An inline slice from K12CD showing the manually-picked points in blue and the infilled points in yellow. Of particular note is the large vertical drop near the center of the slice, where the aggressive smoothing of the infilled points results in the basis mesh cutting through part of the target horizon.



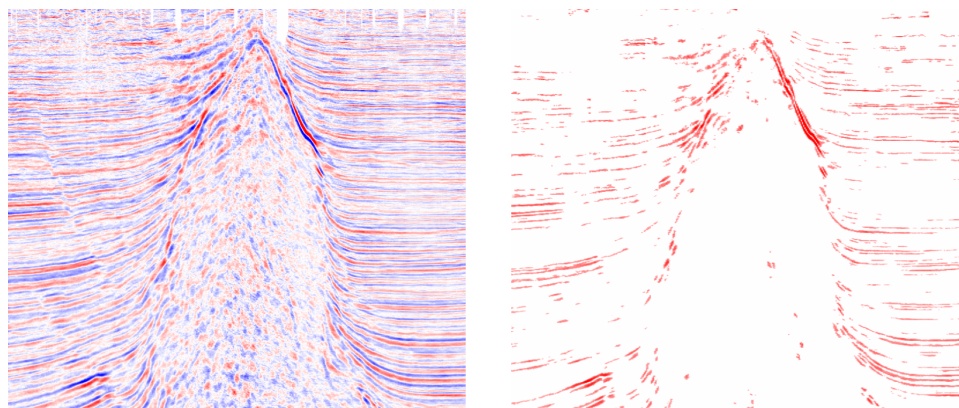
**Figure 4.4:** A shaded, orthographic rendering of the final draped (snapped to peaks and tightened) surface for the K12CD base of salt, colored to show the vertical distance to the manually-picked surface according to the scale provided in Figure 4.1.



**Figure 4.5:** Comparison of manually-picked and draped surfaces for K12CD on one slice. The top image shows the intersection of the draped surface (colored according to the scheme used in Figure 4.1) with an inline slice from the original amplitude volume (highest values white, lowest values black). The bottom image shows the manually-picked surface in blue, the basis mesh in orange, and the final draped result in red.

601 inline, 701 crossline, and 1000 time, although the salt body is contained within a subvolume of approximately 400 x 360 x 700 samples. The surface of the EI175 salt dome had been manually interpreted as a Z-grid in 1996, taking roughly 28 hours to complete.

Surface Wrapping was used to create a new pick for the bounds of the salt dome. Because of the chaotic reflections within the salt, it was important to filter the volume prior to creating the basis mesh while still preserving enough information that the boundary of interest was still pickable. With the exception of its top quarter, the salt

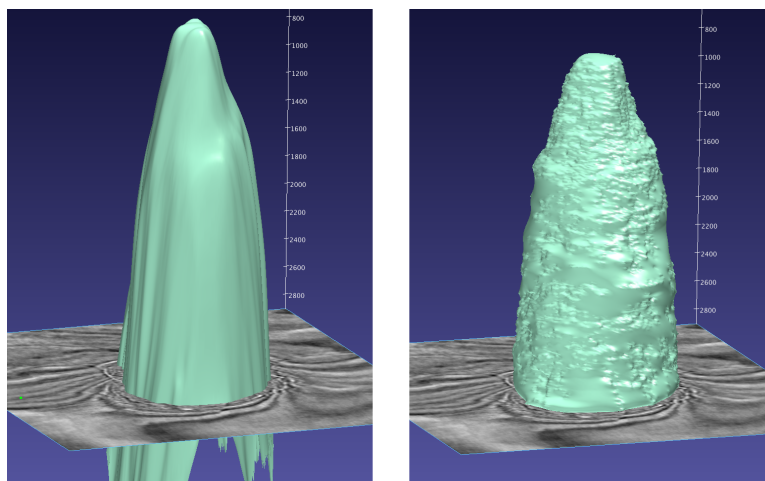


**Figure 4.6:** Comparison between the original EI175 amplitude volume (left) and the processed volume (right).

dome is best imaged by the terminations of the horizons that surround it, making the use of a traditional autotracker impossible. The volume was processed such that these terminations were retained as much as possible while much of the noise was cleared from the interior, making it possible to define the basis mesh inside the salt dome. Figure 4.6 shows a comparison between a crossline slice from the original amplitude volume and the same slice from the filtered volume.

After filtering the volume—a process which took approximately 45 minutes—a basis mesh was defined inside the salt dome and expanded to fit the region of interest in 30 minutes. In total, about 1 hour and 15 minutes was required to achieve a reasonable pick.

Unlike the K12CD base of salt surface, the original interpretation for the EI175 salt dome is known to be inaccurate: the salt dome has a complex vertical profile that cannot be captured using a single-Z-value surface because it includes overhangs. For this reason, a direct quantitative comparison between the two surfaces would be deceptive, since neither interpretation can be held as a ground truth. Both surfaces are shown from the same angle in Figure 4.7; the original pick is unrealistically smooth, while the new pick appears to be more plausible. A better means of comparison is



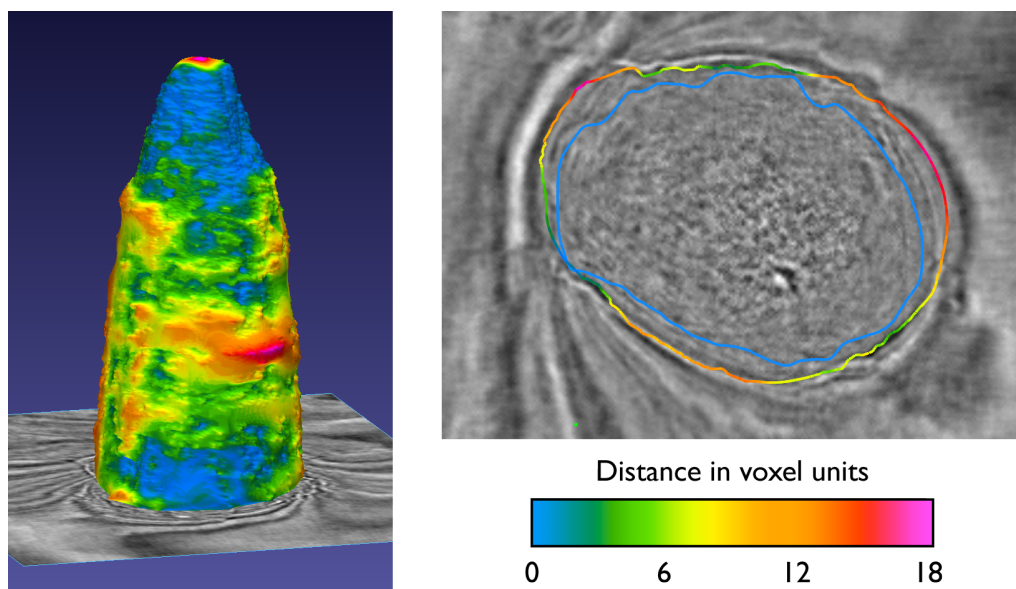
**Figure 4.7:** The original manual interpretation of the EI175 salt dome (left) compared to interpretation performed using Surface Wrapping (right).

provided in Figure 4.8, in which the mesh created via Surface Wrapping has been colored according to the 3D distance of each vertex relative to the nearest point on the original Z-grid. Examination of individual time slices almost invariably shows that in the places where the interpretations diverge, the wrapped surface achieves a better fit to the actual geobody than the original interpretation.

#### 4.1.1.3 Quad 15 Dewatered Shale

The Quad 15 volume shows part of a survey from the North Sea, and has been referred to as the “poster child” for Surface Draping because of its highly faulted interval of dewatered shale. The region of interest has a characteristic texture that resembles the cracked ground of a dried-out mud flat (Figure 4.9), which makes the use of auto-tracking methods highly inefficient for creating a contiguous surface: because there is no large, consistent peak or trough that defines the surface, only one small patch can be tracked per seed point. Each patch must be checked and cleaned up individually, and further editing must be performed if a gapless Z-grid is desired. While Surface Draping does necessitate a non-trivial amount of manual input to create the basis mesh, the





**Figure 4.8:** Final surface-wrapped mesh for the EI175 salt dome, colored according to the 3D distance to the manually-interpreted surface. The left image shows a 3D rendering of the mesh, and the right image shows a slice that is representative of the poor agreement between the results. The intersection of the manual pick with the slice is shown in light blue; in both images, the wrapped surface is colored according to the scale given at the bottom right.

total amount of time needed to create a complete, accurate surface is a small fraction of what would be required using other methods.

The Quad 15 volume contains 1000 inline and 1000 crossline slices, covering a square 12.5 km x 12.5 km area. Because of the size of the volume and the difficulty of picking a surface in the region of interest, there is no existing interpretation against which the results of Surface Draping can be compared. To illustrate the amount of time required to pick the surface using various techniques, 30 minutes was allotted for each of three different methods: manual picking, Insight Earth's 3D autotracker, and Surface Draping. All three tests were performed by the same user, with a graphics tablet serving as the pointing device. Lacking an existing surface created by an expert interpreter for evaluation of accuracy, the focus of these tests is primarily the mechanics of the different techniques.

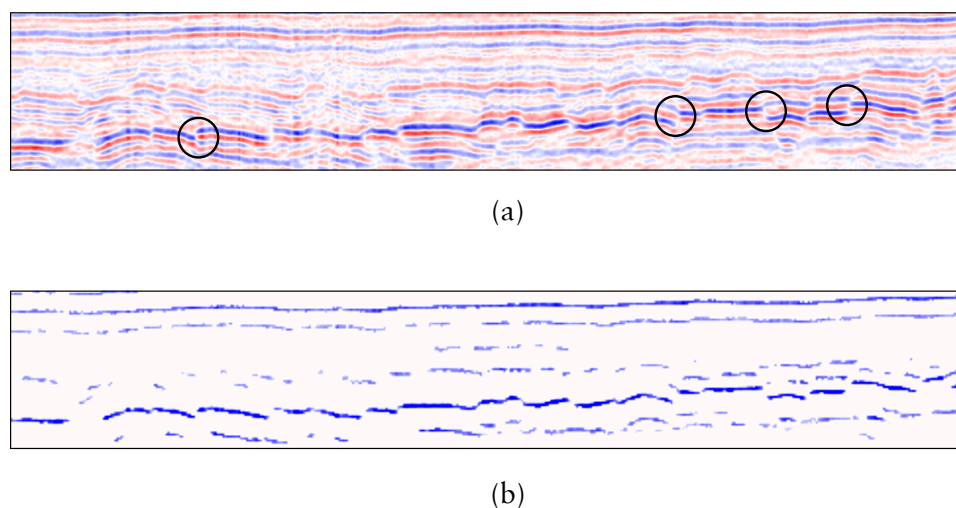


**Figure 4.9:** Cracked mud, similar in visual texture to the dewatered shale of the Quad 15 volume.

When manually picking the surface, 5 inline slices were completed after 30 minutes, covering 0.5% of the surface area. Proceeding at the same rate, it would take 100 hours to complete the interpretation.

The 3D autotracker fared better: approximately 3.3% of the surface area was covered after 30 minutes, yielding an estimated completion time of roughly 15 hours. This estimate, however, assumes that no further cleanup would be required to achieve an acceptable final pick, which is unlikely to be realistic based on the results observed during this session. The most conspicuous problem encountered was cycle-skipping, where a patch would grow across small faults and continue onto reflections lying on the same horizontal plane as the seed point, but which did not belong to the surface of interest (Figure 4.10a). A small amount of manual cleanup was performed—primarily using interactive control of the autotracker’s search radius—but it was readily apparent that significant additional editing would be necessary in a real-world scenario.

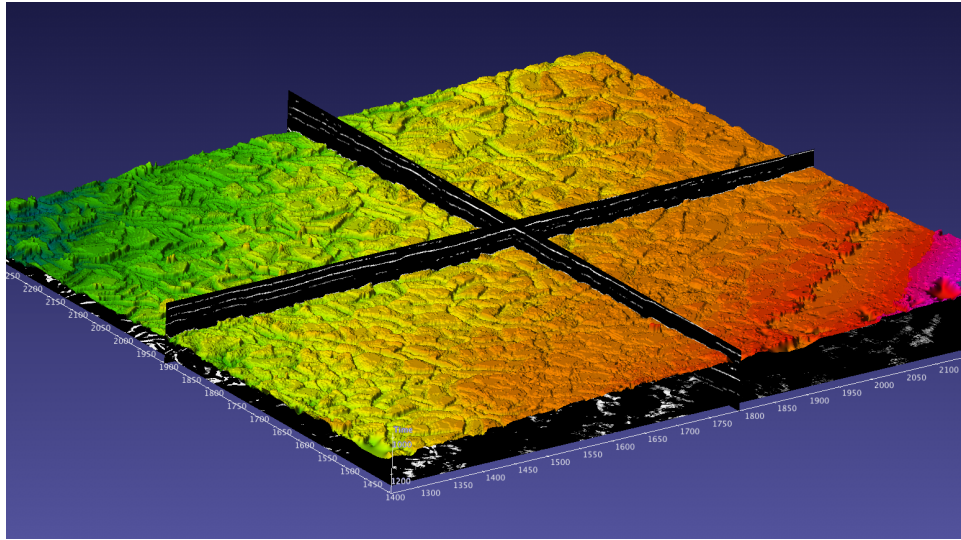
Using Surface Draping, the entire surface was picked in 28 minutes, with manual picks for the basis mesh on every tenth inline. Because of the explicitly-placed



**Figure 4.10:** Part of an inline slice from the Quad 15 volume, shown as raw amplitude data prior to processing (a), and processed to isolate troughs (b). In the top image, four locations have been circled where cycle-skipping occurs when using Insight Earth’s 3D autotracker.

initial boundary, the only instances of cycle-skipping in the final surface were the result of carelessness on the part of the user, and while such errors would require manual cleanup of either the initial picks or the final surface, they were much less pervasive than the problems encountered using the autotracker. Figure 4.11 shows a rendering of the complete draped surface, snapped to troughs.

It should be noted that Surface Draping was performed on a version of the volume that had been filtered to isolate strongly-reflecting troughs (Figure 4.10b). This voxel classification step took approximately half an hour in addition to the 28 minute time measure given above, and used a typical processing workflow that started with the raw amplitude volume. Although preprocessing was not strictly necessary for this dataset, it enabled faster placement of the basis mesh because of the broader area of zero-value voxels above the surface of interest, essentially allowing the interpreter to be more sloppy in regions where there was little ambiguity.



**Figure 4.11:** Orthographic rendering of the draped Quad15 base of salt mesh, colored by depth to illustrate the complexity of the surface.

#### 4.1.2 Medical Data

As stated in Section 2.1, less emphasis has been placed on the evaluation of Surface Wrapping for segmentation of medical volumes than seismic volumes. Medical image data has, however, been used during the development of every version of the software, starting with the earliest prototypes of the technique (Section B.1.1). This has served as a kind of sanity check to prevent the software from becoming overly-specialized: it has always been the intention that Surface Wrapping should be applicable to as broad a range of domains as possible, and the very different characteristics of medical volumes can expose weaknesses that would have otherwise gone unexamined when working exclusively with geological data.

For the purposes of this evaluation, Surface Wrapping was used to segment several features of an MR volume of a patient with brain cancer. No existing segmented surfaces were available against which the results could be compared, but it is still possible to gain some insight into the effectiveness of the current implementation when applied to medical data.

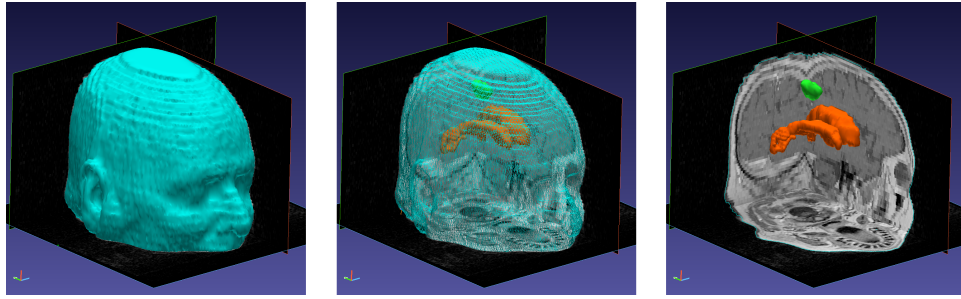


Figure 4.12: Surface Wrapping used to segment various features of an MR volume.

Figure 4.12 shows the final surfaces that were generated for the dermis, corpus callosum, and tumor. While none of the structures were difficult or time-consuming to isolate, requiring between 5 and 10 minutes each, several problems were encountered that will need to be addressed if Surface Wrapping is to be used for medical volume segmentation in general.

The most significant problem is that the Mesh Painter is not well-suited to the definition of complex boundaries that are not predominantly orthogonal to the viewing plane, a limitation that also affects the usability of the tool for interpreting large salt bodies in seismic volumes. When creating the basis mesh for the corpus callosum in particular, it was necessary to work one slice at a time because of the complex topology, a restriction which eliminates much of the performance benefits of Surface Wrapping relative to other semi-automatic approaches. If the user must work slice-by-slice, it seems likely that a method such as Live Wire would provide results that are equivalent, if not better, as a consequence of having an interface that is designed for a 2D mode of operation.

An unexpected problem was the rather poor performance of surface permeability when used with this and other MR and CT scans. In the leftmost image in Figure 4.12, aliasing artifacts are quite noticeable, and small spikes are present in some parts of the surface that cannot be explained at this time. Permeability values ranging

from 0.4 to 0.9 were tested, with a value of 0.7 yielding the best results. It may be significant that seismic volumes tend to have a much higher resolution in Z than the medical volumes that were examined, especially since the worst artifacts are seen in regions of the MR scan where the region of interest varies the most in Z: common vertical to horizontal aspect ratios of voxels in seismic volumes range from 10/1 to 5/2, while the Z to X-Y voxel aspect ratio in the MR volume shown is 1/3. Despite the different spatial characteristics of the data, these findings are puzzling, because surface permeability has never been intentionally optimized for seismic interpretation, and it was expected that the results would be equally good when used with medical volumes.

Finally, a surface registration issue was encountered that prevented a more rigorous analysis of the results: all of the final meshes appear offset by about 1 voxel unit along the sagittal plane relative to the volume. This is most likely a rendering bug in the application rather than a flaw of Surface Wrapping itself, but it will need to be resolved before further testing is performed with medical volumes.

#### **4.1.3 Stanford Bunny**

The Stanford Bunny is a well-known dataset that is often used to demonstrate new computer graphics capabilities. Originally provided as a triangulated mesh consisting of 69,451 triangles, the surface was generated from a range scan of a 7.5 inch tall ornamental terracotta rabbit. In 2000, a CT scan of the original Stanford Bunny was created, and it was felt that this dissertation would be incomplete without taking the opportunity to show Surface Wrapping applied to this dataset.

The volume consists of 361 slices at 512 x 512 resolution, with X and Y voxel scales of 0.337891 mm and a Z scale of 0.5 mm. From examining the volume, it appears that the bunny was resting on the right side of its face when the scan was made, presumably to reduce the surface area that was in contact with the bed of the scanner. The sculpture itself is hollow, with two large holes in the base, and a wall thickness of



**Figure 4.13:** The surface-wrapped Stanford Bunny (left) and its real-life counterpart (right).

approximately 15 mm.

Figure 4.13 shows the mesh created using Surface Wrapping side by side with a photograph of the original sculpture. The pattern of ridges that appears across the surface is the result of the limited resolution of the CT scan. Note that while these aliasing artifacts have been somewhat reduced with the use of a surface permeability value of 0.7, distinctive small-scale features, such as the chip near the top of the bunny’s left ear, have been preserved. Unfortunately, no existing segmentation of this volume has been found against which these results can be directly compared, and the odd angle at which the CT scan was made would make accurate registration of a mesh generated from the original range scan impractically challenging.

Because it is easy to use Surface Wrapping to isolate arbitrary regions of a contiguous body, two additional segmentations of this volume were created: one which captures the exterior surface of the sculpture, and a second which captures only the hollow interior. Figure 4.14 shows the outer mesh as a red wireframe rendering, with the inner mesh rendered as a solid green surface. The meshes are intersected by three slices from the volume, with foreground values (representing the clay) displayed in

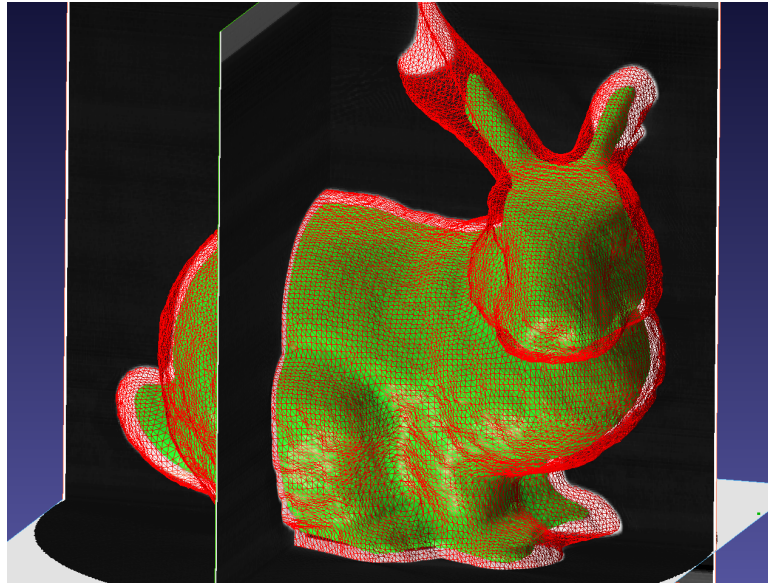


Figure 4.14: Stanford Bunny, interior and exterior surfaces.

white. This is possibly the first time that a semi-automatic segmentation technique has been used to capture both the exterior and interior surfaces of the Stanford Bunny as separate meshes.

## 4.2 Usability

The capability of any tool (software-based or otherwise) to produce good results is arguably irrelevant if that tool is not usable in practical situations. If a tool is difficult to learn how to use, clumsy to operate, or can only perform efficiently when applied to trivial tasks, it will not be readily accepted. As Donald Norman aptly states in *The Design of Everyday Things* [38], “what good is the technology if it is too complex to use?”

This section describes how usability evaluation has been incorporated into the development cycle of Surface Wrapping, and how feedback from users continues to guide ongoing research.



#### **4.2.1 The Geoscience Interpretation Visualization Consortium**

Surface Wrapping has been in active development since May 2004, beginning with a small proof of concept project for the Geoscience Interpretation Visualization Consortium (GIVC) at the BP Center for Visualization, a research organization within the University of Colorado at Boulder that would later separate from the university to form the company TerraSpark Geosciences. The consortium has included representatives from almost all of the major multinational oil and gas corporations, several national and smaller, independent energy companies, and one of the three major seismic interpretation software providers. During the past five and a half years, the objectives of Surface Wrapping have changed greatly—often in response to feedback from members of the GIVC—which has, in turn, significantly impacted the design of the software. A history of major milestones is provided in Appendix B.

From the project's inception, the GIVC has served as a pool of domain experts, initially providing assessment of research results presented at semi-annual consortium meetings, and later (as the program grew into a tool that could be employed in practical situations) as an informal user group. Both individual training sessions and group workshops have been held for consortium members, and feedback is frequently received as Surface Wrapping is used in the field. On several occasions, it has been possible to work on site with some of the GIVC representatives, allowing a much clearer understanding of the interpretation problems faced by this group of users.

#### **4.2.2 Responding to User Feedback**

Feedback received from both GIVC members and other users outside of the consortium has provided the motivation for numerous changes in the software's interface.

One of the most notable examples of this kind of exchange has already been described in Section 3.5.2, which discusses the evolution of the various affordances

that indicate whether or not the paint brush falls within the specified slab of slices. Both onion skinning and the use of separate display and target volumes are features that were designed in response to criticism about the difficulty of defining a boundary in relation to 3D data within the constraints of a limited 2D view. Surface Draping was implemented as a separate Insight Earth process module from Surface Wrapping because users found the original dual mode of operation confusing, despite the similar controls and what was previously assumed to be viewed as identical behavior.

#### 4.2.3 Ease of Learning

The Mesh Painter was designed, in part, to present a familiar appearance to users of existing 2D and 3D segmentation tools. The mesh deformation algorithm was also intended to provide a straightforward conceptual model that would lessen the difficulty of applying Surface Wrapping to target bodies with complex 3D topologies. In theory, these considerations should make the software easier to learn.

Practice seems to bear out this theory to a large extent. During several training sessions with both individuals and groups of up to 12 people, users have consistently grasped the basic concepts of Surface Wrapping in 15 to 30 minutes, and become comfortable enough to use the software independently on non-trivial datasets within 2 hours. The quality of the results achieved continue to improve after initial training, which is not unexpected with a tool that relies so heavily upon human input.

Although the use of the Mesh Painter is not always immediately apparent to novices, the idea of “painting in slabs” is easily demonstrated, after which users tend to regard this approach to creating a 3D surface as fairly straightforward. In general, users also seem to find the behavior of the deformable mesh to be intuitive based on experience with real-world analogs, and are able to make effective use of the process’s Elasticity parameter and the Tighten Mesh button to achieve a better fit to the region of interest. The Surface Permeability parameter is less commonly adjusted, with most

users keeping the default value.

One interesting finding is that although the surface elasticity model currently employed is generally easy for new users to understand and work with, in certain situations, it produces results that are highly counter-intuitive; specifically, the mesh sometimes appears to shrink when the user has indicated that it should expand. This behavior is most commonly encountered when a long, thin structure has been painted, and the results of the surface deformation are being viewed with the widest part of the mesh facing the user—for example, a wide but thin channel seen from an overhead view. This phenomenon is similar to watching the inflation of a balloon that has been laid flat on a table: at first, the edges of the balloon will appear to contract, because they are being pulled inward as the top and bottom of the balloon draw apart. Even though this concept can be explained to new users, many find it difficult to remember or apply, leading to the conclusion that the behavior of the mesh deformation algorithm is not always as intuitive as hoped.

#### **4.2.4 Use In The Field**

Perhaps most indicative of the success of Surface Wrapping is the extent to which it is already being used independently, without pressure from the software's developers. While it has been applied to channel interpretation for academic research, by far the most common use scenario thus far has been the interpretation of large salt bodies. It is with regard to this latter use case that two of the most common criticisms have been leveled against Surface Wrapping, both relating to the Mesh Painter.

The first criticism is that when the Mesh Painter is used to cover large slabs of the volume, it is difficult, if not impossible, to create a smooth boundary that follows the contours of a complex salt body. This leads to poor results in parts of the mesh that are too far away from the surface that is being picked, or where the mesh unintentionally extends into the boundaries of the region of interest. The same issue affects

segmentation of medical volumes, as discussed in Section 4.1.2.

The second criticism is that the painting process itself is too time consuming. Users have frequently suggested that a mechanism such as a seeded region growing algorithm could be used as a starting point for additional manual painting, which may be a useful approach to resolving this problem.

Although these criticisms have been voiced by a number of different users, many of the same individuals have also commented that Surface Wrapping is still a much faster way to segment large salt bodies than any existing approach, even taking into account the time required to manually paint a complex basis mesh. One experienced interpreter, for example, stated that Surface Wrapping “has the capability to be a major step forward in modeling of salt bodies.”

## Chapter 5

### Conclusions

Surface Wrapping shows promise as a fast, simple, accurate, and practical solution to the problem of volume segmentation that fills a gap between previous semi-automated approaches and manual segmentation. By relying on an explicitly-defined initial bounding surface and using a simplified physical deformation model with behavior that can be intuitively understood, the user has a level of control over the final results that is second only to fully manual picking, yet is orders of magnitude faster in common scenarios.

#### 5.1 Contributions

The main difference between Surface Wrapping and other semi-automatic volume segmentation approaches is the precision with which the user can control the final boundary by the placement of the initial boundary. The requirement of manually defining a complex basis mesh, which may seem at first glance to be an excessively time-consuming process, is actually the major advantage of the technique, because it places the responsibility of making those decisions which require human intelligence in the hands of the user, not the computer. This allows Surface Wrapping to be used with complex, noisy, or poor-quality data where more conventional seeded region-growing algorithms fail, but still improves on purely manual interpretation by allowing the software to automatically capture detailed surfaces where possible with-

out requiring precise user input.

The use of a rasterized initial boundary, combined with the Mesh Painter interface, has proven to be a very effective approach for defining complex 3D surfaces in relation to volumetric data. Hands-on experience teaching the software to new users has shown that the underlying concepts are not difficult to understand, with most novice users becoming comfortable enough to work effectively within fifteen minutes to half an hour. As opposed to techniques which rely on more technically elaborate methods (e.g., defining a series of polylines or surface patches that must be stitched together manually), bounded regions of arbitrary complexity can be defined in a single pass, easily accommodating overhangs, disconnected bodies, inclusions, bifurcations, and any other forms that can be represented in a volume. The onion skinning feature overcomes one of the main obstacles to working with 3D data using a 2D display and input device by displaying a contextually relevant view of the target voxels across a range of slices, allowing the user to work in 2D while still maintaining control over the placement of the 3D boundary relative to features in the volume. Overall, the Mesh Painter achieves the goal of facilitating the rapid creation of approximate bounding surfaces while still allowing fine-grained manual control when necessary.

Another strength of Surface Wrapping is the novel approach to elastic surface simulation that is used to fit the basis mesh to the volume data. The mesh deformation algorithm itself is relatively simplistic compared to many existing deformable surface approaches, having more in common with techniques used for real-time cloth animation than either accurate physical simulation of elasticity or deformation that is guided by complex heuristic analysis of the image data.

This simplicity is advantageous in two important ways. First, as a consequence of the low computational complexity, the software's performance is fast enough that experimentation with different parameter settings (e.g., elasticity and surface permeability) is not prohibitively time-consuming for the user. This also encourages an it-

erative approach to defining the basis mesh, in which the user paints a coarse initial mesh, deforms the mesh, examines the results for quality of fit, then refines the painted region in areas that need improvement and repeats the subsequent steps until the desired goal is reached. In this respect, the requirements of the user interface influenced the development of the mesh deformation algorithm as much as the characteristics of the algorithm dictated the interface presented to the user. The second advantage of the algorithm's simplicity is the predictability, from the perspective of the user, of the final results based on the placement of the initial boundary. Because the mesh behaves comparably to an actual physical surface as it deforms, the user can leverage their mental model of elastic materials such as plastic wrap or rubber balloons and expect the mesh to warp and stretch similarly.

Finally, surface permeability has proven to be a remarkably effective mechanism for mitigating the negative effects of some common types of noise, often reducing the need for excessive data filtering and allowing the user to be less precise with the placement of the basis mesh. It also has the unanticipated side benefit of smoothing the mesh in a way that honors the target regions of the volume data, which helps diminish the aliasing artifacts that can result from the rasterized boundaries of the painted mesh.

## 5.2 Future Directions

The most obvious opportunities for additional research lie in addressing the known limitations of Surface Wrapping as it is currently implemented. In particular, a new approach for defining the basis mesh is needed to support the segmentation of complex salt bodies and anatomical structures in medical volumes. Improvements to the mesh deformation algorithm are also needed to address the problems of folding and sagging, and it would also be worthwhile to investigate ways of addressing situations in which the mesh deforms in ways that are counter-intuitive (e.g. wide, flat

regions appearing to shrink instead of expand).

Remeshing—changing the density, connectivity, or distribution of vertices in a mesh without significantly altering the shape or topology of the surface—is an area of research that has a variety of potential applications for Surface Wrapping. The types of remeshing that would be most obviously useful are mesh simplification (for reducing the memory and computational requirements for handling very large meshes) and locally increasing or decreasing vertex density (for improving the quality of fit that can be achieved with the mesh deformation algorithm).

Based on user feedback, two forms of mesh simplification need to be implemented: simplification for display and optimization for file output. In the current implementation, the usability of the tool is greatly impacted by slow rendering when working with very large meshes (e.g. meshes with hundreds of thousands to millions of triangles, depending on the capabilities of the host computer) because interactivity of the Wrapping Steps slider and Tighten Mesh button is essentially lost as a result of rendering delays. Mesh optimization [22] for file output is mainly important for the sake of ensuring that results generated via Surface Wrapping can be used by downstream applications that cannot handle very large, dense meshes by reducing the number of triangles used to represent relatively flat regions of the output mesh. This form of optimization could also be used to boost the performance of the mesh deformation algorithm if used in conjunction with dynamic local remeshing to increase the density of the mesh only where needed.

Localized remeshing could significantly improve the ability of Surface Wrapping to accurately pick complex surfaces. Because the mesh is fit to the region of interest on a per-vertex basis, as the distance between adjacent vertices increases, the resolution decreases. This is a practical concern when regions of a mesh expand during deformation, because this causes a local decrease in vertex density. By locally refining the density according to the distance between adjacent vertices, a more detailed and ac-



curate final surface could be generated without requiring a huge increase in the total vertex count.

In the same way that parts of a mesh can expand too far, potentially causing loss of detail, it is also possible for vertices to become too dense in areas of the mesh that have contracted, which sometimes causes a mesh to appear “bunched up,” a phenomenon which is easily observed in physical shrink wrapping. One possible mechanism for addressing this issue without localized remeshing would be to allow fixed vertices to slide across the surface of target voxels, which could lead to more uniform global vertex distribution.

## Bibliography

- [1] M. Antonelli, B. Lazzarini, and F. Marcelloni. Segmentation and reconstruction of the lung volume in CT images. In SAC '05: Proceedings of the 2005 ACM symposium on Applied computing, pages 255–259, New York, NY, USA, 2005. ACM.
- [2] Apple Inc. GarageBand. <http://www.apple.com/ilife/garageband/>.
- [3] M. Bowers, N. Trinh, G. Tung, J. Crisco, B. Kimia, and B. Fleming. Quantitative MR imaging using “livewire” to measure tibiofemoral articular cartilage thickness. Osteoarthritis and Cartilage, 16(10):1167–1173, October 2008.
- [4] S. Brandel, S. Schneider, M. Perrin, N. Guiard, J.-F. Rainaud, P. Lienhardt, and Y. Bertrand. Automatic building of structured geological models. In SM '04: Proceedings of the ninth ACM symposium on Solid modeling and applications, pages 59–69, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.
- [5] A. R. Brown. Interpretation of Three-Dimensional Seismic Data. SEG/AAPG, 5th edition, 1999.
- [6] G. Celniker, I. Chakravarty, and J. Moorman. Visualization and modeling of geophysical data. In VIS '93: Proceedings of the 4th conference on Visualization '93, pages 362–365, Washington, DC, USA, 1993. IEEE Computer Society.
- [7] T. F. Chan and L. A. Vese. Active contour and segmentation models using geometric PDE's for medical imaging. In R. Malladi, editor, “Geometric Methods in Bio-Medical Image Processing”, Series: Mathematics and Visualization, pages 63–75, Los Angeles, CA, USA, 2002. Springer.
- [8] L. Cohen and I. Cohen. Deformable models for 3D medical images using finite elements & balloons. Computer Vision and Pattern Recognition, 92:592–598, June 1992.
- [9] L. D. Cohen. On active contour models and balloons. CVGIP: Image Understanding, 53(2):211–218, 1991.
- [10] P. Cooke. Poser Tool Box. <http://www.philc.net/PoserToolBox.php>.

- [11] G. Dorn. Modern 3-D seismic interpretation. The Leading Edge, 17:1262–1269, 1998.
- [12] G. Dorn. Method and system for horizon interpretation of seismic surveys using surface draping. United States Patent 5,894,417, April 1999.
- [13] G. Dupuy, B. Jobard, S. Guillon, N. Keskes, and D. Komatitsch. Isosurface extraction and interpretation on very large datasets in geophysics. In SPM '08: Proceedings of the 2008 ACM symposium on Solid and physical modeling, pages 221–229, New York, NY, USA, 2008. ACM.
- [14] D. Emery and K. J. Myers. Sequence Stratigraphy. Blackwell Publishing, Oxford, 1996.
- [15] J. Fan, G. Zeng, M. Body, and M.-S. Hacid. Seeded region growing: an extensive and comparative study. Pattern Recogn. Lett., 26(8):1139–1156, 2005.
- [16] D. Farin, M. Pfeffer, P. de With, and W. Effelsberg. Corridor scissors: a semi-automatic segmentation tool employing minimum-cost circular paths. In ICIP04, pages II: 1177–1180, 2004.
- [17] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. Computer Graphics: Principles and Practice in C. Addison-Wesley Professional, second edition, August 1995.
- [18] M. Gleicher. Image snapping. In SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pages 183–190, New York, NY, USA, 1995. ACM.
- [19] W. S. Hammon. Computer Techniques to Aid the Interpretation of Salt Bodies and Stratigraphy in 3D Seismic Volumes. PhD thesis, University of Colorado at Boulder, 2009.
- [20] M. Harders and G. Székely. New metaphors for interactive 3D volume segmentation. EuroHaptics, pages 129–134, July 2001.
- [21] L. Hermoye, I. Laamari-Azjal, Z. Cao, L. Annet, J. Lerut, B. Dawant, and B. V. Beers. Liver segmentation in living liver transplant donors: Comparison of semi-automatic and manual methods. Radiology, 234:171–178, 2005.
- [22] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Mesh optimization. In SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques, pages 19–26, New York, NY, USA, 1993. ACM.
- [23] R. Huang and K.-L. Ma. RGVis: Region growing based techniques for volume visualization. In PG '03: Proceedings of the 11th Pacific Conference on Computer Graphics and Applications, page 355, Washington, DC, USA, 2003. IEEE Computer Society.
- [24] H. James, G. Schaetzlein, and T. Stark. Geophysical interpretation: From bits and bytes to the big picture. Oilfield Review, 6(3):23–31, March 1996.

- [25] W.-K. Jeong and C.-H. Kim. Direct reconstruction of a displaced subdivision surface from unorganized points. Graph. Models, 64(2):78–93, 2002.
- [26] B. Kadlec. Interactive GPU-based “Visulation” and Structure Analysis of 3-D Implicit Surfaces for Seismic Interpretation. PhD thesis, University of Colorado at Boulder, 2009.
- [27] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. International Journal of Computer Vision, 1(4):321–331, January 1988.
- [28] C. Kirbas and F. Quek. A review of vessel extraction techniques and algorithms. ACM Comput. Surv., 36(2):81–121, 2004.
- [29] L. P. Kobbelt, J. Vorsatz, U. Labsik, and H. Seidel. A shrink wrapping approach to remeshing polygonal surfaces. EUROGRAPHICS ’99, 18(3), 1999.
- [30] Konrad-Zuse-Zentrum für Informationstechnik Berlin, Berlin, Germany. Amira User’s Guide and Reference Manual, 3.1 edition, 2003.
- [31] B. K. Koo, Y. K. Choi, C. W. Chu, J. C. Kim, and B. T. Choi. Shrink-wrapped boundary face algorithm for mesh reconstruction from unorganized points. ETRI, 27(2):235–238, April 2005.
- [32] S. Loncaric, D. Kovacevic, and E. Sorantin. Semi-automatic active contour approach to segmentation of computed tomography volumes. In Proceedings of SPIE Medical Imaging, volume 3979, 2000.
- [33] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. SIGGRAPH Comput. Graph., 21(4):163–169, 1987.
- [34] R. Lu, P. Marziliano, and C. Thng. Liver tumor volume estimation by semi-automatic segmentation method. In Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the, pages 3296–3299, 2005.
- [35] S. Matuszek. Volume rendering of the photographic visible human data set. Technical report, University of Maryland, December 1998.
- [36] E. N. Mortensen and W. A. Barrett. Intelligent scissors for image composition. In SIGGRAPH ’95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pages 191–198, New York, NY, USA, 1995. ACM.
- [37] H. Müller and M. Wehle. Visualization of implicit surfaces using adaptive tetrahedrizations. In DAGSTUHL ’97: Proceedings of the Conference on Scientific Visualization, page 243, Washington, DC, USA, 1997. IEEE Computer Society.
- [38] D. A. Norman. The Design of Everyday Things. Basic Books, 1988.
- [39] L. O’Donnell. Semi-automatic medical image segmentation. Master’s thesis, Massachusetts Institute of Technology, October 2001.

- [40] A. Olowoyeye, M. Tuceryan, and S. Fang. Medical volume segmentation using bank of gabor filters. In SAC '09: Proceedings of the 2009 ACM symposium on Applied Computing, pages 826–829, New York, NY, USA, 2009. ACM.
- [41] X. Papademetris, M. Jackowski, and N. Rajeevan. BioImage Suite User's Manual. Yale School of Medicine, 2008.
- [42] S. Pedersen, T. Randen, L. Sonneland, and O. Steen. Automatic 3D fault interpretation by artificial ants. In 64th Meeting, EAEG Expanded Abstracts, May 2002.
- [43] A. S. Pinto. Shrinkwrap modifier. <http://wiki.blender.org/index.php/User:Jaguarandi/SummerOfCode2008/ShrinkwrapModifier>, 2008.
- [44] S. Qatarnah, M. Noz, S. Hyödynmaa, G. Q. J. Maguire, E. Kramer, and J. Craford. Evaluation of a segmentation procedure to delineate organs for use in construction of a radiation therapy planning atlas. International journal of medical informatics, 69(1):39–55, January 2003.
- [45] J. Raskin. The Humane Interface: New Directions for Designing Interactive Systems. ACM Press, 2000.
- [46] P. K. Sahoo, S. Soltani, A. K. Wong, and Y. C. Chen. A survey of thresholding techniques. Comput. Vision Graph. Image Process., 41(2):233–260, 1988.
- [47] A. Schenk, G. Prause, and H.-O. Peitgen. Efficient semiautomatic segmentation of 3D objects in medical images. MICCAI - Medical Image Computing and Computer-Assisted Intervention, pages 186–195, October 2000.
- [48] J. Smart. wxWidgets. <http://wxwidgets.org/>.
- [49] Smith Micro. Poser. [http://www.smithmicro.com/default.tpl?group=product\\_full&sku=PSRC70DEMR](http://www.smithmicro.com/default.tpl?group=product_full&sku=PSRC70DEMR).
- [50] P. Suetens, P. Fua, and A. J. Hanson. Computational strategies for object recognition. ACM Comput. Surv., 24(1):5–62, 1992.
- [51] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, pages 205–214, New York, NY, USA, 1987. ACM.
- [52] U.S. National Library of Medicine. The visible human project. [http://www.nlm.nih.gov/research/visible/visible\\_human.html](http://www.nlm.nih.gov/research/visible/visible_human.html).
- [53] O. Wirjadi. Survey of 3D image segmentation methods. Technical Report 123, Fraunhofer ITWM, Kaiserslautern, Germany, 2007.
- [54] J. Zhou, W. Xiong, Q. Tian, Y. Qi, J. Liu, W. K. Leow, T. Han, S. Venkatesh, and S.-c. Wang. Semi-automatic segmentation of 3D liver tumors from CT scans using voxel classification and propagational learning. The Midas Journal, 2008.

## Appendix A

### Glossary

This glossary is primarily intended to clarify terminology that may cause confusion because of variations in usage between the medical, geological, and computer sciences.

**Amplitude volume** A time- or depth-migrated seismic volume.

**Attribute** A measurement of some property of seismic data. In the context of this work, attributes are generally used to help isolate the boundary of the region of interest from the surrounding data.

**Autotracker** A type of seeded region growing algorithm that is used to automatically pick seismic events using one or more user-supplied seed picks as input. Autotrackers may be either 2D, restricting automatic picks to the plane of one slice, or 3D, allowing (in principle) automatic interpretation of an entire event.

**Axis labels** In seismic volumes, the X, Y, and Z axes are commonly referred to as inline, crossline, and time or depth, respectively. Whether the X axis is labeled inline and the Y axis crossline or those labels are reversed is somewhat arbitrary, and depends upon the organization of the volume and how the data was acquired, amongst other factors. Whether the Z axis is labeled time or depth depends upon whether the volume is time-migrated or depth-migrated, a distinction which is not addressed in this text.

**Basis mesh** In Surface Wrapping, the mesh that represents the initial, approximate surface that bounds the region of interest in a volume.

**(Seismic) Interpretation** Interpretation is a broadly-defined term in the geological sciences, referring not only to the process of segmentation as described below, but also to any other approach used to analyze seismic data with the goal of constructing a (computational, conceptual, or other) model of the geology in a subsurface area. In this text, interpretation is essentially used interchangeably with the term segmentation.

**Mesh** An unstructured grid. In the context of this work, this term refers specifically to a triangulated polygon mesh unless otherwise stated. Meshes are heavier-weight than single Z-value surfaces because vertex connectivity information must be stored, but they are able to represent surfaces with arbitrarily complex topology.

**Segmentation** Segmentation is most commonly defined as the subdivision of image data into discrete regions, or “segments.” Less commonly (but still correctly), segmentation may also refer to the more general process of isolating any borders of interest in an image, whether or not those borders define a closed boundary. Unless otherwise stated, the term “segmentation” in this text will refer to 3D segmentation of volumetric data.

**Seismic volume** A 3D volume generated via the process of seismic acquisition and migration. See Section 2.1.2.

**Single Z-value surface** A 3D surface in which any Z coordinate value can be uniquely identified by an X-Y coordinate pair. Single Z-value surfaces are commonly used in seismic interpretation, in part because they are easy to represent in software as lightweight regular grids. This simplicity can also be a major disadvantage, since a single Z-value surface cannot represent overhangs, closed bodies, inclusions, or other more complex topological features. Common synonyms include Z-grid, XYZ surface, and 2.5D surface.

**Slice** A 2D plane intersecting a 3D volume onto which the data values at the plane's location are displayed.

**Snapping** Automatically repositioning a pick to the center of a seismic event, typically a peak, trough, or zero-crossing.

**Stratal volume** A seismic volume that has been processed such that the depositional layers have been flattened. In a stratal volume, stratigraphic features, such as channels, can be viewed on a single map-view slice because the effects of dip and faulting have been removed.

**Voxel** The smallest unit of a volume; the 3D equivalent of a pixel.



## **Appendix B**

### **Research And Development History**

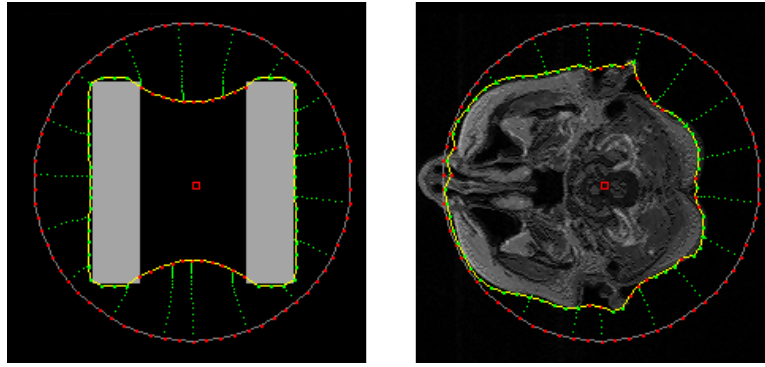
This appendix provides high-level descriptions of the major versions of Surface Wrapping and Surface Draping, focusing on the design goals at each phase, the most significant differences from previous versions, and lessons learned. It is worth noting in particular that there are several instances where requirements of the user interface impacted the evolution of the mesh deformation algorithm, or where characteristics of the algorithm influenced the design of the GUI.

#### **B.1 Surface Wrapping**

##### **B.1.1 Surface Wrapping, Version 1 (2D Prototype)**

The initial research goal was to develop a feasible elastic surface deformation algorithm. It was assumed from the outset that some form of connected mesh (i.e., a mesh for which a list of adjacent vertices can be obtained for any individual vertex) would be used to represent the surface, but no other assumptions were made about the surface structure. In order to simplify the exploration of algorithmic concepts, a 2D prototype was implemented to test the results on individual slices, using a closed loop of connected vertices to represent the bounding surface.

No user interface was developed for this prototype. A fixed-diameter circle was used as the initial boundary, and all parameters were either hard-coded or input as command line arguments. Results were captured as image files, showing both the



**Figure B.1:** Test results from the first 2D Surface Wrapping prototype, showing synthetic data (left) and a slice from an MRI volume (right). The initial circular boundary is drawn as a grey border with vertices drawn as red dots. The final border is drawn in yellow, with fixed vertices drawn in green and non-fixed vertices in red. Dotted green lines between the initial and final borders trace the paths of a subset of the vertices as the surfaces were deformed.

initial and final boundaries superimposed on a linear greyscale image of the target data. Tests were performed using two images as the target data: a slice from an MRI scan of a human head for assessing performance on complex, organic boundaries, and a synthetic image containing two widely-separated rectangular regions for determining the general characteristics of the deformation algorithm.

The deformation algorithm that was developed using this prototype is similar to that which is used in the current implementation of Surface Wrapping, except that vertices are attracted to a center point instead of following a trajectory given by the vertex normal. The approach of simulating elasticity by using a weighted mean of neighboring vertex positions is essentially the same in this 2D implementation, disregarding some minor implementation details. The stopping condition was based on a single threshold value: voxels above the threshold were considered “solid,” while those below the threshold were treated as “empty space.”

As a means of brainstorming and quickly testing algorithmic concepts, the simple 2D prototype was very useful, and since the code was written in anticipation of

being extended to 3D, it was possible to repurpose much of the code for the next version of the software with minimal rewrites. Even when using solid high-level libraries, the many complications of generating 3D graphics can dominate the development process and detract from the primary conceptual work. By simplifying the problem and initially targeting 2D graphics, it was possible to rapidly create a solid foundation for future research.

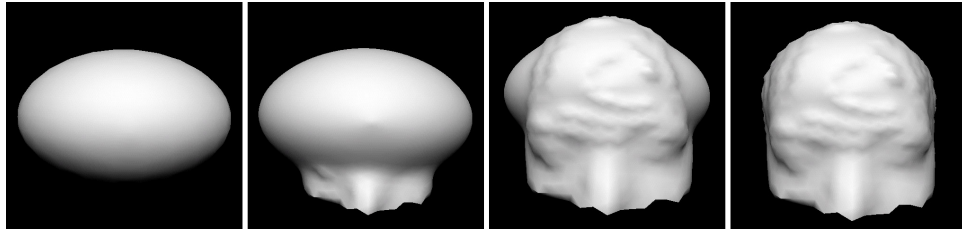
### **B.1.2 Surface Wrapping, Version 2 (3D With Minimal GUI)**

The major goal of the next version of Surface Wrapping was to extend the techniques developed in the first version to 3D, with a secondary goal of testing the effectiveness of the approach for segmenting simple, channel-like features. Once again, the main focus of this effort was the development of the mesh deformation algorithm, and little emphasis was placed on usability; the only graphical interface provided was a simple 3D viewer in which the final surface was displayed.

Apart from being updated to operate on a 3D mesh, the mesh deformation algorithm was essentially unchanged from the previous version. A fixed-diameter, fixed-resolution geodesic sphere served as the basis mesh, and vertices were pulled in toward the center of the sphere.

Two volumes were used for testing: the MRI volume from which a single slice had been used for the 2D tests, and a new synthetic volume that was designed to resemble a simple channel-like geobody (which came to be referred to as the “peanut” volume on account of its bearing a greater resemblance to a peanut than a channel).

The results of the algorithm when applied to the MRI volume were promising, but it was also evident that the software was unable to capture detail in certain areas of the volume (Figure B.2). The lack of detail could be partially explained by the low resolution of the basis mesh, but in some places—the regions between the eyes and the ridge of the nose being the most obvious examples—it was apparent that the mesh was

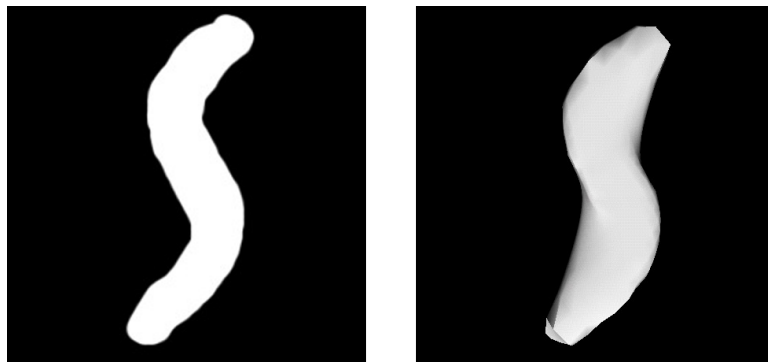


**Figure B.2:** Application of the initial 3D version of the Surface Wrapping algorithm to an MRI volume, shown left to right at iterations 0, 15, 20, and 25. (Images are not shown at equal scale.)

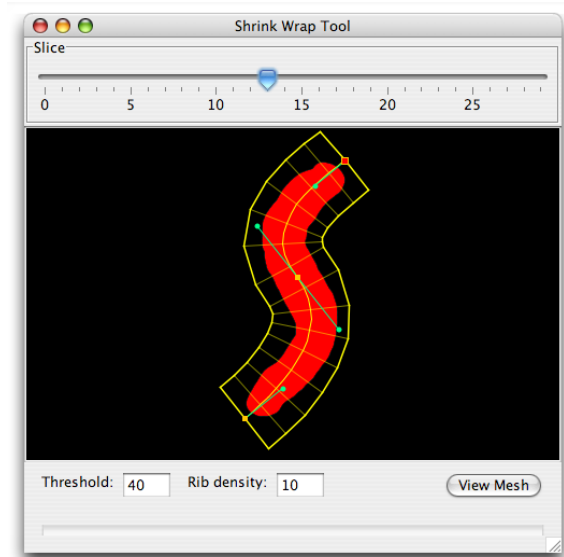
simply not conforming well to the data.

The problems with this approach were made even more plain when the synthetic “peanut” volume was used (Figure B.3). The ends of the peanut are not captured well in the final surface because of the low resolution of the basis mesh, showing a blockiness that is not found in other areas of the surface. Even more significantly, the final mesh stretches across concavities without reaching the actual boundaries of the peanut, regardless of the number of iterations used.

The results of these tests led to two conclusions. First, the approach of moving vertices toward a fixed target point is not generally effective, because it gives the deformation algorithm an artificial bias favoring a regular geometrical structure. Second,



**Figure B.3:** Test of the initial 3D version of Surface Wrapping: a slice from the “peanut” test volume (left), and the resulting 3D mesh (right).



**Figure B.4:** Graphical interface for creating a basis mesh as a flexible tube with a user-definable diameter.

using a basis mesh that does not approximately conform to the shape of the target body can result in a significant loss of detail in the final bounding surface.

### B.1.3 Surface Wrapping, Version 3 (Flexible Tube Approach)

For the third version of Surface Wrapping, the project's goals shifted somewhat to focus more on the problem of channel segmentation, where the software had performed poorly in its previous incarnation. This was also the first version of the program that incorporated a graphical interface for defining the basis mesh: instead of using a simple geometric primitive, a spline-based drawing tool was implemented that allowed the user to create a tube that followed the contours of a channel (Figure B.4).

The tool's main window provided a slider control to allow the user to select the slice upon which the center of the tube would be anchored, and voxels whose values fall within a user-specified threshold were highlighted in red in order to clearly indicate the boundaries of the feature. The tube was defined by clicking on the slice

to add anchor points for the central spline, then dragging control points to adjust the curvature, similar to the spline editing tools common in illustration software such as Adobe Illustrator. A constant-width manifold was drawn surrounding the central spline, indicating the outer bounds of the tube; the diameter of the tube could be changed interactively by clicking and dragging either edge of the manifold.

A separate 3D viewer window provided a button which, when pressed, would apply a fixed number of iterations of the mesh deformation algorithm. One important feature that was initially added as a debugging measure was animation of the algorithm's progress in the 3D viewer. While seemingly inconsequential at the time, this capability had a major long-term impact on the development of both the mesh deformation algorithm and the user interface.

Following analysis of the results from the previous version of Surface Wrapping, the mesh deformation algorithm was modified such that the vertices were now moved along their normal vectors. The original intention was to use the tube's central spline as the "attractor" for the vertices, much like the sphere's center point had been used in the previous version; but, as the earlier tests indicate, this sort of approach tends to cause detail to be lost in regions where the target body does not conform well to the shape defined by the attractor. It was observed that as the mesh deforms and parts of the surface become fixed on the target, the normals of the non-fixed vertices tend to become perpendicular to the surface of the target. The use of vertex normals to specify the vectors of motion for the surface has continued through all subsequent versions of the software, regardless of the method used to define the basis mesh.

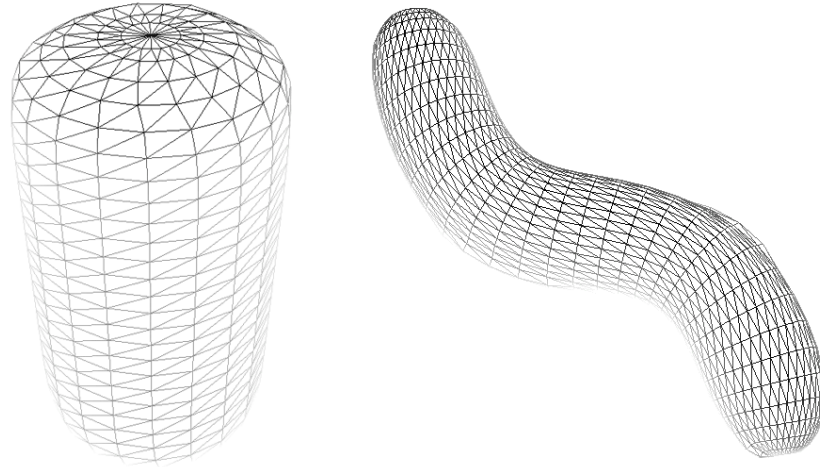
The results obtained for the "peanut" test volume were far superior to those produced with the previous version, though there was still much room for improvement. As shown in Figure B.5, the final surface now conformed much more closely to the bounds of the peanut, but the mesh appeared "bunched up" in some regions—particularly at the ends, where the surface became so pinched as to resemble the



**Figure B.5:** The final bounding surface for the “peanut” test volume, angled to show the pinching of the mesh that resulted from uneven vertex distribution in the basis mesh.

twisted ends of sausage links. The likely source of this problem becomes apparent by examining a wireframe rendering of the basis mesh (Figure B.6): the ends of the tube were closed by connecting the edge vertices to a centralized point, with the result that the two vertices at the ends have far greater numbers of adjacent vertices than found in other regions of the mesh.

A second criticism against the flexible tube approach was raised by GIVC members when the software was demoed at a consortium meeting: it is not capable of handling channel bifurcations, or other objects with more complex shapes. Additionally, the placement of the tube relative to the target channel was too imprecise to effectively segment even simple channels in real seismic volumes, where regions of noise or other undesired features often abut the target geobody, requiring precise, manual interpretive control. Although the user interface for defining the tube was well received, it was clearly a dead end; a new approach was needed for defining the basis mesh.



**Figure B.6:** Two wireframe renderings of basis meshes created using the flexible tube approach. The left image shows one end of a simple cylinder, while the right image shows a tube that follows the contours of the “peanut” in the test volume.

#### **B.1.4 Surface Wrapping, Version 4 (Mesh Painting, Bounded by a Fixed Range of Slices)**

Koo et al. [31] demonstrated a method for reconstructing arbitrarily complex surfaces from point clouds by first partitioning the space using a set of cubes, then using the outer shell of the cubes as an initial mesh that is deformed to fit the target data. While that technique was entirely automated and not intended for application to noisy data, this approach to creating a basis mesh with complex topology served as the inspiration for the interactive “mesh painting” technique developed in the fourth version of Surface Wrapping.

Channel interpretation remained the main goal of the project, but the scope had expanded (based largely on feedback from research consortium members) beyond simple channels to include interpretation of more complex elements of depositional systems, such as fan systems and deltas. The secondary goal of this phase was to improve the results of the mesh deformation algorithm, which seemed to fare the best in regions where the basis mesh had relatively even vertex distribution and connectivity,



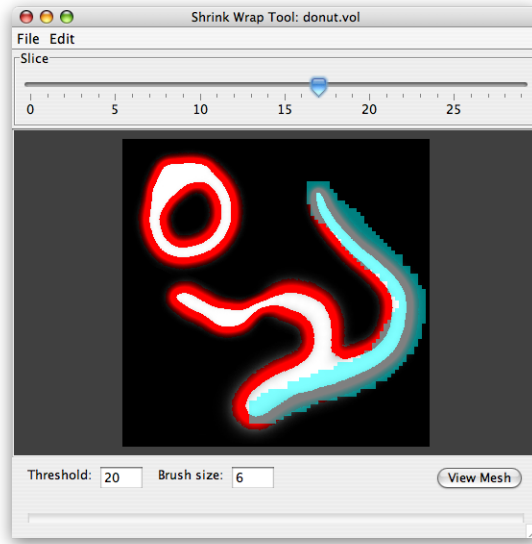


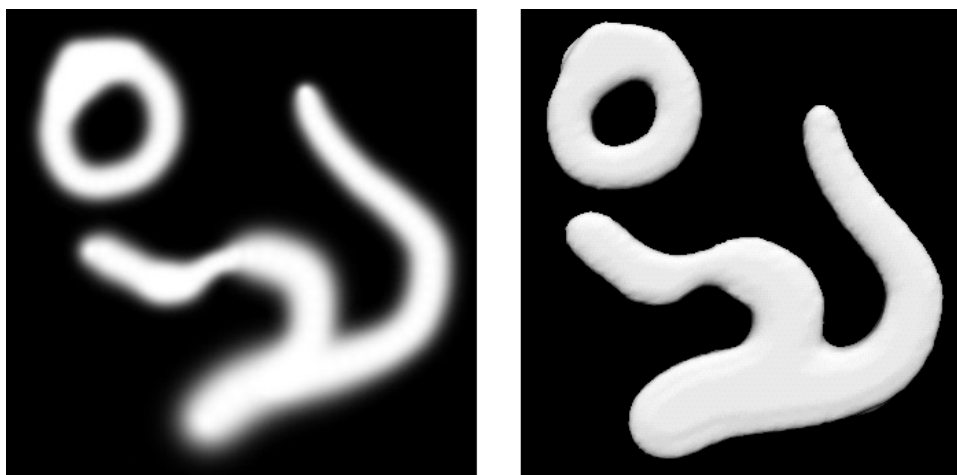
Figure B.7: First implementation of the Mesh Painter tool.

and not surprisingly, where the basis mesh more closely fit the boundary of the target body.

These objectives led to the creation of a new tool that allowed the user to define arbitrary 3D regions using a 2D painting approach that operated on slabs of multiple slices (Figure B.7). As this was essentially a proof-of-concept implementation, the brush covered a hard-coded range of slices centered on the selected slice.

After good results were achieved using the “peanut” test volume, another synthetic volume (“donut”) was created as a slightly more difficult test, incorporating a bifurcated object and a separate, doughnut-shaped object (Figure B.8). Overall, this approach performed far better than expected, but it was also apparent that the painting tool needed to be redesigned before it could be effectively used on real datasets; in particular, centering the slab covered by the paintbrush on the current slice made it impractical, if not impossible, to define an accurate boundary.

The 3D viewer in this version included two buttons: a “Start” button to initiate the deformation process, and a “Stop” button to halt it. When the Start button was



**Figure B.8:** A slice from the “donut” test volume (left), and the final bounding mesh (right).

pressed, up to 10 iterations of the algorithm would be applied, with the mesh updating in the viewer at the end of each iteration. In this way, the user could gauge when the best global fit had been achieved, and stop the process early if, for example, a region of the mesh was being pulled too far into a concavity.

#### **B.1.5 Surface Wrapping, Version 5 (C++ Port, Range Slider Introduced)**

All previous versions of Surface Wrapping had been implemented as stand-alone Java applications. To evaluate its effectiveness when applied to non-synthetic data—seismic volumes, in particular—it was important to be able to access, at the source code level, tools for loading, viewing, processing, and comparing data in a variety of different formats. At this point, it was decided that the best way to proceed would be to integrate Surface Wrapping with the Insight Earth application framework (Section C.1), which was also under development at the time at TerraSpark Geosciences. This necessitated porting the existing code to C++, which required rewriting all of the rendering- and user interface-specific code, but also offered many opportunities for optimization or architecture-level changes that could enhance the usability of the tool.

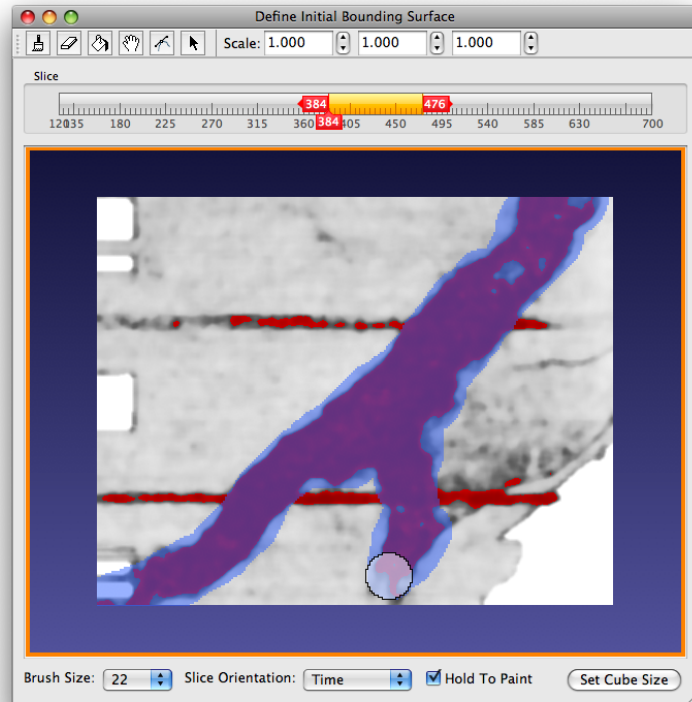
Early optimizations to the mesh deformation algorithm provided a ten-fold speedup, which paved the way for rethinking the animation feature introduced earlier. In the previous version, performance was slow enough that updating the 3D model after each iteration yielded smooth animation as the algorithm progressed for the rather small test meshes. Now, that animation proceeded too quickly to give useful feedback.

Rather than adding a timed delay between iterations to slow the animation, a new approach was chosen: the vertex positions at each iteration would be cached, allowing the user to scrub back and forth across different states of the mesh in real-time. While this allowed the user to control the pace of the animation, the main advantage was that it was now possible to back up to an earlier state of the mesh if the deformation had gone too far. The user-specifiable elasticity parameter was also added to the GUI, along with the Tighten Mesh button (Section C.5.7), both of which made it easier to achieve a good global fit of the surface.

The other major change in this version of Surface Wrapping was the addition of the range slider widget for bounding the slab of slices covered by the 3D paintbrush, explained in detail in Section C.3.1.2. This feature, in conjunction with a number of other minor improvements (e.g., double thresholding, a round paintbrush, undo/redo capability, interactive zoom, and independent axis scales), made it practical to work with real-world datasets. An early version of this interface is shown in Figure B.9.

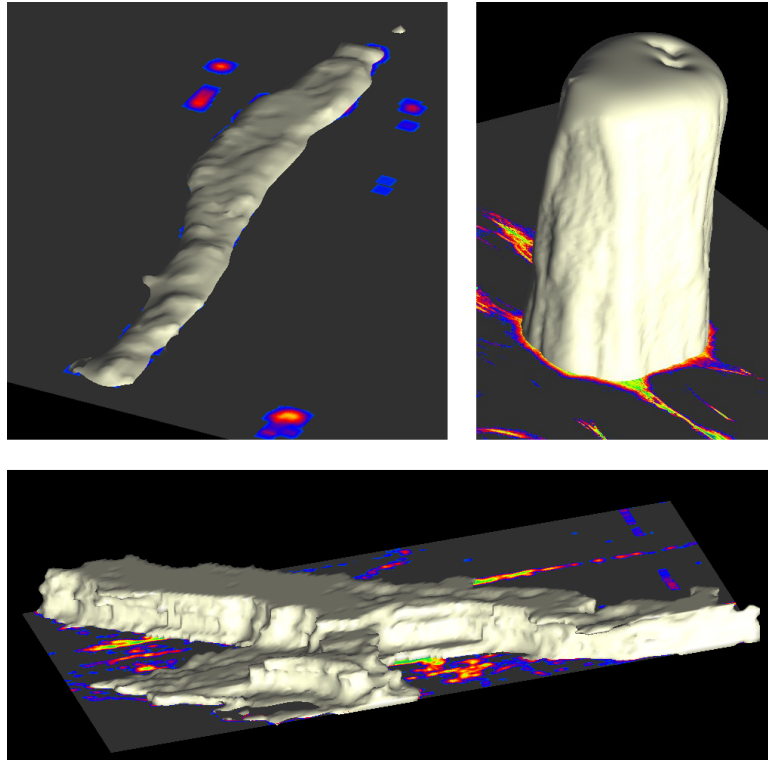
After finding promising results for interpreting simple channels, interpretation of a broader range of geobodies was attempted. Since the painting interface was mainly intended for segmenting bodies with boundaries that vary greatly in X and Y but not in Z (e.g., channels in a stratal volume), it should not be surprising that the most success was achieved for geobodies with predominantly straight vertical borders, such as canyons and salt diapirs. Figure B.10 shows the final surfaces from a few of these early tests.

It is worth noting that it was not until seeing the results from this version of Sur-



**Figure B.9:** An early version of the Mesh Painter window after Surface Wrapping had been incorporated into Insight Earth.

face Wrapping, in late 2006 (more than two years since the project's inception), that members of the GIVC research consortium began to show more than passing curiosity of this effort. Although the technology had been applied to non-synthetic data in the past, all such work had been performed with medical volumes, and understandably, showing Surface Wrapping applied to seismic volumes generated much greater interest. There was also more discussion of potential future applications of Surface Wrapping; this was the point at which consortium members expressed interest in using Surface Wrapping to interpret larger and more complex salt bodies, a task for which the painting interface had never been intended to be used.



**Figure B.10:** Results from the first Insight Earth-based implementation of Surface Wrapping: a small channel (top left), part of a salt diapir (top right), and a canyon (bottom).

## B.2 Surface Draping

### B.2.1 Surface Draping, Version 1 (Early Prototype)

As discussed in Section 2.2, Surface Draping was originally developed by Geoffrey Dorn in the late 1990s [12]. The original implementation was only intended as a proof of concept, and as such, its user interface was quite limited. The initial surface was picked by single-clicking a series of points on a slice, and linear interpolation was used to connect the manual picks. This process was repeated for every slice in the volume to create a Z-grid to be draped onto the horizon of interest. The surface deformation algorithm was also comparatively simple: each point in the surface was independently shifted downward until it reached a peak (or trough) in the amplitude

volume.

While this prototype demonstrated the potential of Surface Draping as an effective means of interpreting complex, highly-faulted surfaces, a more efficient interface for defining the initial Z-grid would be required to make the technology useable in practice. Improvements to the deformation algorithm, such as treating the surface as a connected, elastic sheet, were also considered.

### **B.2.2 Surface Draping, Version 2 (Bezier Patch-Based Surfaces)**

The decision to revive Surface Draping as a tool distinct from Surface Wrapping came in late 2006 as the result of an exploratory project outside of the GIVC. The project in question involved the interpretation of large salt bodies, and having experienced some of the limitations of using the Mesh Painter to segment bodies with borders that vary greatly on all three axes, an effort was made to design an interface that would allow the creation of a basis mesh that smoothly follows the contours of complex, undulating geological features while reducing the amount of manual input required. Because it was felt that the new mesh deformation functionality—such as simulated mesh elasticity (Section 3.6) and surface permeability (Section 3.6.1)—that had been developed for the segmentation of bounded objects would be equally applicable to a “draping” mode of operation, no major algorithmic changes were implemented; therefore, the most significant research topic related to Surface Draping has been the means by which the user defines the basis mesh in relation to the volume data.

Two different approaches have been investigated for creating a drapable mesh. Both methods define the surface using splines, but differ in that the first approach relies on direct manipulation of the parametric curves, while the second fits a curved surface to an unordered set of points.

The main goal of the first interface was to allow the interpreter to create the

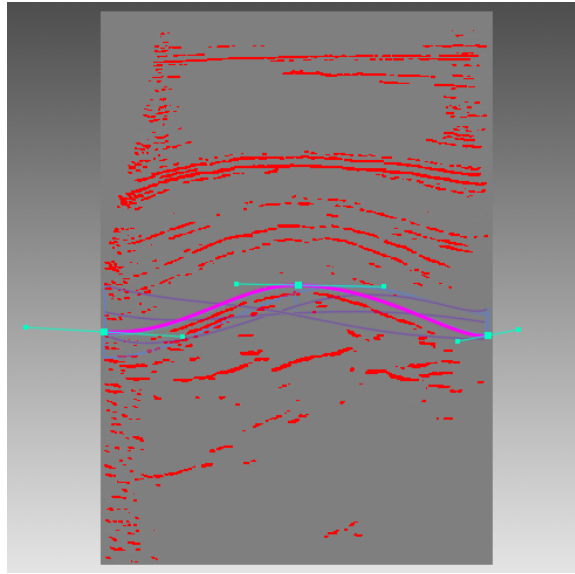


Figure B.11: Defining a simple contoured 3D surface using spline patches.

drapable basis mesh very rapidly, with minimal input required to create a complex, smoothly-contoured surface. This was accomplished using Bezier spline-based surface patches, which are directly defined by the user via spline drawing tools similar to those found in many CAD and illustration software applications (Figure B.11).

To construct the full basis mesh, a 2D spline is defined first on a single slice of the volume by clicking on the image to add anchor points, then manipulating the associated control points to change the curvature of the spline. After the spline has been defined above (or below) the entire horizon on the current slice, the user advances to the next slice in the volume where the contour of the horizon has changed significantly, then duplicates the previously-defined spline at the new slice position, which automatically creates surface patches filling the space between the two splines. After adjusting the positions of the anchor and control points of the duplicate spline, this procedure is repeated until the entire horizon is covered. By successive iterations of extruding, then modifying the contour of the extruded spline, a surface can be built up very rapidly.

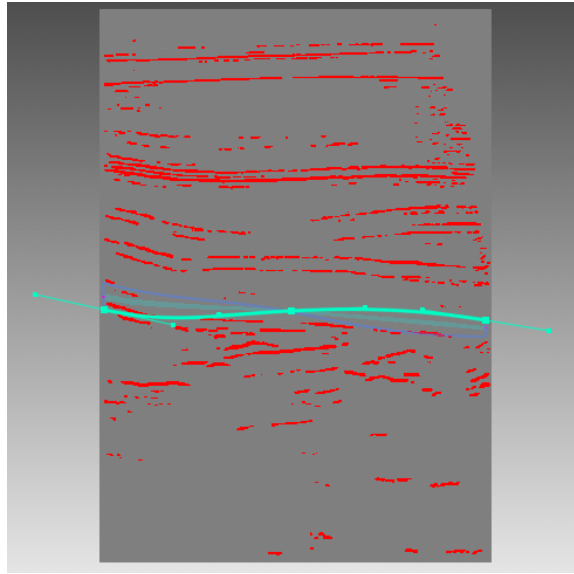


Figure B.12: Highlighting the intersection of the spline surface and the current slice.

Some facilities were included to make it easier for the user to determine the relationship between the 3D surface and the volume data. As the user scrubs through the slices in a particular orientation, the region in which the surface interfaces with the current slice is displayed as a light green highlight, allowing the user to verify that the surface falls on the appropriate portion of the volume in regions between the slices where 2D splines were explicitly defined (Figure B.12). When the current slice falls directly on one of the explicitly-defined splines, the curve shows a bright purple highlight.

This interface was not fully completed, but had been fleshed out sufficiently to discover its main strengths and weaknesses. The major benefit is that it allows some complex surfaces to be built very quickly. However, defining a grid of spline patches in this manner leads to difficulties in situations where it would be preferable to perform detailed manual interpretation in some areas, but revert to a coarser level of detail in other parts of the surface: directly binding the spline patches together in the typical manner [17] requires the interpreter to use the same resolution throughout the entire



surface. Another arguable disadvantage is that because most current seismic interpretation software does not offer spline-based drawing tools, this form of interaction will be unfamiliar to many users, requiring increased training and possible acceptance difficulties.

### **B.2.3 Surface Draping, Version 3 (Point Set Infill)**

The second, and most current, strategy for creating a basis mesh for Surface Draping is detailed in Section 3.5.3. The decision to abandon the previous interface was largely motivated by the need for a greater level of detailed control than the spline patch system seemed likely to allow without fundamental architectural changes. A future hybrid of the two approaches has been considered, in which large, smooth regions could be modeled using connected patches, with arbitrary interpreted lines added on an as-needed basis where more precision is required.

## **Appendix C**

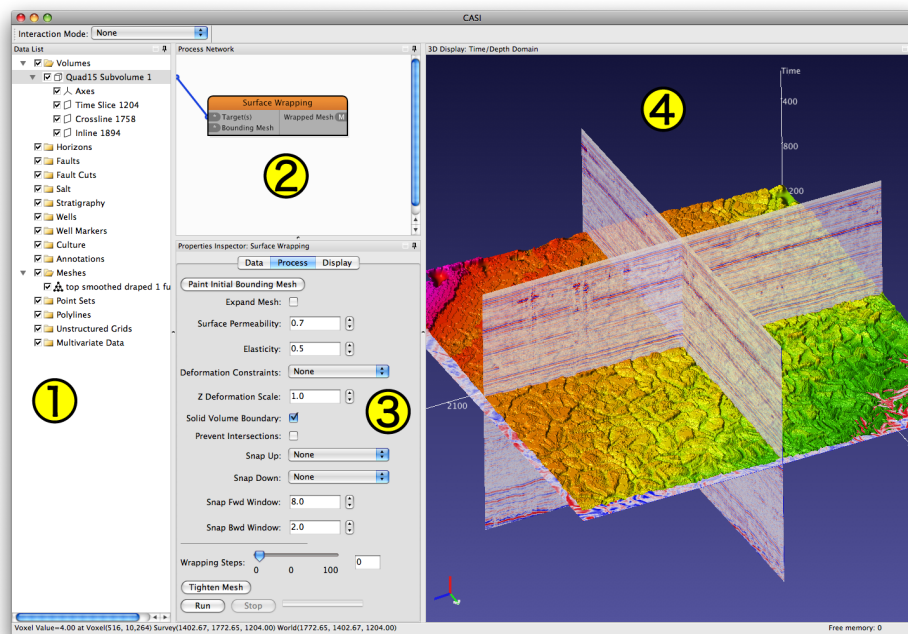
### **User Interface Implementation**

This appendix describes the user interface for the current implementation of Surface Wrapping, starting with a brief overview of Insight Earth, the application framework within which the tool has been developed. The Surface Wrapping and Surface Draping process modules are then described, followed by detailed explanation of the user-facing aspects of the software that are common to both processes.

#### **C.1 The Insight Earth User Interface**

The Insight Earth application framework provides the basic infrastructure needed for 3D interpretation, and allows interactive processes to be added via a plugin architecture. Process modules are shown to the user as nodes in a Process Network (Figure C.1, pane 2), with the process name at the top of the node, data input ports arranged on the left and output ports on the right. A data object (such as a volume or a mesh) can be connected to an input port by dragging its icon from the Data List (Figure C.1, pane 1) or by directly linking the output of one process to the input of another. When a process node is clicked, its modifiable parameters are listed in the Properties Inspector pane (Figure C.1, pane 3). Data objects are shown in the 3D Display (Figure C.1, pane 4), which allows both navigation and various interaction modes for directly editing data objects.

All Insight Earth processes, including Surface Wrapping, are used in approxi-



**Figure C.1:** The main window of Insight Earth, showing the following panes: 1) Data List, 2) Process Network, 3) Properties Inspector, and 4) 3D Display.

mately the same way. A process is instantiated by selecting its name from a global menu, which adds the corresponding node to the Process Network. The user specifies the data object(s) to which the process will be applied by making connections to one or more of the node's input ports, and various process parameters can then be adjusted. Normally these are simply checkboxes, number entry fields, or popup menus that provide control over how a process is run, but process parameters can also be used in more complex ways (as will be discussed at several later points in this appendix). Once the parameters have been set as desired, the process is applied to the input data by clicking the Run button at the bottom of the Properties Inspector. If the process completed successfully, the resulting output data are shown in the 3D Display and added to the Data List.

Data objects have various user-modifiable display options, which are exposed via the Properties Inspector when the user selects an item in the Data List. Typical

display options include, for example, the color map and histogram clipping range for volumes, and the color and specular highlighting for meshes. Display options specifically related to the use of Surface Wrapping are explained in Section C.6.

## C.2 The Surface Wrapping and Surface Draping Processes

Surface Wrapping was initially implemented as a single process module, but some users expressed confusion that this one module encompassed both the “wrapping” behavior (i.e. expanding or contracting a completely closed basis mesh) and the “draping” behavior (i.e. raising or lowering a laterally-extensive and roughly planar basis mesh). Based on this feedback, a separate Surface Draping process module was created, which uses different terminology for some of the IO ports and process parameters, and also provides different default values for some parameters, but otherwise exactly duplicates the functionality of the Surface Wrapping process. This appendix will therefore treat Surface Wrapping and Surface Draping as two distinct entities within Insight Earth, unless otherwise specified.

## C.3 Surface Wrapping Process

This section describes aspects of the user interface for Surface Wrapping that differ from those implemented in the Surface Draping process.



Figure C.2: The process node for Surface Wrapping.

The Surface Wrapping process node (Figure C.2) provides two input ports. The first input, “Target(s),” must be connected to a volume and/or point set to be used as

the target(s) of the mesh deformation algorithm. The optional second input, “Bounding Mesh,” can be used if a suitable basis mesh already exists. Typically, however, the basis mesh for Surface Wrapping will be created within the Mesh Painter tool, described in Section C.3.1. The “Wrapped Mesh” output port provides the final bounding mesh that is created by running and interacting with this process.

<b>Name</b>	<b>Control</b>	<b>Description</b>
Paint Initial Bounding Mesh	button	Opens the Mesh Painter window, where an initial bounding mesh can be created.
Expand Mesh	checkbox	When checked, causes the mesh to expand instead of contract. The default value is Off.
Surface Permeability	number field	Data-aware surface smoothing. Possible values range from 0.0 to 1.0. The default value is 0.7.
Elasticity	number field	Controls the elasticity of the bounding mesh. Possible values range from 0.0 (more stiff) to 1.0 (more flexible). The default value is 0.5.

Table C.1: Surface Wrapping process parameters.

Name	Control	Description
Deformation Constraints	popup menu	<p>Restricts how the mesh will deform relative to the orientation of the input volume.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> <li>• None</li> <li>• No Time Deformation</li> <li>• No Inline and Crossline Deformation</li> </ul> <p>The default value is None.</p>
Z Deformation Scale	number field	A multiplier that specifies how rapidly the mesh should deform along the Z axis relative to the X and Y axes. The default value is 1.0.
Solid Volume Boundary	checkbox	When checked, the sides of the target volume are treated as target voxels, preventing the mesh from expanding beyond the volume bounds. The default value is On.
Prevent Intersections	checkbox	When checked, stops the bounding mesh from passing through itself. The default value is Off.

Table C.1: Surface Wrapping process parameters.

Name	Control	Description
Snap Up	popup menu	<p data-bbox="824 268 1398 485">Specifies the snapping mode for vertices that have an upward-pointing normal when they become fixed on the target. Possible values are:</p> <ul data-bbox="899 548 1019 751" style="list-style-type: none"> <li data-bbox="899 548 992 575">• None</li> <li data-bbox="899 632 987 659">• Peak</li> <li data-bbox="899 716 1019 743">• Trough</li> </ul> <p data-bbox="824 810 1159 835">The default value is None.</p>
Snap Down	popup menu	<p data-bbox="824 873 1398 1089">Specifies the snapping mode for vertices that have a downward-pointing normal when they become fixed on the target. Possible values are:</p> <ul data-bbox="899 1152 1019 1356" style="list-style-type: none"> <li data-bbox="899 1152 992 1180">• None</li> <li data-bbox="899 1236 987 1264">• Peak</li> <li data-bbox="899 1320 1019 1348">• Trough</li> </ul> <p data-bbox="824 1415 1159 1440">The default value is None.</p>
Snap Fwd Window	number field	<p data-bbox="824 1478 1333 1694">Specifies the distance (in voxel units) to search forward from a vertex's location when a snapping mode is enabled. The default value is 8.0.</p>

Table C.1: Surface Wrapping process parameters.

Name	Control	Description
Snap Bwd Window	number field	Specifies the distance (in voxel units) to search backward from a vertex's location when a snapping mode is enabled. The default value is 2.0.
Wrapping Steps	slider	After pre-processing has completed, this slider is used to interactively fit the bounding mesh to the volume data. This control is disabled until the Surface Wrapping process has been run. The default value is 0.
Tighten Mesh	button	Clicking this button pulls the mesh slightly more taut. It can be clicked repeatedly to achieve better results. This button is disabled until the Surface Wrapping process has been run.

Table C.1: Surface Wrapping process parameters.

### C.3.1 Mesh Painter

In most cases, the user will create the basis mesh using the Mesh Painter tool, which is opened by clicking the “Paint Initial Bounding Mesh” button at the top of the process parameters list.

The main components of this window are shown in Figure C.3.



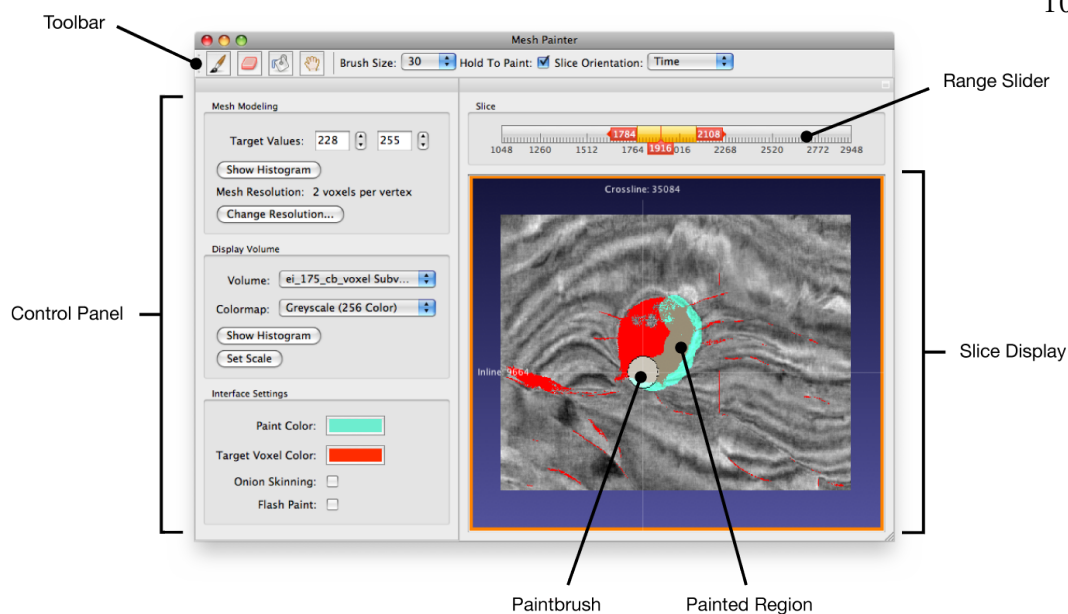


Figure C.3: Major components of the Mesh Painter window.

### C.3.1.1 Setting the Target Voxel Range

When the Mesh Painter is first opened, a separate floating palette is also displayed that shows the histogram of the target volume (Figure C.4), enabling the user to specify the target voxel range. When the target voxel range is modified, the voxels that are highlighted in the Slice Display update correspondingly.

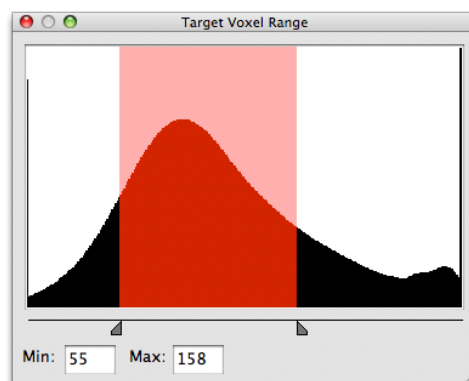


Figure C.4: Setting the target voxel range using the histogram palette.

### C.3.1.2 Slice Display

The Slice Display shows an inline, crossline, or time/depth view of the volume data, with target voxels highlighted in the user's choice of color (with bright red being the default). The basis mesh is created by using a 3D paintbrush within the Slice Display to approximately bound target voxels belonging to the geobody or horizon of interest.

#### Range Slider

The range slider (Figure C.5) is a specialized control used in the process of painting the basis mesh. It has two functions:

- (1) Setting the slice that is currently shown in the slice display.
- (2) Selecting the range of slices over which the 3D paintbrush operates.

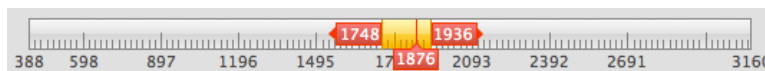
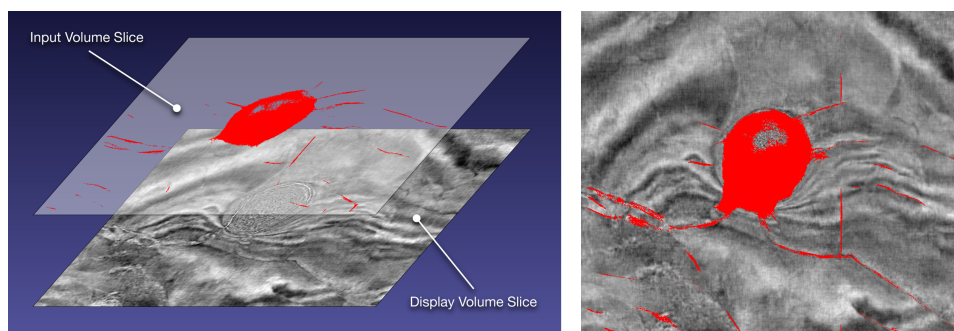


Figure C.5: The range slider control.

The current slice is set by dragging the red handle at the bottom to a different position; the exact slice number is displayed in the handle. The left and right arrow keys can also be used to change the slice number incrementally.

The yellow region in the slider between the left- and right-facing red handles indicates the range of slices affected by the 3D brush. This range is inclusive: the brush will paint on all slices within the range, plus the two slices indicated in the handles. As with the slice handle, the range handles can be moved incrementally by single-clicking a handle, then using the left and right arrow keys.

When the current slice falls within the selected range, the background of the Slice Display changes from grey to blue and an orange border appears, indicating that the paint brush can be used.



**Figure C.6:** Showing different display and target volumes in the Mesh Painter. The highlighted voxels from the target volume slice are overlaid on top of the display volume slice.

### Display Volume (vs. Target Volume)

By default, the slice that is shown in the Mesh Painter window is taken from the target volume. However, it is often important to be able to see the original, unprocessed image data (or a differently-processed version of the volume) while setting the target voxel range or painting the basis mesh.

To facilitate this, the user can choose a display volume that is different from the target volume. When a different display volume is used, the highlighted voxels from the current slice in the target volume are overlaid on top of the same slice in the display volume (Figure C.6).

The display volume can be changed via the Volume popup menu in the Display Volume section of the control panel (Section C.3.3). In the current implementation, the display volume must have the same dimensions as the target volume.

By default, the display volume uses the same colormap as that which has been assigned to the target volume in the main 3D Display, but a different colormap can be selected using the Colormap popup menu. The histogram clipping range can also be changed by clicking the Show Histogram button in the Display Volume section of the control panel, which opens a histogram palette similar to that used elsewhere in Insight Earth (Figure C.7).

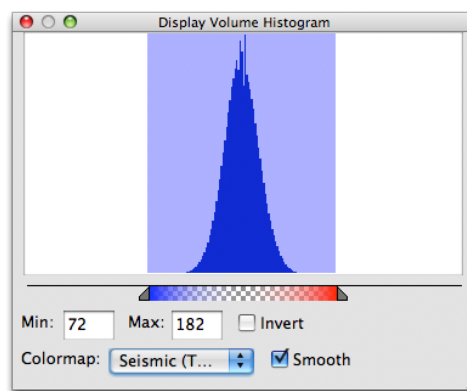


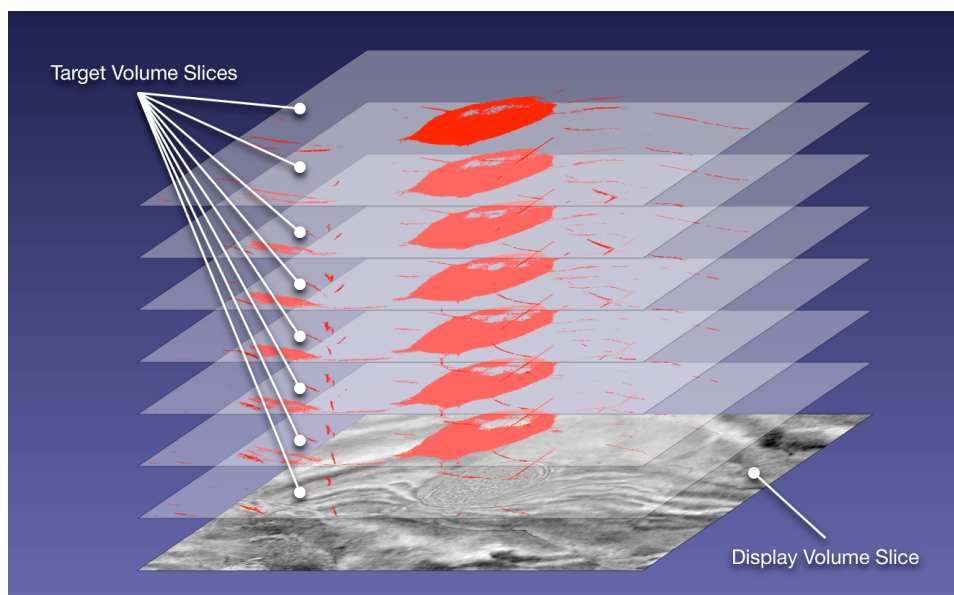
Figure C.7: The histogram palette for the display volume.

### Onion Skinning

When working with a large range of selected slices, it can be difficult to know where the outer boundary of the mesh should be painted, since it is not possible to view multiple slices simultaneously. This problem can be eliminated in many cases by enabling the *onion skinning*<sup>1</sup> option in the control panel. When onion skinning is enabled, a composite of all the target voxels in the selected range of slices is overlaid on top of the current slice from the display volume, making it easier to determine the maximum extent of object of interest.

Enabling onion skinning can make the software slower to respond when changing the selected range of slices, because the composite image needs to be recalculated to show the target voxels within this range. This performance degradation will vary based on the number of slices selected.

<sup>1</sup> The term “onion skinning” originally derives from hand-drawn cel animation. To create a motion sequence—a bouncing ball, for example—the animator would sketch individual frames on translucent onionskin paper. These frames can then be stacked up on top of each other over a light box, which makes it easier to see where the object should be placed in the next frame in the sequence relative to its position in the previous frames.



**Figure C.8:** Onion skinning shows all the target voxels in the selected range of slices layered on top of the current slice from the display volume.

### C.3.2 Toolbar

The toolbar (Figure C.9) at the top of the Mesh Painter window contains the painting tools plus other frequently-used controls.

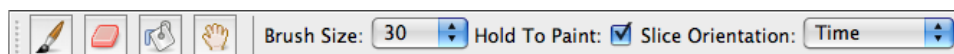



Figure C.9: The toolbar of the Mesh Painter window.

Each of these tools and controls is described below.

Tool	Keyboard Shortcuts
 Brush	<ul style="list-style-type: none"> <li>• Press B to switch to the brush tool.</li> <li>• Hold Shift to temporarily switch to the eraser while the brush tool is active.</li> <li>• Press [ (left bracket) to make the brush smaller.</li> <li>• Press ] (right bracket) to make the brush larger.</li> </ul>

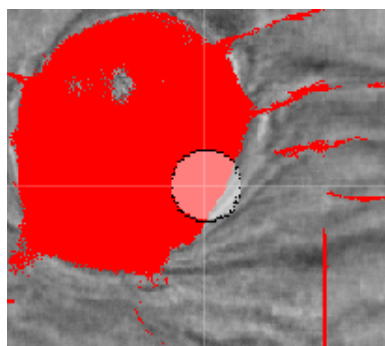




Figure C.10: The brush cursor.

The brush is used to create the initial bounding mesh. When the brush tool is active, the cursor changes to a white translucent circle while inside the slice display (Figure C.10).

Tool	Keyboard Shortcuts
 Eraser	<ul style="list-style-type: none"> <li>• Press E to switch to the eraser tool.</li> <li>• Hold Shift to temporarily switch to the brush while the eraser tool is active.</li> <li>• Press [ (left bracket) to make the eraser smaller.</li> <li>• Press ] (right bracket) to make the eraser larger.</li> </ul>

The eraser is used to delete parts of the initial bounding mesh. When the eraser is active, the cursor changes to a white translucent circle with an X through it (Figure C.11).

Tool	Keyboard Shortcuts
 Flood fill	<ul style="list-style-type: none"> <li>• Press F to switch to the flood fill tool.</li> </ul>

The flood fill tool allows closed regions of the initial bounding mesh to be filled in with a single click. When the flood fill tool is active, the cursor changes to a paint bucket. Like the paint brush and eraser, the flood fill tool operates in 3D, bounded by the slices selected in the range slider. This allows the user to paint the boundaries

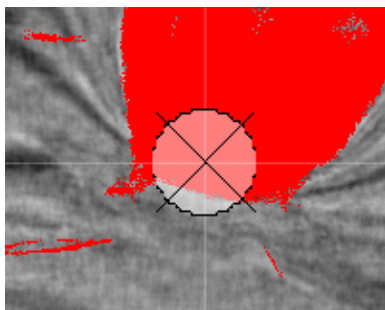



Figure C.11: The eraser cursor.

an enclosed region using the brush, then rapidly infill that region with a single click (Figure C.12), behaving similarly to flood fill tools in conventional 2D graphics applications.

Tool	Keyboard Shortcuts
 Pan	<ul style="list-style-type: none"> <li>• When any tool is active, hold down the Space key to temporarily switch to the pan tool.</li> </ul>

The pan tool allows the view to be panned by clicking and dragging within the slice display. When the pan tool is active, the cursor changes to a hand.

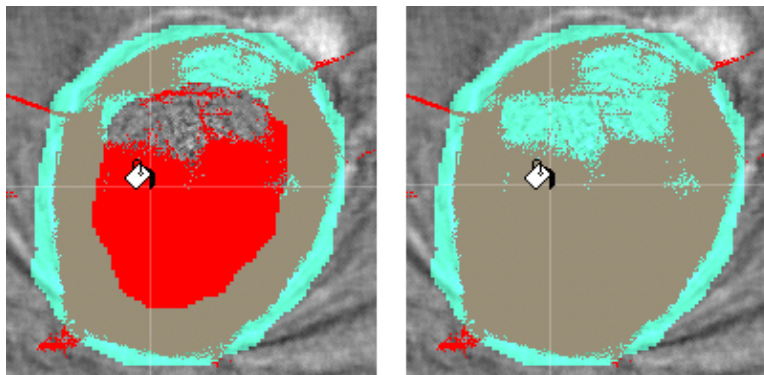


Figure C.12: Using the Flood Fill tool to fill a closed painted region.

<b>Tool</b>	<b>Keyboard Shortcuts</b>
Brush Size	<ul style="list-style-type: none"> <li>• Press [ (left bracket) to make the brush or eraser smaller.</li> <li>• Press ] (right bracket) to make the brush or eraser larger.</li> </ul>

This menu shows the current size of the paint brush (or eraser), and allows the selection of a different size.

<b>Tool</b>	<b>Keyboard Shortcuts</b>
Hold To Paint	none

When this checkbox is turned on, the brush will only paint while the left mouse button is held down (the same way that painting tools work in programs like MS Paint and Adobe Photoshop). When it is turned off, a single click of the mouse will start painting, and a second click will stop painting.

<b>Tool</b>	<b>Keyboard Shortcuts</b>
Slice Orientation	none

This menu allows the user to choose between inline, crossline, and time/depth orientations of the displayed slice.

### **C.3.3 Control Panel**

The Control Panel (Figure C.13) on the left side of the Mesh Painter window provides settings that are typically changed only once, or infrequently, while constructing a basis mesh. It is split into the following three sections: Mesh Modeling, Display Volume, and Appearance.

#### **C.3.3.1 Mesh Modeling (Control Panel Section)**

The Mesh Modeling section of the control panel contains basic settings for establishing the target voxel value range and the resolution of the mesh.



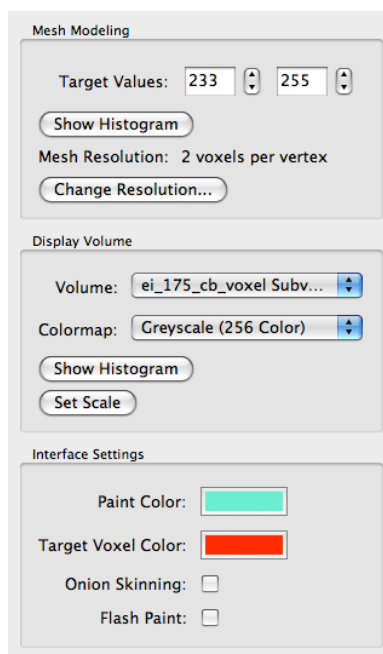


Figure C.13: The Control Panel for the Mesh Painter.

The range of target voxel values can be set either by entering values directly in the Target Values text boxes, or by clicking the Show Histogram button and changing the clipping range in the Target Voxel Range histogram (Figure C.14). This palette is also shown when the Mesh Painter window is first opened.

Below the Show Histogram button, the resolution of the initial bounding mesh is displayed in terms of the voxel unit spacing between neighboring vertices (i.e. voxels

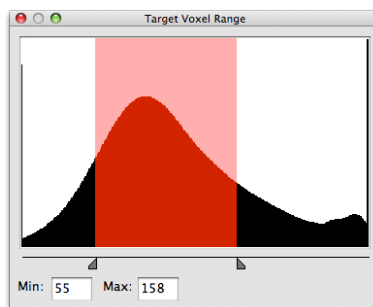


Figure C.14: The Target Voxel Range histogram.

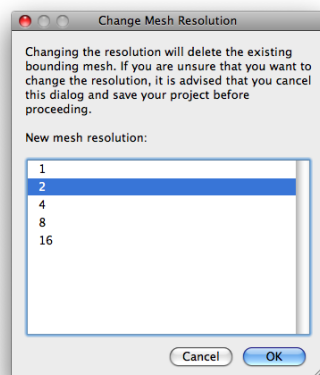


Figure C.15: The Change Mesh Resolution dialog.

per vertex). A different resolution can be set by clicking the Change Resolution... button and selecting a new resolution from the list in the Change Mesh Resolution dialog (Figure C.15). In most situations, this should only be done before starting to paint, because setting a new mesh resolution will erase any existing painted regions.

### C.3.3.2 Display Volume (Control Panel Section)

The Display Volume section of the control panel contains settings related to the presentation of volume data in the slice display. (See Section C.3.1.2 for an explanation of the display volume.)

The **Volume** popup menu allows the user to change the current display volume by selecting from the volumes that are currently loaded in the Data List. By default, the target volume is also used as the display volume, but any other volume of equal dimensions can be used.

The **Colormap** popup menu allows the user to change the colormap of the display volume slice.

The **Show Histogram** button opens the Display Volume Histogram window (Figure C.16), where the histogram clipping for the display volume can be set independent

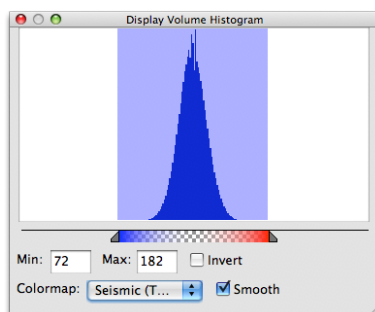


Figure C.16: The Display Volume Histogram window.

of the target volume.

### Scale Editor

The default scale for the data displayed in the Mesh Painter is the same as that in the main 3D Display. The scale used in the Mesh Painter can be set independently by clicking the Set Scale button (in the Display Volume section of the control panel), which opens the Mesh Painter Scale window (Figure C.17). The values shown in this window are multipliers for each of the three axes.

#### C.3.3.3 Appearance (Control Panel Section)

The **Appearance** section of the control panel contains settings related to the display of target voxels and painted regions.

The **Paint Color** button opens a color picker window that allows the user to set the color of painted regions. The paint color is always partially translucent.

The **Target Voxel Color** button opens a color picker window that allows the user to set the color of target voxels.

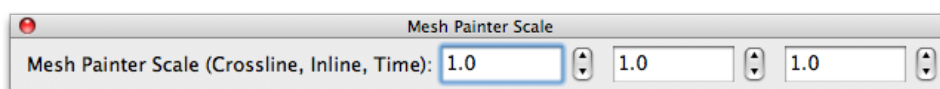


Figure C.17: The scale editor for the Mesh Painter.

The **Onion Skinning** checkbox turns the onion skinning feature on or off. This feature is described in more detail in Section C.3.1.2.

When the **Flash Paint** checkbox is turned on, the painted region will flash rapidly. This can be helpful for drawing attention to small areas of paint that need to be cleaned up. Pressing the T key will also toggle this option on or off.

#### C.3.3.4 Hiding the Control Panel

Frequently, the control panel settings will only need to be changed once, when the Mesh Painter is first opened, and it can be useful to keep it hidden at other times in order to provide more space to the slice display. The control panel can be hidden by expanding the slice display to fill the entire window, which is accomplished by clicking the maximize button at the upper right corner of the display (Figure C.18).<sup>2</sup>

### C.4 Surface Draping Process

This section describes aspects of the user interface for Surface Draping that differ from those implemented in the Surface Wrapping process.

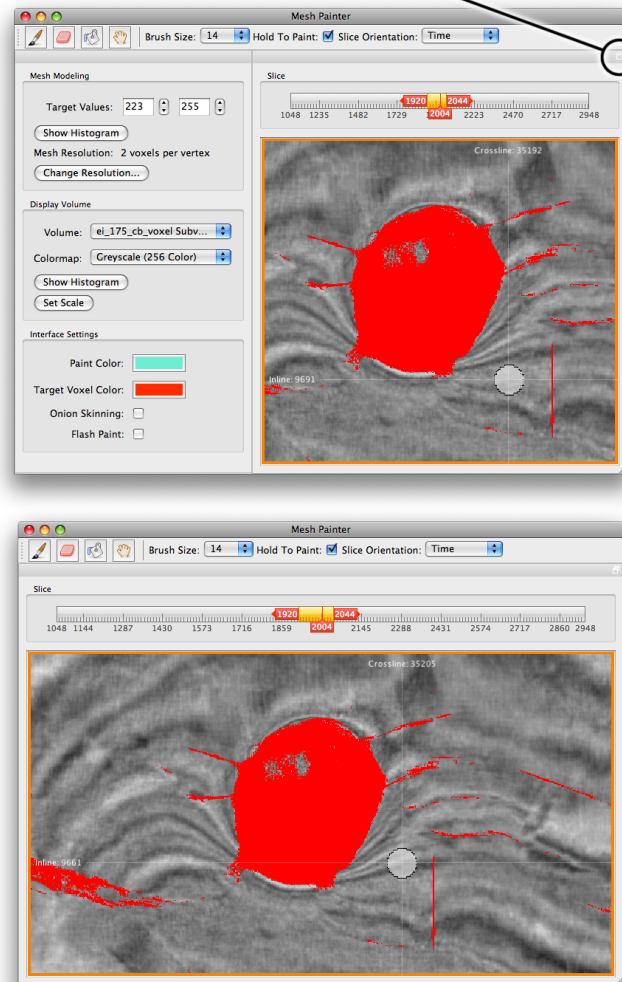
The input and output ports for the Surface Draping node are similar to those of the Surface Wrapping node, but the names and uses of the ports are somewhat different, reflecting the different typical applications of Surface Draping.

The first input port, “Target(s),” has the same purpose as that port on the Surface Wrapping node: it is a required input that accepts one target volume or point set, or both. The name of the second input was changed from “Bounding Mesh” to “Initial Surface,” since the draping behavior is typically applied to laterally extensive, non-closed meshes (although there is no mechanism that would prevent the user from

---

<sup>2</sup> Arguably, a disclosure triangle or similar affordance at the top left of the control panel itself would serve this purpose better. The mechanism used here was chosen purely for expediency of implementation since this feature was built into the windowing toolkit with which Insight Earth is written, and since the “maximize button” affordance is used elsewhere in Insight Earth, its appearance here is less likely to seem foreign to users of the application.

Click to maximize the slice display



**Figure C.18:** Clicking the maximize button for the Mesh Painter's slice display (top) hides the control panel (bottom).

connecting a closed mesh to this port). Unlike Surface Wrapping, the basis mesh for Surface Draping is always created externally, so this input is not optional.

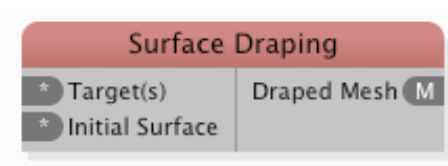


Figure C.19: The process node for Surface Draping.

Name	Control	Description
Set Target Voxel Range	button	Opens a separate window with which the range of target voxel values can be set.
Raise Mesh	checkbox	When turned on, causes the mesh to raise upward rather than move downward. The default value is Off.
Surface Permeability	number field	Data-aware surface smoothing. Possible values range from 0.0 to 1.0. The default value is 0.7.
Elasticity	number field	Controls the elasticity of the bounding mesh. Possible values range from 0.0 (more stiff) to 1.0 (more flexible). The default value is 0.5.

Table C.2: Surface Draping process parameters.

Name	Control	Description
Deformation Constraints	popup menu	<p>Restricts how the mesh will deform relative to the orientation of the input volume.</p> <p>Possible values are:</p> <ul style="list-style-type: none"> <li>• None</li> <li>• No Time Deformation</li> <li>• No Inline and Crossline Deformation</li> </ul> <p>The default value is “No Inline and Crossline Deformation.”</p>
Z Deformation Scale	number field	<p>A multiplier that specifies how rapidly the mesh should deform along the Z axis relative to the X and Y axes. The default value is 1.0.</p>
Solid Volume Boundary	checkbox	<p>When checked, the sides of the target volume are treated as target voxels, preventing the mesh from expanding beyond the volume bounds. The default value is On.</p>
Prevent Intersections	checkbox	<p>When checked, stops the bounding mesh from passing through itself. The default value is Off.</p>

Table C.2: Surface Draping process parameters.

Name	Control	Description
Snap Up	popup menu	<p data-bbox="824 268 1403 487">Specifies the snapping mode for vertices that have an upward-pointing normal when they become fixed on the target. Possible values are:</p> <ul data-bbox="899 550 1016 751" style="list-style-type: none"> <li data-bbox="899 550 993 575">• None</li> <li data-bbox="899 638 984 663">• Peak</li> <li data-bbox="899 726 1016 751">• Trough</li> </ul> <p data-bbox="824 814 1162 840">The default value is None.</p>
Snap Down	popup menu	<p data-bbox="824 877 1403 1096">Specifies the snapping mode for vertices that have a downward-pointing normal when they become fixed on the target. Possible values are:</p> <ul data-bbox="899 1159 1016 1360" style="list-style-type: none"> <li data-bbox="899 1159 993 1184">• None</li> <li data-bbox="899 1247 984 1272">• Peak</li> <li data-bbox="899 1335 1016 1360">• Trough</li> </ul> <p data-bbox="824 1423 1162 1449">The default value is None.</p>
Snap Fwd Window	number field	<p data-bbox="824 1486 1403 1696">Specifies the distance (in voxel units) to search forward from a vertex's location when a snapping mode is enabled. The default value is 8.0.</p>

Table C.2: Surface Draping process parameters.



Name	Control	Description
Snap Bwd Window	number field	Specifies the distance (in voxel units) to search backward from a vertex's location when a snapping mode is enabled. The default value is 2.0.
Draping Steps	slider	After pre-processing has completed, this slider is used to interactively fit the basis mesh to the volume data. This control is disabled until the Surface Draping process has been run. The default value is 0.
Tighten Mesh	button	Clicking this button pulls the mesh slightly more taut. It can be clicked repeatedly to achieve better results. This button is disabled until the Surface Draping process has been run.

Table C.2: Surface Draping process parameters.

#### C.4.1 Setting the Target Voxel Range

The target voxel range for Surface Draping is set by clicking the Set Target Voxel Range button in the Properties Inspector. This opens a window similar to the Mesh Painter for Surface Wrapping, but with the a restricted set of controls pertaining only to specifying the threshold values for target voxels (Figure C.20).

The options shown in this window have the same functionality as those used for Surface Wrapping, as described in Section C.3.1.

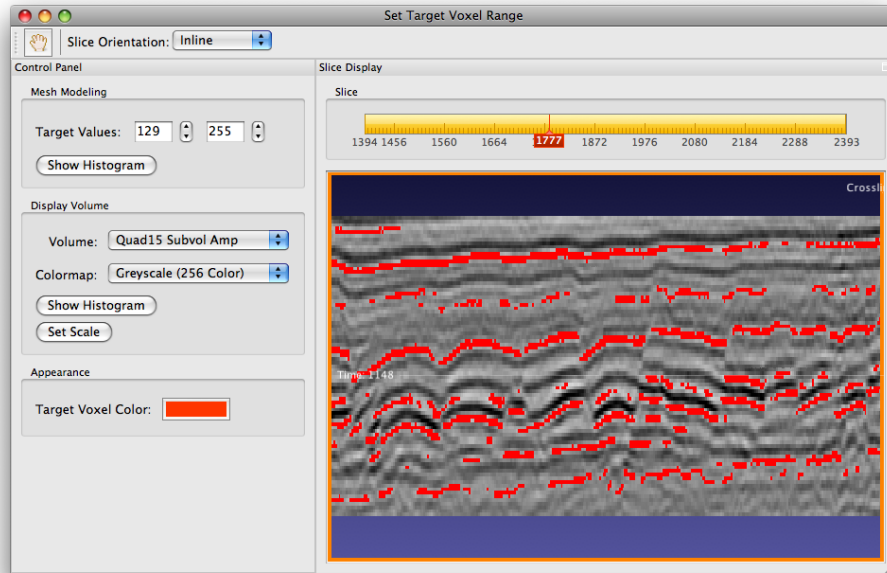
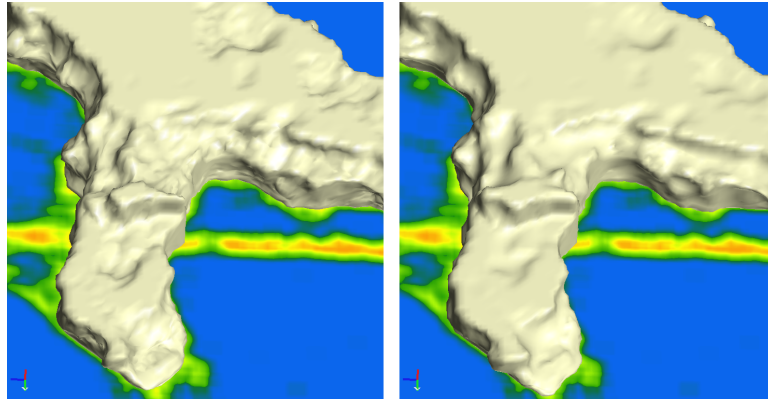


Figure C.20: Setting the target voxel range for Surface Draping.

#### C.4.2 Creating a Basis Mesh for Surface Draping

Unlike Surface Wrapping, the basis mesh for Surface Draping is always created from outside of the process itself and then connected to the Initial Surface input port in the corresponding node in the Process Network. The two most common methods for defining an initial surface mesh are importing an existing interpreted surface or using the Point Set Infill process to create a new surface within Insight Earth (Section 3.5.3).

Either a mesh or a Z-grid may be connected to the Initial Surface input port on the Surface Draping process. If a Z-grid is connected, the process converts it internally to a basis mesh with a vertex density of one vertex per voxel on the X-Y plane. When using an existing interpreted surface that has been picked directly on the target event, a static shift in Z should be applied prior to connecting the surface to the input port so that there is some vertical separation between the basis mesh and the horizon of interest.



**Figure C.21:** Close-up view of a bounding mesh with an elasticity value of 1.0 (left) vs. an elasticity value of 0.0 (right).

## C.5 Common Process Parameters

This section details the major parameters shared by both the Surface Wrapping and Surface Draping processes.

### C.5.1 Elasticity

The Elasticity parameter controls the simulated elasticity of the bounding mesh: a higher value will produce a more flexible, stretchy mesh, while a lower value will cause the mesh to be more stiff (Figure C.21). Using a high elasticity value can allow more detail to be captured, but can also cause the mesh to extend farther into gaps in the data. A low elasticity value produces a smoother mesh at the expense of detail.

### C.5.2 Surface Permeability

Surface Permeability is a form of data-aware smoothing operation that helps reduce unwanted spikes in the final bounding mesh (see Section 3.6.1). For many datasets, it is not possible or practical to eliminate outlying voxels whose values fall within the target voxel range, but which are not part of the region of interest. Because these are classified as target voxels, the mesh will normally become fixed at those out-

lier points which fall between the basis mesh and the boundary of the target object, causing undesired spikes in the final mesh.

Possible values for this parameter range from 0.0 to 1.0. A value of 0.0 disables Surface Permeability, while using the maximum of 1.0 will cause the mesh to pass through all target voxels. This parameter should be adjusted on a case-by-case basis, but in general, a value between 0.5 and 0.8 will yield good results.

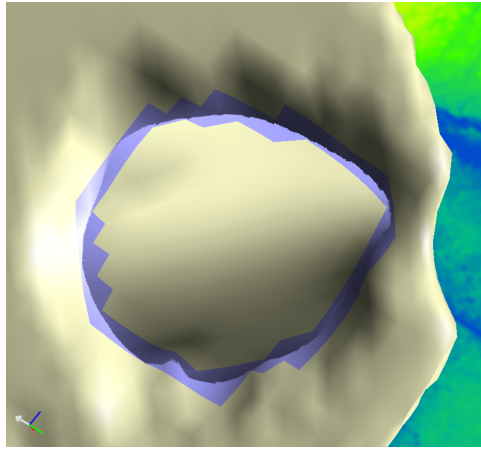
### **C.5.3 Prevent Intersections**

When enabled, the Prevent Intersections option will stop the bounding mesh from passing through its own surface. Self-intersections can happen in places where the basis mesh is painted in a very thin layer across a portion of the volume where there are no target voxels, or when the basis mesh was painted too far from the region of interest.

This feature is disabled by default because it can significantly slow down the preprocessing step of Surface Wrapping and Surface Draping. For very large meshes, it is usually best to run the process first with Prevent Intersections disabled, then turn on the mesh's Highlight Intersections display option, which will indicate any places where self-intersections occur in blue (Figure C.22). If intersections do exist, the process can be rerun with the Prevent Intersections option enabled, using the same values for all other parameters; or, if there are only a few intersections, it may be faster to smooth those regions using local mesh editing tools.

### **C.5.4 Deformation Constraints**

This parameter allows restrictions to be imposed on how the mesh will shrink or drape relative to the orientation of the target volume. "No Time/Depth Deformation" only allows the mesh to deform in inline and crossline, which is often useful for interpretation of channels. "No Inline and Crossline Deformation" only allows deformation



**Figure C.22:** A self-intersecting mesh, with the Highlight Intersections display option enabled.

in time/depth; when used with a single-Z-value basis mesh, this will yield single-Z-value output, which is otherwise not guaranteed. The “None” setting allows full 3D deformation.

### C.5.5 Z Deformation Scale

This parameter may be used when it is necessary to compensate for the aspect ratio of a target volume that has a vertical scale that is greatly different from its horizontal scale (a common occurrence in both medical and seismic volumes). The value acts as a multiplier: for example, a value of 2.0 will cause the mesh to deform twice as rapidly along the Z axis as the X and Y axes. This parameter only has an effect when the Deformation Constraints parameter is set to None, since the other two settings restrict the mesh deformation to either exclusively vertical or exclusively horizontal motion.

### C.5.6 Wrapping/Draping Steps

The Wrapping Steps (for Surface Wrapping) and Draping Steps (for Surface Draping) sliders control the extent to which the initial mesh is deformed. For smaller meshes, the slider can be scrubbed back and forth to interactively set this value. For

larger meshes, it may be preferable to enter a value directly in the text box to the right of the slider, depending on how rapidly the mesh updates in the 3D Display.

### **C.5.7 Tighten Mesh**

When the Tighten Mesh button is clicked, portions of the mesh that are not fixed are pulled taut, which can help improve the fit of the final surface in some situations. The button can be pressed multiple times to further increase the simulated surface tension.

## **C.6 Related Mesh Display Options**

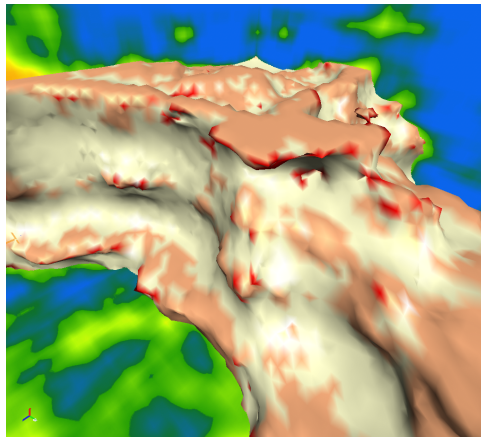
There are two display options for meshes that are particularly relevant to Surface Wrapping and Surface Draping.

### **C.6.1 Show Intersections**

When this option is turned on, any places where the mesh intersects with itself will be highlighted in blue (Figure C.22). This can be a helpful diagnostic tool for determining if the process should be re-run with the Prevent Intersections feature enabled.

### **C.6.2 Highlight Fixed Vertices**

When this option is turned on, vertices that have become fixed on target voxels are highlighted in light red (Figure C.23). If the value of the Surface Permeability parameter is greater than 0.0, any vertices which are sharp enough to invoke the smoothing operation are highlighted in dark red.



**Figure C.23:** Highlight Fixed Vertices turned on, with a surface permeability value of 0.8.