MULTIVARIATE TRANSFER FUNCTION DESIGN

by

Liang Zhou

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computing

School of Computing

The University of Utah

August 2014

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Liang Zhou**

has been approved by the following supervisory committee members:

Charles Hansen	, Chair	**04/29/2014** Date Approved
Christopher Johnson	, Member	**04/29/2014** Date Approved
Ross Whitaker	, Member	**04/29/2014** Date Approved
Valerio Pascucci	, Member	**04/29/2014** Date Approved
Claudio Silva	, Member	**04/29/2014** Date Approved

and by **Ross Whitaker**, Chair of the Department of **School of Computing**

and by David B. Kieda, Dean of The Graduate School.

ABSTRACT

Visualization and exploration of volumetric datasets has been an active area of research for over two decades. During this period, volumetric datasets used by domain users have evolved from univariate to multivariate. The volume datasets are typically explored and classified via transfer function design and visualized using direct volume rendering. To improve classification results and to enable the exploration of multivariate volume datasets, multivariate transfer functions emerge. In this dissertation, we describe our research on multivariate transfer function design. To improve the classification of univariate volumes, various one-dimensional (1D) or two-dimensional (2D) transfer function spaces have been proposed; however, these methods work on only some datasets. We propose a novel transfer function method that provides better classifications by combining different transfer function spaces. Methods have been proposed for exploring multivariate simulations; however, these approaches are not suitable for complex real-world datasets and may be unintuitive for domain users. To this end, we propose a method based on user-selected samples in the spatial domain to make complex multivariate volume data visualization more accessible for domain users. However, this method still requires users to fine-tune transfer functions in parameter space transfer function widgets, which may not be familiar to them. We therefore propose GuideME, a novel slice-guided semiautomatic multivariate volume exploration approach. GuideME provides the user, an easy-to-use, slice-based user interface that suggests the feature boundaries and allows the user to select features via click and drag, and then an optimal transfer function is automatically generated by optimizing a response function. Throughout the exploration process, the user does not need to interact with the parameter views at all. Finally, real-world multivariate volume datasets are also usually of large size, which is larger than the GPU memory and even the main memory of standard

work stations. We propose a ray-guided out-of-core, interactive volume rendering and efficient query method to support large and complex multivariate volumes on standard work stations.

For my wife, family and friends.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION

One-dimensional (1D)

Two-dimensional (2D)

Three-dimensional (3D)

Transfer Function (TF)

Graphics Processing Unit (GPU)

Open Graphics Library (OpenGL)

OpenGL Shading Language (GLSL)

Parallel Coordinate Plot (PCP)

Personal Computer (PC)

Computed Tomography (CT)

Magnetic Resonance Imaging (MRI)

ACKNOWLEDGEMENTS

It was the interest of computer graphics that brought me to University of Utah in August 2008. I took the interactive computer graphics class taught by Professor Charles (Chuck) Hansen, my advisor-to-be. The class gave me an opportunity to become a research assistant in Chuck's research group, which tackles challenging scientific visualization problems. It actually took me a while to fully understand and appreciate the importance and beauty of visualization and to shift my interest from graphics to visualization. I would like to thank Chuck for being so nice as an advisor and for ensuring the funding for my study. Moreover, I wish to thank him for his style of advising: encouraging students to do research projects independently from the beginning to the end. He never tells you what to do next, but expects you to do it by yourself first and then helps to make it better with a full heart. I have to admit that the learning curve is steep but the gains from this process are tremendous and life-long.

A collaboration with ExxonMobil on multivariate volume visualization and analysis was built in the early days in my Ph.D. study and lasted throughout the process and is still active. It was my great pleasure to meet and know Mark Dobin and Guo-shi Li who enabled my research to help real users in the petroleum industry. They also made it possible for me to do an internship in the Upstream Research Company from which several of my research topics stemmed. I would like to thank Mark for his passion, creativity, generosity and helpfulness. Mark, as a geophysicist, always thinks out of the box and gives suggestions for my work. I would like to thank Guo-shi for being such a great mentor.

It has been my privilege to work in the SCI Institute for the past five years. SCI has allowed me to meet and interact with peer students in different research areas, and to learn cutting edge techniques from the talks given by distinguished researchers. SCI has given me the unique opportunity to work in such a great

environment where I can enjoy the beauty of the slopes of the city and aromatic coffee. I wish to thank Chris Johnson for making it all possible and being a member of my committee. I would also like to thank Ross Whitaker, Valerio Pascucci and Claudio Silva for kindly being my committee members. It was a joint effort from all of you that introduced me to the world of visualization and helped me to prepare for the research challenges. I also wish to thank the entire administration team of SCI.

I wish to thank my colleagues for their helps. It would have been impossible to finish this dissertation without the help from members of Chuck's research group. I thank Mathias Schott who helped me write my first manuscript and answered all problems I have on coding and graphics hardware. I would also like to thank Pascal Grosset, Yong Wan, Carson Brownlee, Mark Kim, Guoning Chen and Aaron Knoll for their help in academic and life throughout my study. I remember the inspiring and fruitful discussions with Bo Wang and Wei Liu on many topics related to image processing and machine learning. I would like to thank them and Xianzong Xie, Zhan Wang and Dafang Wang for being such good friends who have helped me out more times than I can count.

My family has always been my greatest support and the biggest reason why I have been working hard. It is my parents who have made it possible for me to pursue my goals in life without any hesitation, and it is also their support that helped me to overcome all the difficulties in this place an ocean away from home. I wish to thank my wife Peng Lin who is such a wonderful companion of mine and helps me not only keep everything in good shape in everyday life but also to find the next goal in my career. Also, I am fortunate to have such caring and supportive parents-in-law and all other family members.

Finally, my research was made possible by the following grants: Award No. KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST), DOE SciDAC:VACET, NSF OCI-0906379, NIH-1R01GM098151-01. DOE NNSA Award DE-NA0000740, DOE SciDAC Institute of Scalable Data Management Analysis and Visualization DOE DE-SC0007446, NSF ACI-1339881, NSF IIS-1162013.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Volumetric data are important to many application domains, and are usually visualized using direct volume rendering. In recent years, the volume datasets utilized by domain users have evolved from univariate to multivariate. Multivariate volume datasets have become increasingly important and popular. Meteorologists simulate atmosphere movements, e.g., hurricanes, which involves the interaction of dozen of physical parameters to study their causes and try to forecast them. Combustion simulations involving multiple chemical elements and physical measurements can help engineers to improve the efficiency of an engine. Geophysicists compute several attributes derived from the amplitude of three-dimensional (3D) seismic survey data to facilitate the exploration of oil and gas [12]. Physicians determine the location of lesions in the brain using MRI scans that contain multiple channels. Transfer function design is the major means for exploring volume data and extracting features in direct volume rendering. Transfer functions of 1D or 2D are commonly used for the exploration of univariate volume datasets, and multivariate transfer functions are used to explore multivariate volume datasets. However, extracting clear features of interest for univariate volumes using 1D or 2D transfer functions is usually difficult due to the limited classification ability of these low-dimensional transfer functions. On the other hand, although methods and systems have been proposed for multivariate volume exploration and visualization, domain users urgently need an intuitive and flexible multivariate visualization tool that is able to extract meaningful features in complicated real-world datasets. Moreover, due to the boost of accuracies of acquisition devices, storage and computational capacities of computers, multivariate volume datasets are generated with ever increasing size.

Giga-scale or even tera-scale multivariate volume datasets have become common, and fast volume rendering and query techniques are vital for the visualization of these datasets.

In this dissertation, we present our proposed approaches to the multivariate transfer function design problem. This chapter first introduces transfer function combinations, in which we propose to combine the best features of existing transfer function spaces to create a transfer function space that provides better classification. It then introduces a transfer function design approach based on user-selected samples in the spatial domain to make multivariate volumetric data visualization more accessible for domain users. Next, it introduces GuideME: an automated technique for designing optimal multivariate transfer functions with a simple and easy-to-use slice-based user interface for highly complex volumes. Finally, it introduces our work on scalable out-of-core methods for interactive rendering and efficient querying for large multivariate seismic volumes on consumer level PCs.

1.2 Transfer Function Combinations

Direct volume rendering is an active area of research. Mapping of data values to optical properties, known as classification, remains a challenging problem. Transfer functions are most commonly used for classification in volume rendering, yet finding good transfer functions remains a difficult problem. For material boundaries, it has been shown that 2D transfer functions provide greater specificity[1] than 1D transfer functions [54, 60]. In many datasets, separate features may share the same scalar value and gradient magnitudes and as such scalar value, gradient magnitude tuples are not sufficient for separating such features.

Recently, many new 2D transfer function spaces have been proposed to improve the classification from different metrics. The size-based transfer function is a transfer function space [14] built upon blob detection techniques using scale space theories to classify objects based on their sizes. The occlusion spectrum is another 2D transfer function space [15] that takes into consideration ambient occlusion

[1]We use the disambiguation definition of specificity rather than the statistical definition, which means the proportion of negatives in a binary classification test that are correctly identified.

within the volume for discriminating between features of similar scalar values. It is also possible to compute statistical measurements such as mean value and standard deviation in a local region around a voxel [38] to form a 2D transfer function space. All these methods are effective on some datasets. Other datasets, however, may contain materials that have similar statistical properties but occlude each other, or have materials that share similar statistical properties and occlusion measurements but differ in size.

We propose, therefore, to combine the best features of these transfer functions to create a transfer function space that provides better specificity. Our contributions in this work are threefold: 1) combining 2D transfer function space with 1D transfer function spaces with a basic approach for selecting combinations, 2) a user interface supports transfer function design in the combined transfer function space, 3) experiments and detailed discussions of different transfer function combinations and original 2D transfer functions on various datasets.

We experimented with combinations of these transfer function spaces and discuss a basic approach for selecting combinations that improve classification and show that this combined transfer function space provides better classification than 2D gradient magnitude transfer functions, 2D statistical transfer functions, 2D occlusion-based transfer functions or 2D size-based transfer functions.

1.3 Transfer Function Design Based on User-Selected Samples for Intuitive Multivariate Volume Exploration

Multivariate dataset visualization has been an active research area for the past decade and remains a challenging topic. A linked-view visualization system that enables the users to explore the datasets in both the transfer function domain and the spatial domain may boost their understanding of the data. In recent years, visualization researchers have been studying this topic and some solutions have been proposed [21, 1, 5, 31]. These linked-view systems provide users the ability to explore the dataset with closely linked scientific visualization views, e.g., volume rendering or isosurface rendering, and information visualization views, e.g., scatter plots, parallel coordinate plots (PCP) or dimensionality reduction

views. Typically, the user explores and extracts features of interest by interactively designing transfer functions (TFs) in the value domain over the information visualization contexts and examining classified results in the spatial domain from the scientific visualization view. Successful examples using these systems are clearly shown for simulation datasets. However, extracting meaningful features in real-world measurement datasets, e.g., multivariate 3D seismic survey, via these systems is not trivial. Features inside the seismic dataset have to be recognized in the spatial domain by a geology expert, and the features have complicated combinations of attribute values and subtle differences from their surroundings. Therefore, it is too laborious to extract features by iterating between TF design in the value domain and getting feedback from the results rendered in the spatial domain, especially when the dimensionality is high. Our geophysicists collaborators have found extracting features with only the value domain TF widgets, e.g., on a PCP to be cumbersome, and specifically asked for more automated methods.

We propose a TF design approach based on user-selected samples from the spatial domain represented as slices for more intuitive exploration of multivariate volume datasets. Specifically, the user starts the visualization by probing features of interest in a panel view, which simultaneously displays associated data attributes in slices. Then, the data values of these features can be instantly and conveniently queried by drawing lassos around the features or, more easily, by applying "magic wand" strokes. High-dimensional transfer functions (HDTFs) can then be automatically and robustly generated from the queried data samples via the kernel density estimation (KDE) [87] method. The TFs are represented by parallel coordinate plots (PCPs) and can be interactively modified in an HDTF editor. Automatically generated Gaussian TFs in dimensionality reduced 2D view can also be utilized to extract features. The extracted features are rendered in the volume rendering view using directional occlusion shading to overcome artifacts from Phong shading in the multivariate case. To further refine features, which share similar data value ranges, direct volume selection tools on the volume rendering view or the panel view can be applied.

The contributions of this work are the following: First, we propose a transfer

function design method for multivariate volume visualization based on user-selected samples, specifically an HDTF generation method based on KDE and a Gaussian mixture model based 2D Gaussian TF generation method. Second, we also propose an interactive multivariate volume visualization system based on the proposed method that has been implemented to allow domain users to extract refined features in very complicated multivariate volume datasets more intuitively.

1.4 GuideME: Slice-guided Semiautomatic Multivariate Volume Exploration

The state-of-the-art method for exploring multivariate volumes is user interaction with multiple linked view systems. This method requires the user to explore the volume using parameter views, e.g., parallel coordinate plots (PCP) or histograms, in a trial-and-error manner [1, 5, 31]. Although these systems have been successful in simulation datasets where the user understands the "recipe" of the parameter space, i.e., knows what combinations of value ranges of attributes may result in interesting features, it is difficult for the user to explore complex measured datasets, e.g., seismic datasets. To this end, research efforts have addressed the exploration of complex datasets such as seismic data [103, 43, 44]. In [103], the approach allows domain users to apply their expertise to the finding of features by directly selecting a region of interest in a multipanel slice view. However, these methods either work only on univariate seismic data for a certain type of features [43, 44] or still require transfer function tuning with a PCP-based or a histogram-based editor [103], which can be unintuitive and time consuming for domain users. Moreover, switching between multiple views may be somewhat distracting.

Seismic datasets are an important tool for the petroleum industry which is the driving application of our method. Geophysicists interpret features that indicate potential oil and gas reservoirs, including channels and salt domes, on 3D seismic data slices. To interpret the seismic data, they first identify and locate geological features on slices from the 3D seismic data through examination and selection. With the advancement of multivariate 3D seismic [12] interpretation, attributes derived

from the seismic amplitude are used to aid in the extraction of relevant features. Interpretation is done mainly by free-hand drawing on slices and refinement of the features through multiattribute transfer functions [103, 90]. Users typically perform the following tasks during interpretation: selecting features by drawing on 2D slices, refining those features through transfer function manipulation, and examining results in 3D renderings. Other domains use similar tasks and we demonstrate the generality of our approach with multimodal MR brain scans.

We therefore propose GuideME, a novel method for multivariate volume exploration that strives to provide the user with a very simple and intuitive exploration process for highly complex datasets. Instead of multiple linked views, our method has only one slice view coupled with a focus overlay and a volume rendering view, and the tedious trial-and-error interactions are largely replaced by a guided uncertainty-aware lasso and automated feature extraction. The user starts the exploration by browsing through the slices and detects a feature of interest using his/her domain knowledge. A focus window that allows the user to inspect other attributes that can be placed over the feature of interest. Through the inspection, the user determines attributes that best describe the boundaries of the feature. A boundary confidence image is then blended with the slices, and the user can easily select the region with a guided uncertainty-aware lasso that automatically snaps to the detected feature boundaries. The selected region is then used as input for the automated feature extraction. Eventually, the feature is volume rendered and may be optionally edited directly in the 3D view. Using a highly complex real-world seismic dataset and multimodal MR brain scans, we show our approach is efficient, and is able to create results comparable to those given by previous method and ground-truth segmentations.

In this work, we make the following contributions:

- A novel slice-guided semiautomatic multivariate volume exploration workflow. The user is freed from unfamiliar parameter space views and tedious trial-and-error transfer function tunings.

- A guided uncertainty-aware lasso for region selection, based on edge detection and Dijkstra shortest path algorithms.

- A technique to automatically fine-tune a multivariate transfer function given the lasso, based on the optimization of a response function.

1.5 Interactive Multivariate Volume Rendering and Efficient Query on Standard PCs

Due to the advance in 3D seismic imaging techniques, resolution of 3D seismic datasets used in the petroleum industry is usually of giga-bytes. Multiple attributes derived from the original seismic amplitude dataset have been used for aiding the interpretation of the seismic survey. With these additional attributes, however, the size of the entire dataset may become far larger than the capacity of the GPU memory and even the main memory of a typical workstation. Recently, GPU-based multiresolution out-of-core volume rendering systems have been proposed. The Gigavoxel approach by Crassin et al. [16] and CERA-TVR by Engel [23] divide the dataset into multiresolution bricks and utilize an octree structure and ray guided paging system to efficiently render large sparse volume datasets. Hadwiger et al. [35] propose a rendering system that uses a virtual memory system and 2D mipmapping to support dense and noisy petascale microscopy scans. However, none of these methods are able to handle large multivariate volume datasets. In this work, we extend the approach by Hadwiger et al. to support interactive rendering of large multivariate seismic datasets on a consumer-level PC. On the other hand, data value querying raises another challenging issue for multivariate datasets especially when the data are very dense as hierarchical acceleration techniques, e.g., octrees may not be beneficial. We propose an efficient data querying technique based on precomputed per-block Gaussian mixture models and run-time ellipse-polygon intersection detection. An interactive exploration system has been built to allow the user to visualize the multivariate volumes as well as to edit multivariate transfer functions with the query feedback on parallel coordinate plots.

1.6 Dissertation Statement

Multivariate transfer function design is vital for the exploration and visualization of univariate and multivariate volumetric datasets, which are important

to many real-world applications. The design of multivariate transfer functions requires both existing and novel techniques in volume rendering, user interaction, image processing, data analysis and optimization.

1.7 Dissertation Contributions

The main contributions of this dissertation are:

- **More intuitive multivariate volume exploration and visualization workflows designed for domain users.** Our proposed methods strive to maintain the power and flexibility of existing workflows, while overcoming the shortcomings of existing approaches that are designed from the perspective of visualization experts. We provide domain users with new work flows that are designed as a result of close collaborations. The resulting workflows give domain users a central role and try to reduce unfamiliar widgets and views that may hassle users. For example, we allow the user to select samples directly on data slices to generate the initial transfer function, and then fine-tune the transfer function using parameter space widgets. A further developed workflow can be seen in GuideME, in which the parameter space is completely hidden from the user and the manual transfer function tuning process is replaced by an optimization approach. Through real-world examples, domain users find the new workflows more intuitive and efficient.

- **Improved classification ability of transfer functions demonstrated on complex real-world datasets.** We propose a transfer function combination method that generates multidimensional transfer function spaces by ranking and selecting the most helpful 1D or 2D transfer function spaces computed from a univariate volume. Moreover, transfer functions do not contain spatial information. Easy-to-use spatial fine tuning tools are therefore provided. Diffusion-based region growing tools and lasso tools are provided for efficient volume editing on transfer function classified results on either 2D or 3D image spaces. Complex real-world datasets, including CT lung scan, MR brain scans and multivariate seismic datasets, have been successfully used to demonstrate the improved classification ability of transfer functions.

- **Improved user interactions in the visualization process.** The goal for easier visualization experience for domain users is considered in all our proposed work. An easy-to-understand user interface that combines 2D and 1D transfer function widgets is built for the transfer function combination work. The user interface of the multivariate volume visualization system that is based on user-selected samples provides synchronized panel view, click-and-drag transfer function editors and hand drawing-based region selection and spatial fine tuning tools on 2D and 3D views. In GuideME, the user is given suggestions of feature boundaries and uncertainty information, and the selection of region is aided by lassos that automatically snap to feature boundaries.

1.8 Outline

The background and important related work are explained in Chapter 2. Chapter 3 details the work of transfer function combinations. The work of multivariate transfer function design based on user-selected samples in the spatial domain is presented in Chapter 4. Chapter 5 presents the method of GuideME, slice-guided multivariate exploration of volumes. In Chapter 6, we present the work of interactive rendering and efficient query of large multivariate seismic volumes on consumer-level PCs. Finally, conclusions and future work are given in Chapter 7.

CHAPTER 2

BACKGROUND

2.1 Transfer Function Spaces

Volume datasets can be explored using transfer functions. The most frequently used transfer function for volume rendering is a 1D transfer function that uses scalar values for classification. Realizing the poor classification ability of that transfer function space, Levoy [60] and Kindlmann et al. [54] use the gradient magnitude of the volume as another property for better classification. Kniss et al. [56] advocate and implement multidimensional transfer functions widgets, making the 2D transfer function a standard method in modern volume renderers. By far, the 1D and 2D transfer functions are the most popular and practical techniques for classification in volume rendering; however, great efforts have been made to define new transfer function spaces to improve the classification ability.

Due to noise and partial volume effects, selecting a boundary in the arches of a gradient magnitude-based transfer function is not easy and sometimes even impossible. To resolve this problem, Lundstrom et al. [64] employ the local histograms to better discern tissues in medical datasets and propose a 2D transfer function space that uses competitive classification certainty measure in addition to scalar values. Sereda et al. [86] use a 2D LH histogram-based transfer function for easier boundary identification and selection and further use this boundary information for a region growing segmentation schema.

Higher degree derivatives of the original scalar volume can also be used as transfer function spaces. Kindlmann et al. [55] use curvature as a second dimension of their transfer function domain. The curvature-based transfer functions allow nonphotorealistic renderings that highlight the contours of the volume.

The theory of scale-space, developed originally by the computer vision and image processing communities, can be used to classify objects based on their sizes. A commonly used scale-space representation is the linear Gaussian scale-space, which is essentially a convolution of a volume with differently sized Gaussian filters. Lum et al. [63] combine it with an image pyramid representation of different scales to improve classification. Correa and Ma [14] create a continuous scale-space for the volume and use anisotropic diffusion to detect "blobs" in the volume. The size of these defines an additional metric of the volume, which is then used to create size-based transfer functions.

Shape is another important aspect to classify an object. Sato et al. [83] use eigenvalue analysis on 3D local intensity structures to classify tissues in medical datasets with 2D transfer function spaces created using shape measurements: sheet, line or blob along with the scalar value. Prassni et al. [78] propose shape-based transfer functions by computing shape descriptors over presegmented volume to provide a manageable set of shape classified volumetric features with an intuitive optical properties assignment interface.

In many cases, different features occlude with each other but share similar scalar values, e.g., the skin and the gray matter in an MR brain scan. Correa and Ma [15] use the occlusion of a voxel as an additional dimension of the transfer function domain to classify features of similar scalar value, but different local neighborhoods.

Volumes can also be classified based on their statistical metrics, such as mean value or standard deviation of voxels in a certain neighborhood. Caban et al. [10] compute local statistical metrics and use their linear combinations to classify fine structures. Patel et al. [74] use a dynamically changing neighborhood to compute mean value and variance for voxels, thus defining a transfer function domain. A user interface then allows a selection of features based on the mean value, variance and radius of the neighborhood. Haidacher et al. [38] further extend this approach by selecting the radius semiautomatically via an adaptive sample selection technique.

Transforming the volume data into frequency domain is another idea for

generating transfer function spaces. Vucini et al. [94] utilize GPU-based fast Fourier transformation to support interactive frequency-based transfer function design that enhances conventional volume visualization.

2.2 User Interfaces for Transfer Function Design

A 1D transfer function that uses scalar values of the volume or a 2D transfer function that has the gradient magnitude of the volume as a second property for better classification [54] are most frequently used. As in most volume visualization systems available nowadays, e.g., Voreen [93], VisIt [62] and ImageVis3D [48], the transfer functions can be interactively defined by 1D transfer function widgets or 2D transfer function widgets proposed by Kniss et al. [56].

However, to design a good transfer function, the user has to manipulate the transfer function widgets in the transfer function space and check the result in the volume rendered image, which is laborious and time consuming. To address this issue, researchers have proposed to automate the transfer function design process. Many researchers focus on the automation of user interactions on the transfer function space. Maciejewski et al. [65] utilize KDE to structure the data value space to generate initial transfer functions. Wang et al. [96] initialize transfer functions by modeling the data value space using the Gaussian mixture model and render the extracted volume with preintegrated volume rendering. Most recently, Ip et al. [50] propose a hierarchical visual segmentation method using normalized-cut on the intensity-gradient magnitude 2D transfer function space to assist the volume exploration process. Although these automated methods applied on the transfer function space significantly reduce the time a user spends in the volume exploration process [50] compared to the most commonly used transfer function widgets, interacting with the transfer function space is not intuitive.

Therefore, the potential of transfer function design on more intuitive spaces has also been studied. One strategy is to provide the user with a gallery of predefined transfer functions, and then the user can easily design customized transfer functions by picking features of interest from the gallery and refining them. Marks et al. [67] propose *Design Gallery* as a general user interface for

computer graphics applications. As for volume rendering, the design gallery automatically defines a set of random 1D transfer functions and generates the resulting thumbnails by the dispersion heuristic. The resulting thumbnails are arranged using multidimensional scaling algorithm. The user can then select a preferred thumbnail and fine tune its associated transfer function. Similarly, a spreadsheet-like interface is proposed by Jankun-Kelly and Ma [51] where the user can explore a range of parameter combinations at the same time and compare the results side-by-side. Tory et al. [89] propose a parallel coordinates style interface that provides an overview of rendering options and transfer function settings. The user can easily explore different parameters and backtrack the visualization history using the proposed interface. Tzeng and Ma [92] propose to use ISODATA (Iterative Self-Organizing Data Analysis) clustering on a small subset of all data voxels to classify the volume, and then these classified features are displayed in a cluster-space user interface. With this user interface, the user is able to design transfer functions without knowing the transfer function space, simply by picking features from the gallery of preclassified features and refining them using the clustering operation buttons and rendering property adjustment widgets.

Interacting with the spatial domain views, e.g., sketching on the volume rendering view, is another intuitive option for the user. Several methods have been proposed to enable the user to design transfer functions by sketching on the spatial domain. Tzeng et al. [90, 91] propose systems that allow users to sketch the volume slices to assign color and transparency, and then high-dimensional transfer functions are generated using artificial neural network. Based on user sketches on the rendered images, Wu and Qu [99] fuse multiple features in distinct rendering results into a comprehensive visualization. Ropinski et al. [80] propose a sketch-based user interface for 1D transfer function design where the user draws strokes to define foreground and background to extract layers in the volume, and then the transfer functions are refined by adjusting the color and opacity of each layer. A more convenient sketch based volume exploration system is proposed by Guo et al. [30] where a full set of tools have been developed to enable direct manipulation of color, transparency, contrast and other optical properties on the

volume rendering view by means of user drawn strokes. In its essence, the system intelligently defines 1D transfer functions based on user sketches.

Other researchers have proposed methods to design transfer functions by semantics. Salama et al. [82] propose a framework that allows visualization experts to design high-level transfer function user interface with semantic information. Given specific features of interest by the domain user, e.g., bones, skin and blood vessels, the visualization expert creates transfer function models described by sets of control points from many data instances of the same type. Each transfer function model can be written as a data point in a high-dimensional space, and the semantic parameters can be generated using principal component analysis (PCA) on these high-dimensional data points. A simple semantics editor user interface can then be created from the semantic parameters that modify the control points of the low level 2D transfer function widgets. The user simply needs to adjust a set of sliders for each semantic parameter.

2.3 Multivariate Data Visualization

Visualizing and understanding multidimensional datasets has been an active research topic in information visualization. The scatter plot matrix is a straightforward yet scalable way that utilizes pairwise scatterplots as matrix elements to represent multidimensional datasets. The scatter plot matrix trades the resolution of each scatter plot to display more plots. Due to the large amount of plots shown in a scatter plot matrix, exploration of the multivariate space becomes cumbersome and time consuming. Elmqvist et al. [22] therefore propose a method for navigating through the plots. Other researchers have studied how to choose the display order of the scatter plots [85].

Parallel coordinate plots (PCPs) [49] is a popular multivariate visualization technique that overcomes the two-variable limit of the scatter plot. PCP arranges individual variable axes parallel to each other and represents individual samples as a polyline passing through all axes. With increasing samples, the PCP will become more cluttered and the rendering cost will be prohibitively expensive. A large amount of research efforts have been done to address this over-plotting

issue. Fua et al. [26] propose to cluster the data values and render only the representing polylines of each cluster. Novotny and Hauser [70] separate the number of polylines to render from the number of data samples by generating the PCP from 2D histograms of adjacent variable pairs, and outliers are also identified through histogram analysis. Zhou et al. [102] perform visual clustering on the PCPs by drawing curved edges instead of polylines and optimizing the arrangement of these curved edges. Heinrich and Weiskopf [41] show how PCPs can be made continuous, which gives a smooth and uncluttered representation. The over-plotting issue can also be resolved by edge bundling, and researchers have proposed a variety of techniques: e.g., geometry-based edge bundling [17], hierarchical edge bundling [45] and force-directed edge bundling [46]. McDonnell and Mueller [68] address the over-plotting issue using illustrative rendering which applies opacity and shading effects, silhouettes emphasizing, shadows and halos to edge bundled PCPs.

Dimensional reduction and projection are other techniques for multidimensional data visualization. These techniques provide a similarity-based overview for multidimensional data. Numerous research efforts have been focused on this topic, and popular methods include principal component analysis (PCA) [53], multidimensional scaling (MDS) and Isomap [88]. To reduce computation complexity, techniques that apply classical dimensional reduction and projection methods to only a small subset of representative samples and project the remaining samples via interpolation have been proposed. These techniques include Landmarks MDS [19] and Pivot MDS [8]. Alternatively, Faloutsos and Lin propose the Fastmap algorithm [24], which has exactly $O(n)$ complexity. Fastmap utilizes dissimilarities between each sample and two pivot elements per coordinate axis to make its distance computation $O(n)$. More recently, Paulovich et al. [75] propose the part-linear multidimensional projection (PLMP) method, which requires only distance information between pairs of representative samples, and therefore is faster than previous methods for large datasets. Moreover, with a representative sample positioning strategy, PLMP is able to conduct dimensional projection for out-of-core datasets.

Parallel coordinate plots capture individual dimensional information well but suffer from the clutter problem and such plots require expertise to interpret, because data points are transformed into polylines and the polylines occlude with each other. It is hard and sometimes even impossible to check the data correlation between a single pair of attributes, let alone for multiple attributes on the PCPs. On the other hand, pairwise scatter plots provide a clear correlation between a pair of attributes. A 2D scatter plot of the dimensional reduced or projected space of the high-dimensional data provides an easy-to-understand overview of the high-dimensional space at the cost of losing individual dimensional information. It is nontrivial if not impossible to use only one of these techniques to provide the user with proficient insights to a multidimensional dataset. Unfortunately, providing several linked views: one for PCPs, one for pairwise scatter plots and yet another for dimensional reduced/projected scatter plot would cause a context jump for the user. Researchers therefore have proposed to take the advantages of these techniques and combine them in a unified plot. Yuan et al. [100] propose SPPS (scattering points in parallel coordinates), which draws pairwise scatter plots between each pair of PCP axes or adopts a DIMDS (dimensional incremental multidimensional scaling) scatter plot between a selected pair of PCP axes. They convert parallel coordinates segments into point plots and draw the PCPs as curves that pass through their associated points. To provide a seemless integration, a uniform brushing tool that allows linked brushing on either the PCPs or the scatter plots is also proposed.

2.4 Interactive Linked View Multivariate Volume Visualization System

Multivariate volume datasets can be explored using linked view systems that have shown to be useful for multivariate simulation data exploration. Early studies utilize multiple linked scatter plots as the data value view and the user brushes regions of interest on these plots to design transfer functions. The SimVis system [21, 77] allows the user to interact with several 2D scatter plot views using linked brushes to select features of interest in particle simulations rendered as polygons and particles.

As parallel coordinate plots have become a widely accepted method for multidimensional data visualization, researchers propose to build data value view together with transfer function widgets based on parallel coordinate plots. Akiba and Ma [1] propose a tri-space exploration technique involving parallel coordinate plots together with time histograms to help the design of high-dimensional transfer functions for time-varying multivariate volume datasets. Blaas et al. [5] extend parallel coordinate plots for interactive exploration of large multitime point datasets rendered as isosurfaces. Rübel et al. [81] build a cluster-based multivariate visualization system centered on the histogram-based parallel coordinates for very large multivariate time-varying particle simulation. Given a user-selected multivariate value range on any attribute axis on the parallel coordinates, a fast multivariate query using bitmap indexing is conducted. The query result is then represented as parallel coordinate plots using the method proposed by Novotny and Hauser [70]. Finally, the particles that satisfy the query are rendered in the spatial view using particle systems.

A parallel coordinate plot provides a good context for the definition of each attribute value in high-dimensional transfer functions. However, due to data point to polyline transformation and the occlusion issue by its nature, it is hard to observe high-dimensional features and check correlations between attributes in a parallel coordinate plot as stated in Section 2.3. Researchers therefore resort to the dimensional reduction and projection techniques in conjunction with parallel coordinate plots to provide the user with more insight. Zhao and Kaufman [101] combine multidimensional reduction and transfer function design using parallel coordinates but their system is able to handle only very small datasets. Guo et al. [31] propose an interactive HDTF design framework using both continuous PCPs and MDS technique accelerated by employing an octree structure. Guo et al. [32] develop parallel algorithms for multivariate volume rendering, continuous PCPs computation and MDS computation to make their work [31] scalable.

However, we have observed two limitations in the above systems: 1) the user has to explore the data via interactions on the transfer function view, which may be unintuitive for domain users and moreover makes exploration for real-

world datasets difficult, and 2) the visualization is merely produced with transfer functions and it is difficult to achieve a more refined result.

2.5 Interactive Large Volume Rendering

Ray casting and slice-based volume rendering are the two methods used for direct volume rendering. Thanks to the computational power of GPUs, volume rendering has become interactive. However, with the ever increasing size of volume datasets, interactive direct volume rendering of large volume datasets that cannot fit into the GPU memory, i.e., out-of-core volume rendering, is still a challenging topic. Building hierarchical structure for a volumetric data is a common way to enable and accelerate out-of-core volume rendering. In the early work of hardware assisted volume rendering, LaMar et al. [59] and Weiler et al. [97] propose to use hierarchical bricking schemes. Boada et al. [6] build a mipmap-like structure based on an octree, and then they choose a cut through the tree and use the mipmap data of the leaves during rendering. Guethe and Strasser [33] use hierarchical wavelet representation and screen-space error estimation for level of detail selection. The ImageVis3D system [48] uses a kD tree to subdivide data and each brick in the tree is rendered in one rendering pass. Gobbetti et al. [28] determine the visibility of octree nodes using the corresponding partial octree on the CPU, which is then downloaded to the GPU.

All work above requires a CPU-based traversal of an explicit hierarchical structure, which can be very expensive. More recently, due to the improvement of the GPU, GPU-based ray-guided volume rendering frameworks enable efficient rendering of gigascale and even petascale volume data on a single consumer level GPU. Crassin et al. [16] propose a GPU-based ray-guided octree volume rendering framework, called "gigavoxel," which uses ray casting information to directly guide the data streaming. The "gigavoxel" framework is efficient as the ray casting information naturally determines voxel visibility and view frustum. However, "gigavoxel" is intended for entertainment applications, which usually result in sparse octrees, and moreover, the kD restart octree traversal scheme requires a full path traversal from the root of the tree for each voxel, which can

be costly. Hadwiger et al. [35] present an interactive volume renderer that scales to petascale which uses a visualization-driven virtual memory approach. Similar to "gigavoxel," ray casting is also utilized to detect visible data, however, [35] avoids the potentially costly kD restart octree traversal. Unlike "gigavoxel,: [35] it is designed and optimized for dense anisotropic microscopy data. In contrast to all previous work, the virtual memory approach requires no precomputation of a multiresolution hierarchy. Instead, it constructs volume data from the 2D raw images based on a visualization-driven data construction scheme via on-the-fly stitching and re-sampling.

CHAPTER 3

TRANSFER FUNCTION COMBINATIONS

Direct volume rendering has been an active area of research for over two decades. Transfer function design remains a difficult task since current methods, such as traditional 1D and 2D transfer functions, are not always effective for all datasets. Various 1D or 2D transfer function spaces have been proposed to improve classification exploiting different aspects, such as using the gradient magnitude for boundary location and statistical, occlusion or size metrics. In this chapter, we present a novel transfer function method that can provide more specificity for data classification by combining different transfer function spaces. In this work, a 2D transfer function can be combined with 1D transfer functions and improves the classification. Specifically, we use the traditional 2D scalar/gradient magnitude, 2D statistical, and 2D occlusion spectrum transfer functions and combine these with occlusion and/or size-based transfer functions to provide better specificity. We demonstrate the usefulness of the new method by comparing it to the following previous techniques: 2D gradient magnitude, 2D occlusion spectrum, 2D statistical transfer functions and 2D-size based transfer functions.

3.1 Combining Transfer Functions

3.1.1 Formulating Transfer Function Combinations

In practice, using just one or two metrics during volume classification makes it difficult to robustly classify and separate features in complex volumes. Using more properties in the transfer function space often can better describe features in the volume; however, user interaction becomes more difficult or even impossible when the number of properties, and thus the dimensionality, of the transfer function space increases. Gaussian transfer functions have been proposed by Kniss et al. [57] to provide analytical multidimensional transfer functions of arbitrary

dimensionality. Also a procedural high-dimensional transfer function model is proposed in [36]. However, in both works, how to provide an effective user interface remains unclear.

The proposed transfer function combination sequentially applies two transfer functions, a 2D and a 1D one, to all voxels v_X, where X is the 3D position, of the input dataset V that has l properties $Y_1, Y_2, \ldots Y_l$.

$$\{C_1, C_2, C_3, \ldots C_k\} = \text{TFw}(V)\text{where}$$
$$C_j := \{v_X | \text{TFw}_j(Y_p(X), Y_q(X)) > 0, p, q \in [1, l]\} \tag{3.1}$$
$$\{W_i \subset C_j\} = \text{TF}_r(Y_r(X))\text{where}$$
$$X \in C_j(X), j \in [1, k], r \in \{1, \ldots, l\}/\{p, q\} \tag{3.2}$$

In Equation 3.1, a number of 2D transfer function widgets, k, are first applied to the volume, resulting in sets of classified voxels $C_1, C_2, C_3 \ldots C_k$, respectively. Then one from a set of r, which is typically 1 or 2, 1D transfer functions is applied to the classified region C_j, yielding the final classified volume region W_i. Each 2D transfer function widget has one associated 1D transfer function.

Kniss et al. [57] clearly show a 2D example that separating high dimensional transfer functions into lower-dimensional transfer functions using multiplication can lead to misclassification, which gets worse when the dimensionality is extended into 3D. Our proposed method, however, does not suffer from such issues as each 2D transfer function widget has a 1D transfer function that helps further separate features within the voxels selected by the 2D transfer function. This dimension reduction method, however, can cause classification inconsistencies compared to a true 3D transfer function. We believe that this is a reasonable compromise, considering that the losses in classification precision compared to using an equivalent higher dimensional transfer function are typically minor.

Rezk-Salama [79] proposed a similar idea called local transfer functions to set transfer functions for segmented volumes, i.e., a transfer function is associated with a tag in the tagged volume; voxels are essentially preclassified and their tags are stored in a volume. Our method is more flexible as the user essentially interactively labels voxels using the 2D transfer functions and then further classifies

the features using the associated 1D transfer function. Bruckner and Gröller [9] similarly use a 1D transfer function to index into a table of style transfer functions for flexible illustrative volume renderings. Their work conceptually differs from ours as our transfer function combination method is utilized to improve the specificity of transfer functions rather than producing illustrative visualizations.

3.1.2 Selecting Combinations

We propose to separate the transfer function space into a 2D transfer function space with a set of 1D transfer function spaces as a trade-off between dimensionality and usability.

A problem naturally arises when more than three properties/attributes are provided, namely which properties contain salient features, which attributes are most effectively used as the 2D transfer function domain, and which are best classified by the associated 1D transfer functions. Thus, we provide a few simple rules to aid the user in selecting appropriate combinations.

For a given set l properties of a dataset, the correlation coefficient matrix R of size $l \times l$ is computed, as well as the entropy vector E of size l, which contains all properties' entropy. The primary property Y_p, is chosen that represents the original information of the dataset (e.g., original intensity dataset or the mean dataset computed from the statistical properties extraction process as shown in Section 3.2.3). A property that is intrinsically associated with Y_p (e.g., gradient magnitude vs. original intensity dataset or standard deviation vs. mean value) is used as the *secondary* property Y_q. The primary and secondary properties define the 2D transfer function space. For all remaining properties Y_i, $i \in [1,l]$ and $i \neq p,q$ a score is computed as a linear interpolation between the correlation coefficient R_{pi} and the normalized entropy $\frac{E(Y_i)}{\max E}$, as shown in Equation 3.3:

$$s_i = -a|R_{pi}| + (1-a)\frac{E(Y_i)}{\max E}, 0 \leq a < 1 \tag{3.3}$$

The correlation coefficient depicts the similarity between properties: a lower correlation coefficient value indicates a higher independence of properties. By intuition, more independent properties correspond to more interesting features, which we hope can be extracted by combining them together. Therefore, we favor

properties that are less correlated with the already chosen properties and as such a negative relationship between the absolute value of correlation coefficient $|R_{pi}|$ and the score s_i is shown in Equation 3.3. Specifically, the coefficient matrix R of property Y_p and Y_i is computed by Equation 3.4.

$$R_{pi} = \frac{Cov(p,i)}{\sqrt{Cov(p,p)Cov(i,i)}} \tag{3.4}$$

where $Cov(p,i)$ is the covariance matrix of property Y_p and Y_i.

However, using the correlation coefficient alone could lead to situations where properties that do not increase classification ability can beat more meaningful properties in the scoring, and to remedy this, the entropy of a property is also considered in Equation 3.3. The entropy value of a property reflects the amount of information contained in that property, shown as a normalized form $\frac{E(Y_i)}{\max E}$ in Equation 3.3. The entropy is defined as

$$E(Y_i) = \sum_{b=1}^{n} p(y_b)\log_2(p(y_b)) \tag{3.5}$$

where n is the number of bins in the histogram of property Y_i, b is the current bin and $p(y_b)$ is the probability of data value y_b at current bin. $E(Y_i)$ describes the homogeneity of property Y_i and is negatively proportional to the homogeneity, i.e., higher entropy represents less homogeneity.

Properties that are less homogenous usually contain more features of interest compared to more homogenous ones. Therefore, low homogeneity can be used to rule out less contributing properties that have higher score from the correlation coefficient. As such, high entropy is desired in our scheme, i.e., properties that are less homogenous are favored over more homogenous ones. However, low entropy may also be of interest on some occasions, e.g., a property contains large homogenous regions but highlights a small feature that no other properties can. The classification ability of those properties, however, is hard to describe by mathematical quantities but can be rather easily determined subjectively.

The parameter a is dataset dependent and allows the user to choose a balance between the correlation of two properties and the amount of information contained in an individual property.

The remaining properties are then ranked based on their scores s_i and used as the *tertiary* attributes for the associated 1D transfer functions. We found that using one or two tertiary attributes provides a good compromise between complexity and effectiveness of the classification. One of the available tertiary attributes is selected as the active one for each widget in the 2D transfer function space.

As an example, the process of combination selection for CT chest scan Artifix discussed in Section 3.3.2 is shown. Using the rules, we compute the correlation coefficients and the entropies of the five properties of the dataset as shown in Table 3.1.

Choosing the scalar value x as the primary attribute suggests using the gradient magnitude $|\nabla x|$ as the secondary attribute. Then scores $s_{\mu,\sigma,\rho}$ for mean, standard deviation and occlusion properties are computed for the remaining attributes by setting a to 0.4, which yields $s_{\mu,\sigma,\rho} = [-0.0347, 0.1938, 0.3086]$.

The occlusion property has the highest score, meaning it is the best property regarding both the correlation between it and the primary attribute and the information it contains. The occlusion property is used as the tertiary attribute to define a combined 3D gradient magnitude/occlusion transfer function space.

Alternatively, choosing the mean value μ and the standard deviation σ as the primary and secondary attributes, the scores $s_{x,|\nabla x|,\rho}$ are computed for the other attributes, yielding $s_{x,|\nabla x|,\rho} = [0.0094, 0.2061, 0.3045]$ for scalar, gradient magnitude and occlusion properties, respectively. The occlusion property has the highest score and is thus used as the tertiary attribute to define a combined 3D statistical/occlusion transfer function space.

3.1.3 User Interface

In general, true 3D transfer function widgets are relatively difficult to interact with, since robust and effective interaction with a 3D space is still an open research problem [7]. The proposed combined transfer function space, however, is separable into a 2D transfer function space and a set of 1D transfer function spaces. Haidacher et al. [37] propose a similar separation method for multimodal visualization. In contrast to their simple triangle shaped windowing function, our method provides more insights and flexible controls for the 1D transfer function

Table 3.1. The correlation coefficients and the entropies of the properties computed from Artifix CT chest scan dataset. The properties are the intensity value x, the gradient magnitude $|\nabla x|$, the mean value μ, the standard deviation σ and the occlusion metric ρ. The bottom row shows the entropy E of each attribute.

	x	$\|\nabla x\|$	μ	σ	ρ
x	1.0000	0.1654	0.9973	0.2435	0.7286
$\|\nabla x\|$	0.1654	1.0000	0.1690	0.9569	-0.0583
μ	0.9973	0.1690	1.0000	0.2464	0.7388
σ	0.2435	0.9569	0.2464	1.0000	0.0067
ρ	0.7286	-0.0583	0.7388	0.0067	1.0000
E	5.3828	3.6077	4.8012	3.8392	7.9090

spaces. This separation, as stated before, can cause decreased classification precision when the 1D transfer function spaces are not independent from the 2D transfer function space compared to a true 3D transfer function space. However, our combination selection rules proposed in Section 3.1.2 help to rule out highly dependent 1D transfer function spaces. Therefore, we believe this separation is a good trade-off between interactivity and classification precision.

Each 1D transfer function is attached to every selected region in the 2D transfer function domain based on the usual transfer function widgets or selectors. Thus, features in the volume can be classified by selecting their voxels in the 2D domain defined by the primary and secondary attributes. In cases where those voxels represent multiple separate features, the additional 1D transfer function can be used to further separate such features within the voxels selected in the 2D domain using one of the tertiary attributes. While adding complexity to the manipulation of transfer functions, this technique provides familiar interaction with each of the 2D and 1D transfer functions (TF). We believe this additional interaction (combining familiar 2D TF manipulation with familiar 1D TF manipulation) provides a reasonable method for interacting with the higher dimensionality of transfer function combinations. However, it does require users to be familiar with such interaction techniques.

Figure 3.1 illustrates the proposed 3D transfer function editor for a 2D gradient magnitude transfer function space with associated 1D occlusion transfer functions. The top part shows the 2D gradient magnitude transfer function domain $x \times$

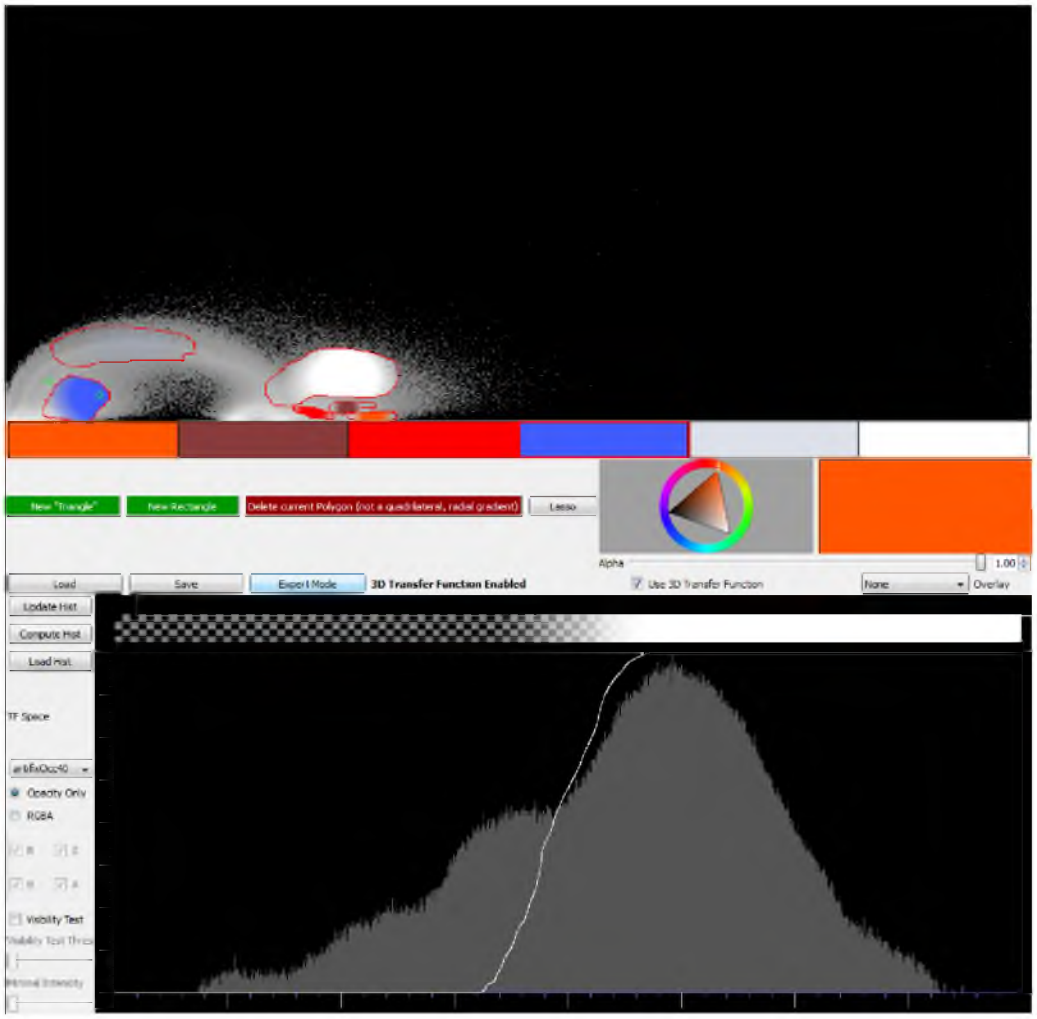


Figure 3.1. The separable 3D transfer function editor. The transfer function editor with the 2D gradient magnitude transfer function space $x \times \|\nabla x\|$ shown on top, and the 1D occlusion space ρ_c attached to the currently selected widget c in the 2D space, shown below. In this example, the blue widget is active, and the 1D histogram represents the occlusion information of all the voxels with statistical properties selected by the widget in the 2D statistical domain.

$\|\nabla x\|$, where the user can place and interact with traditional 2D transfer function widgets [56] TFw_{2D} and a more generic lasso tool. The occlusion volume space ρ_c or the size volume space t_c of the region c selected by the currently active 2D transfer function widget TFw_c is represented by a 1D transfer function editor, shown at the bottom, along with a 1D histogram of the occlusion information of all voxels selected by c. That is, the 1D transfer function editor operates strictly on voxels selected by a 2D transfer function widget (the blue widget in Figure 3.1).

The 2D transfer function widgets, such as ellipse, rectangle or triangle widgets as proposed by Kniss et al. [56], typically include some default shapes with few degrees of freedom. Users are able to set colors, opacities and different fall-offs for each of these widgets. These tools provide facilities to the user for a general exploration of transfer function spaces using easy to manipulate *high-level* widgets.

However, it is difficult for the user to precisely select arbitrary regions. This often prevents a user from exploring the subtle structures in the transfer function domain, which may make a significant difference in the final visualization. Thus, similarly to commonly used image processing applications, we also include a *lasso* tool to allow the user to intuitively and easily select arbitrary regions by drawing the region boundaries directly into the transfer function space. In Figure 3.1, the red curve illustrates the hand drawn boundary path with a spherical fall-off for the color and opacity. A box on the left hand side of the 1D transfer function editor allows the user to select which tertiary attribute is used as the 1D transfer function space for each 2D widget.

The proposed user interface allows the user to interact with the 3D transfer function space intuitively. Whenever the user creates a transfer function widget on the 2D transfer function space, the histogram of the voxels selected by that widget is computed and immediately shown in the 1D transfer function editor. Initially, the 1D transfer function maps, as visible, all voxels that are selected by the 2D transfer function widget. With the help of the 1D histogram, one can then design the 1D transfer function intuitively. As such, users are provided with a familiar interface thus providing intuitive interaction. This user interface adds minimal complexity to the standard 1D and 2D transfer function editors in existing volume visualization systems, e.g., Voreen [93] and ImageVis3D [48]. With a 3D transfer function space we are able to leverage the usability of the user interface; however, we are also interested in extending it for higher dimensional transfer function spaces in the future.

3.2 Specific Transfer Function Spaces Used

In addition to the well-known 2D gradient magnitude and scalar value transfer function, we include several recently proposed transfer functions to be used in combinations. Creation of these transfer functions is generally based on the methods described in the respective papers, but with slight modifications, which are discussed in the following subsections: size-based transfer functions [14] in Section 3.2.1, occlusion-based transfer functions [15] in Section 3.2.2 and statistical transfer functions [38] in Section 3.2.3.

3.2.1 Size Information Computation

Correa and Ma [14] proposed a three-step method to create a size volume S from an input volume. The three steps are scale-space computation, scale detection and back projection. Correa and Ma use anisotropic diffusion to create the scale space with better localization. The classical normalized Laplacian kernel is used to detect the blobs as local maxima both in spatial and scale domains. A back projection step utilizing Shepard's interpolation with Wendland polynomials is then conducted for the detected blob tuple (x, y, z, t).

A single voxel can be part of features with multiple sizes; however, only the largest size value is kept at each voxel. Thus smaller features get masked out by larger ones, which happens in the brain MRI example shown in Section 3.3.4. To avoid this situation, we allow the user to specify an intensity range to compute a scale space specifically for that range.

3.2.2 Occlusion Information Computation

Correa and Ma [15] suggest using an extended ambient occlusion metric to measure the occlusion of the volume. One can view the occlusion information ρ as a weighted sample mean value for a spherical neighborhood with certain radius R centered at each voxel, which results in an isotropic blurring effect that does not preserve the boundaries of the structures.

Sometimes, overly smoothed volumes that lose all their boundary information are not desired, thus we derive a gradient based equation for computing the occlusion information, inspired by work done by Perona and Malik [76].

For a sphere of radius R, we compute the occlusion information of the N voxels x_i surrounding the current voxel x as shown in Equation 3.6:

$$m_x = \frac{g_x}{N} \sum_{i=1}^{N} x_i \tag{3.6}$$

$$g_x = \frac{\eta^2}{\|\nabla I_x\|^2 + \eta^2} \tag{3.7}$$

In Equation 3.7, g_x is a term based on the gradient magnitude of the current voxel x.

The dataset dependent parameter $\eta \in \mathbb{R}^+$ handles gradients of zero magnitude e.g., for $\eta \in [0.001, 0.01]$, essentially helping to preserve boundaries of different structures. If $\eta \geq 1$, the filter behaves similar to a box filter.

Computing m_x is equivalent to convolving the volume with a spherical filter B_R of radius R, and then modulating it with g_x:

$$m_x = g_x \cdot (B_R * I_x) \tag{3.8}$$

The complexity of this operation is $O(mn)$, where $m = \frac{4}{3}\pi R^3 + 1$, and thus very costly, since the radius should be large enough to maximize the variance of the result [15].

This computational scheme is infeasible in practice, due to its computational complexity. However, since each sample inside the sphere is treated equally, a box filter of width $2R$ can be used to approximate the sphere. Exploiting the separability of convolving with a box filter and the performance of modern GPUs allows the computation of m_x within seconds. The 3D convolution is then separated into three consecutive convolutions with a 1D box filter b_{2R+1} of width $2R+1$, as Equation 3.9 shows:

$$m_x = g_x \cdot \{b_{2R+1} * [b_{2R+1} * (b_{2R+1} * I_x)]\} \tag{3.9}$$

This separation considerably reduces the computation time. Such an occlusion metric is view-independent and thus can be precomputed and stored, and therefore does not affect the speed of visualization.

3.2.3 Statistical Properties

We construct the statistical feature space with a procedure similar to that presented by M. Haidacher et al. [38]. They propose to grow a sphere over the neighborhood of each voxel and to compute the following statistical metrics: mean value μ, standard deviation σ, *skewness* as well as *kurtosis*. It is a multistage process: first, extract *statistical metrics* and second, conduct the *normality test*. If the test is passed, continue with the *similarity test*. After the similarity test, if the new samples are similar to the old ones, we combine the statistical metrics. If any of the above tests fail or a user-defined maximum radius r_{max} is reached, the procedure is terminated, otherwise we increase the neighborhood by one voxel.

Haidacher et al. [38] use the Jarque-Bera test [52] for normality since it is easily implementable on a GPU. It, however, requires a relatively large set of samples in order yield results of sufficient quality. Therefore, various other normality tests have been proposed in the literature; we chose D'Agostino's K-squared test [18] as a state-of-the art method. Its robustness with respect to identical values in the dataset makes it a good fit for CT and MRI datasets, which can contain large homogeneous regions.

Utilizing the transformations $Z_1(\sqrt{b_1})$ and $Z_2(b_2)$ of the sample skewness $\sqrt{b_1}$ and the sample kurtosis b_2, the K-squared test (Equation 3.10) is then defined as:

$$K^2 = Z_1(\sqrt{b_1})^2 + Z_2(b_2)^2 \tag{3.10}$$

K^2 is approximately χ^2-distributed with 2 degrees of freedom; we can test its null hypothesis by looking up the χ^2-distribution table. The entry for test level $1-\alpha = 0.999$ with 2 degrees of freedom in the χ^2-distribution table is 13.82. Therefore, the normality test will be passed if

$$K^2 < 13.82 \tag{3.11}$$

If the samples in the spherical neighborhood pass the normality test, it is necessary to further test whether they have the same distribution as that of the samples computed in the previous iteration. As done by Haidacher et al. [38], Welch's T-test [98] is used to compare the similarity of the sample distributions.

3.3 Results and Discussion

The statistical properties, the occlusion information and the size information are all precomputed on the GPU, and those volumes are then used in the interactive visualization stage to define the transfer function space. Users interact with an extended slice-based volume renderer implemented in OpenGL and Qt that supports combined 3D transfer functions to explore and generate final visualizations.

The scoring process is not part of our volume renderer and is conducted in MATLAB only once for a dataset. The input is a matrix where each of its columns is a property volume that is flattened into an 1D array. The correlation coefficient matrix is computed by the MATLAB function `corrcoef`, which uses Pearson's correlation. The MATLAB function `entropy`, which implements Equation 3.5, is applied to compute the entropy of each property by taking the histograms of the properties in a column of the input matrix. The number of bins of the histograms is determined by the number of bits of the data, e.g., an unsigned 8 bit volume has 256 bins. Finally, Equation 3.3 is evaluated for the corresponding row of the major property in matrix R and the normalized entropy vector $\frac{E(Y_i)}{\max E}$. The whole process takes about 10 seconds for each of the examples shown.

The following discussion compares 2D gradient magnitude, 2D statistical, 2D occlusion, 2D size with 3D combined statistical/occlusion, statistical/size, occlusion spectrum/size or statistical/(occlusion, size) transfer functions applied to a synthetic dataset and real-world datasets. The combined 3D transfer functions for each dataset were typically designed within 15 to 20 minutes, similar to the time required to design the traditional 2D transfer functions. The synthetic dataset models a filled shell encompassing varying sized spheres; the "Artifix" dataset has been retrieved from the OsiriX DICOM archive [72]. The back pack and the "Artifix" datasets are CT scans of a back pack and chest, respectively, "CerebrixCrop" is the T1 channel of an MRI scan of a brain.

The parameters used to create the transfer function spaces are chosen by trial-and-error on each dataset. The synthetic dataset and CT datasets are computed with confidence level 0.1, whereas the MRI dataset with a confidence level 0.001 when generating the statistical transfer function space. The radius is set to 40 for all

datasets when creating the occlusion volumes. The synthetic dataset is processed with a boundary preserving parameter $\eta = 1.0$ in order to overcome the noisiness, whereas all other datasets use $\eta = 0.005$ to preserve the boundary details. The size property computed for the MRI dataset is limited to the intensity range $[250, 500]$ in order to classify the tumor.

The transfer function combinations shown below are chosen by applying the algorithm described in Section 3.1.2 with varying parameter a. An exception to this are the results shown in Figure 3.2, where the extremely noisy nature makes Equation 3.3 ineffective.

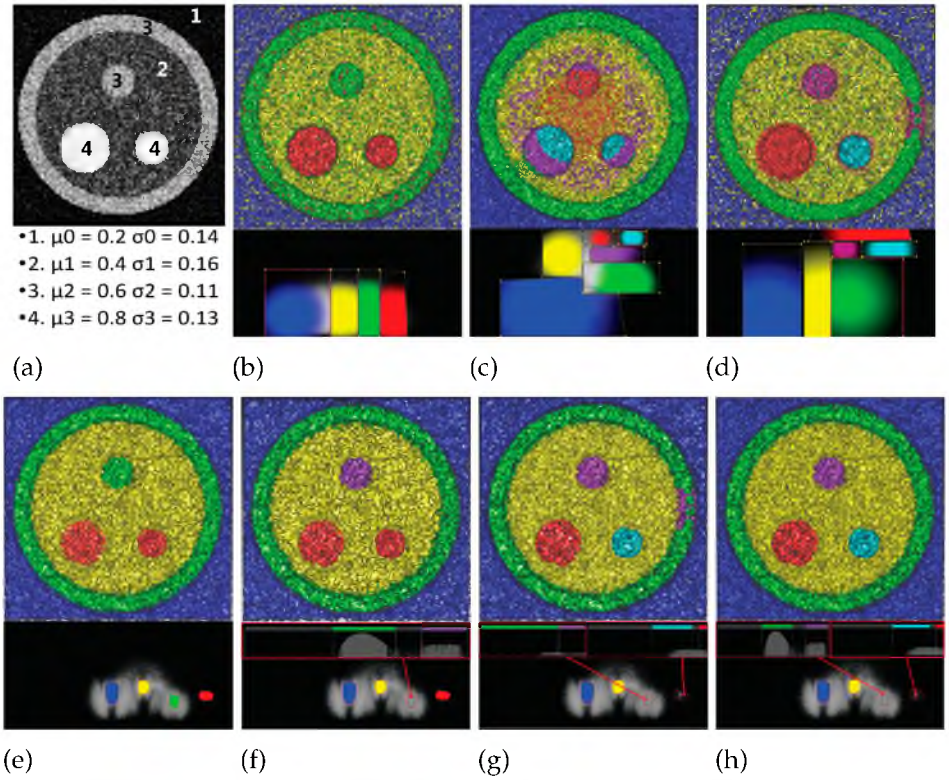


Figure 3.2. Comparisons of transfer function combinations for a synthetic dataset. The synthetic dataset was created as a mixture of overlapping Gaussian distributions with varying parameters to model a filled shell encompassing varying sized spheres as shown in a). The dataset has been classified, from left to right, using transfer functions (shown right below the rendered images) based on b) 2D gradient magnitude, c) 2D occlusion spectrum and d) 2D size-based transfer function, e) 2D statistical, f) combined statistical/occlusion transfer function, g) combined statistical/size transfer function and h) combined statistical/(occlusion, size) transfer function.

3.3.1 Synthetic Dataset

A synthetic dataset was created, as illustrated in Figure 3.2(a), in order to mimic a common scenario in real life medical datasets, such as chest CT scans or head MRI scans, where different structures overlap both spatially and in the scalar values. Often, the outer structures occlude with the inner ones, but they also can have different sizes. The synthetic dataset contains six different materials: the environment with $\mu_0 = 0.20, \sigma_0 = 0.14$, the middle hull with $\mu_1 = 0.40, \sigma_1 = 0.16$, the outer hull and the upper small inner sphere with $\mu_2 = 0.60, \sigma_2 = 0.11$, and both the remaining larger and smaller inner spheres have $\mu_3 = 0.80, \sigma_3 = 0.13$. In addition, low amplitude noise following a Gaussian distribution has been added across the whole domain to simulate noise introduced by acquiring a volumetric image with a scanner.

Various transfer functions have been applied to the synthetic dataset, as shown in Figure 3.2. Traditional 2D gradient magnitude-based transfer functions, as Figure 3.2(b) illustrates, suffer severely from the overlapping scalar values in the transfer function domain. There, features are indistinguishable due to noise, which makes it hard to separate features based on their gradient magnitude, as seen in the joint histogram in Figure 3.2(b).

Occlusion spectrum 2D transfer functions, shown in Figure 3.2(c), are able to separate the inner and outer structures based on their occlusion property as in the transfer function shown in Figure 3.2(c). The three inner spheres, however, cannot be separated clearly due to the similarity in their occlusion information as well as their scalar values. Also, the center of the inner yellow region overlaps with all spheres in the occlusion spectrum, thus causing misclassification.

The size-based 2D transfer function applied to the dataset (Figure 3.2(d)) separates the inner spheres from each other and the outer rings; however, there are classification artifacts at the top and right part of the green outer ring. The small sphere at the bottom right cannot be properly separated from the purple sphere, since they both overlap in their scalar values.

Statistical 2D transfer functions, as demonstrated in Figure 3.2(e), are able to separate the overlap in the (μ, σ) transfer function domain. It is thus possible to

classify them using different properties. However, both spheres at the lower center have the same statistical properties, and similarly, the outermost shell shares the statistical properties with the upper central sphere, yet they represent different structures.

Supplementing the statistical information with occlusion information, as shown in Figure 3.2(f), makes it possible to separate the inner purple sphere, compared to Figure 3.2(e). The transfer function in Figure 3.2(f) shows that the 1D occlusion histogram for the highlighted 2D widget can be used to separate the purple sphere with its low amount of highly occluded voxels from the green outer shell, which has a higher amount of less occluded voxels. However, the two red spheres at the bottom are not separated from each other.

On the other hand, supplementing the statistical information with size information, as shown in Figure 3.2(g), makes it possible to separate the two spheres at the bottom into the cyan small one and the larger red one, when compared to Figure 3.2(e). Noticeable are the purple artifacts in the green outer shell at the right side, since that region has a similar feature size compared to the purple sphere.

Figure 3.2(h) shows that occlusion and size information together are able to classify all the features of the dataset without ambiguity. The 1D transfer function associated with each widget in the 2D statistical transfer function space uses either size information *or* occlusion information to further classify the voxels selected in the statistical 2D transfer function domain, thus allowing the user to exploit the benefits of either method, while being able to interact with 1D and 2D transfer functions, instead of 3D or 4D transfer functions.

3.3.2 CT Scan of a Chest: "Artifix"

In the chest CT scan "Artifix" (Figure 3.3), both traditional 2D and combined 3D transfer functions were used to classify the lung (blue), bones (shades of gray), blood vessels (red), aorta (dark orange), kidney (brown) and the skin (transparent gray).

The gradient magnitude transfer function fails to correctly separate the blood vessels and the kidneys from the bones. Also noticeable is the relatively high amount of noise distributed across the volume.

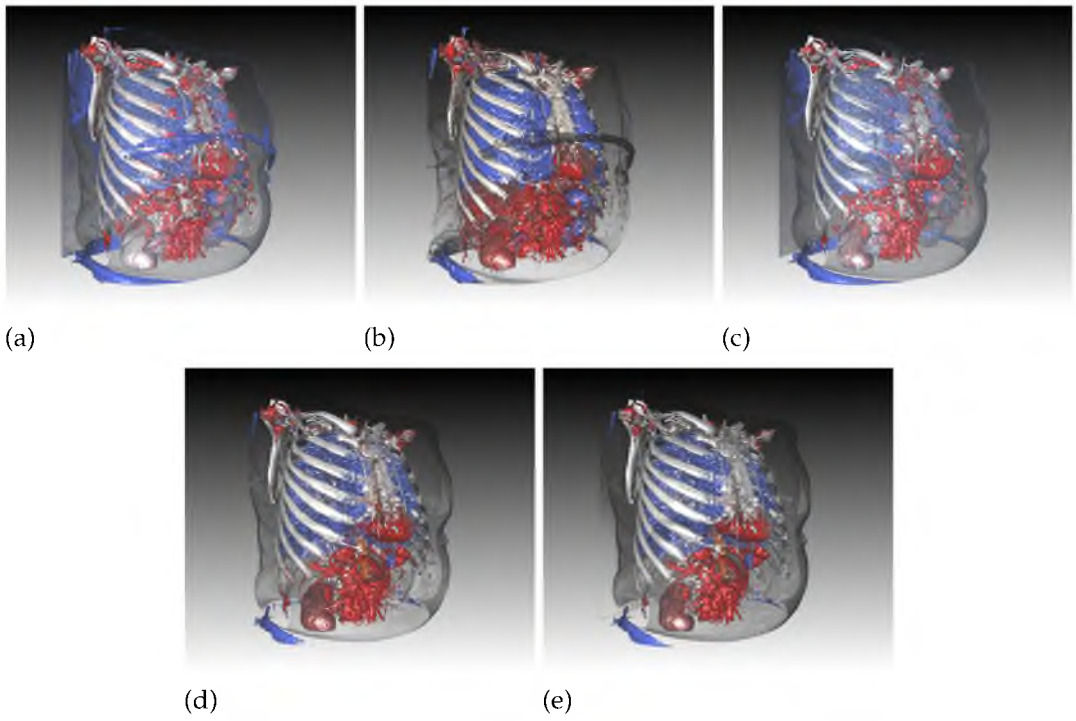


Figure 3.3. Transfer function combinations for a CT chest scan. The chest CT scan "Artifix" classified using transfer functions based on a) 2D gradient magnitude, b) 2D occlusion spectrum, c) 2D statistical, d) combined 3D gradient magnitude/occlusion transfer function and e) combined 3D statistical/occlusion transfer function.

The occlusion spectrum can be used to separate the kidney from the surrounding tissue. However, the aorta is similarly classified, since the aorta and the kidney are overlapping in the occlusion spectrum. Also, details of the lung are lost, since its tissue has similar occlusion values compared to the surrounding tissue, due to the intricacy and delicacy of the alveoli and bronchioles.

A statistical transfer function (Figure 3.3(c)) removes a noticable amount of that noise, but still leaves some areas, such as the front part of the ribs, and the kidney misclassified, since they are close with respect to their statistical properties.

Experimentation with the size-based transfer function as the associated transfer function space did not measurably improve the classification since the relative similarity of the scalar values in this CT scan mapped them to similar size values.

However, combining occlusion information with either a 2D gradient magnitude transfer function (Figure 3.3(d)) or a statistical transfer function (Figure 3.3(e))

increases the ability to correctly separate the kidneys from the aorta. The fine structures of the lung's surface are identifiable, since they have different statistical properties compared to the surrounding tissues. There are only slight differences between the combined transfer functions since they are similar without considering occlusion information.

3.3.3 CT Scan of a Back Pack

Figure 3.4 shows the CT scan of a back pack filled with liquids (in red, green, blue), a battery (in purple) and a box (in cyan) classified with various transfer functions.

The scoring with $a = 0.6$ conducted on the back pack dataset with scalar value chosen as the main property and gradient magnitude as the intrinsically associated secondary property results in:

$$s_{\mu,\sigma,\rho,S} = [-0.5208, -0.3882, 0.2165, -0.1595]$$

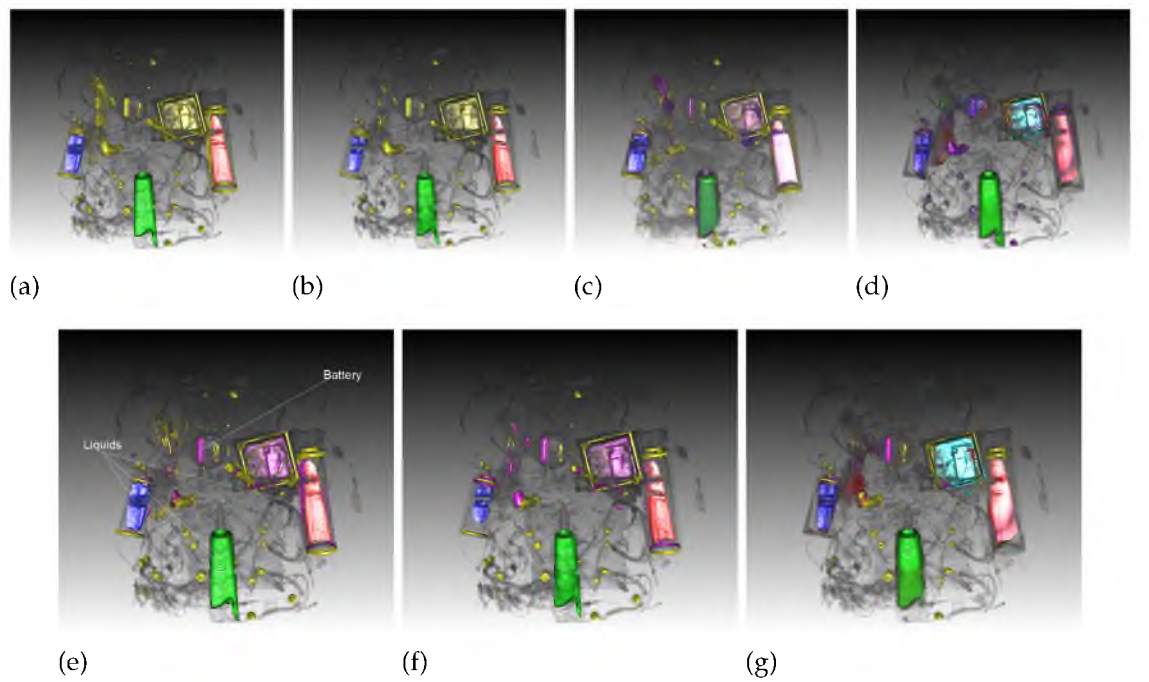


Figure 3.4. Transfer function combinations for a back pack CT scan. The back pack CT scan classified using transfer functions based on a) 2D gradient magnitude, b) 2D statistical, c) 2D size, d) 2D occlusion, e) combined 3D gradient magnitude /occlusion transfer function, f) combined 3D statistical/size transfer function and g) combined occlusion spectrum and size transfer function.

suggests that the occlusion volume ρ and size volume S should be considered for tertiary attributes. Changing the main property to mean volume with standard deviation volume as the secondary attribute gives the scoring for the rest of the properties:

$$s_{x,|\nabla x|,\rho,S} = [-0.4847, -0.2140, 0.2153, -0.1679]$$

also hints to us that the occlusion volume ρ and size volume S should be used as tertiary attributes.

The 2D transfer functions separate the different liquids to varying degrees, but they fail to identify the battery properly. Other features, such as the wires or the small circular shapes, are mapped to the same color yellow (Figures 3.4(a), 3.4(b)), or the same feature is mapped to different colors (Figures 3.4(c), 3.4(d)). Notable is the 2D occlusion transfer function, which allows the extraction of the cyan box but classifies the liquids with less specificity.

Adding occlusion as the third axis did not yield meaningful results, since the dataset itself has many features that are similarly occluded by the clothing articles (showing in transparent gray) inside the back pack, thus reducing the separability in the occlusion channel.

Utilizing a size transfer function as the third axis allows the clear separation of the battery (purple color). The 3D occlusion spectrum/size transfer function (Figure 3.4(g)) is additionally able to visualize the cyan box, which is difficult to do using gradient magnitude (Figure 3.4(e)) and statistical information (Figure 3.4(f)) as the 2D transfer function domain. However, all the 3D transfer functions have problems in classifying the wires as features both connected spatially and with respect to their colors, suggesting further investigations of alternative volumetric attributes as the third axis.

3.3.4 MRI Scan of a Brain: "CerebrixCrop"

MRI datasets, occurring in clinical and research studies where separating the brain from the surrounding tissue is of particular interest, are typically challenging to classify, since they often contain ubiquitous noise [27]. Figure 3.5 shows such a dataset containing a tumor in the center of the brain. Transfer functions are

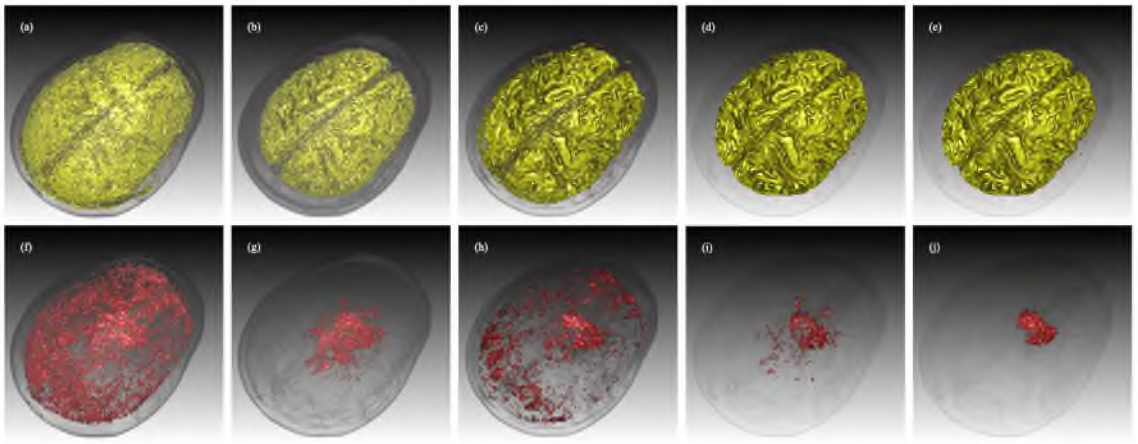


Figure 3.5. Transfer function combinations for an MRI brain scan data. The "CerebrixCrop" MRI dataset shown with focus on the brain tissue shown as yellow (top row) and a tumor shown in red (bottom row). The following transfer functions were applied: a,f) 2D gradient magnitude, b,g) 2D occlusion, c,h) 2D statistical, d,i) 3D statistical/occlusion, e,j) 3D statistical/(occlusion,size).

applied to classify the brain tissue (in yellow) and the fluid inside the tumor (in red). Note that although both features can be shown simultaneously by setting transparency of the brain, we set the brain to be completely transparent in the second row of images for clear visualizations of the tumor.

We apply the scoring process with $a = 0.6$ to the MRI dataset: set scalar value as primary and gradient magnitude as secondary yields:

$$s_{\mu,\sigma,\rho,S} = [-0.3188, -0.1247, 0.1674, -0.0224]$$

meaning that the occlusion volume ρ and size volume S once again should be considered for tertiary attributes. Substituting the main attribute with the mean value with the standard deviation as the secondary attribute gives

$$s_{x,|\nabla x|,\rho,S} = [-0.2841, -0.0591, 0.1505, -0.0298]$$

and leads us to the same decision.

Gradient magnitude-based 2D transfer functions (Figure 3.5(a)) fail to properly separate the brain from the skin, since they both share similar ranges of scalar values and gradient magnitudes.

Figure 3.5(f) demonstrates the inability for the gradient magnitude based 2D transfer functions to clearly pull out the tumor, since similar scalar values and gradient magnitudes appear universally across the dataset.

The occlusion spectrum (Figures 3.5(b) and 3.5(g) helps to better separate the brain from its surrounding tissues as well as remove noise with scalar values similar to the tumor. However, the surface of the brain tissue is still incorrectly classified and a large amount of noise still appears around the tumor due to similar occlusion values in these regions.

Statistical transfer functions (Figures 3.5(c) and 3.5(h)) significantly smooth the dataset, making the creases and recesses of the brain tissue clearly show up, however, noise that heavily affects the visual quality is still seen across the dataset, especially in Figure 3.5(h).

Combining the occlusion information with statistical information, as shown in Figures 3.5(d) and 3.5(i), classifies the brain tissue properly, but fails to clearly extract the tumor.

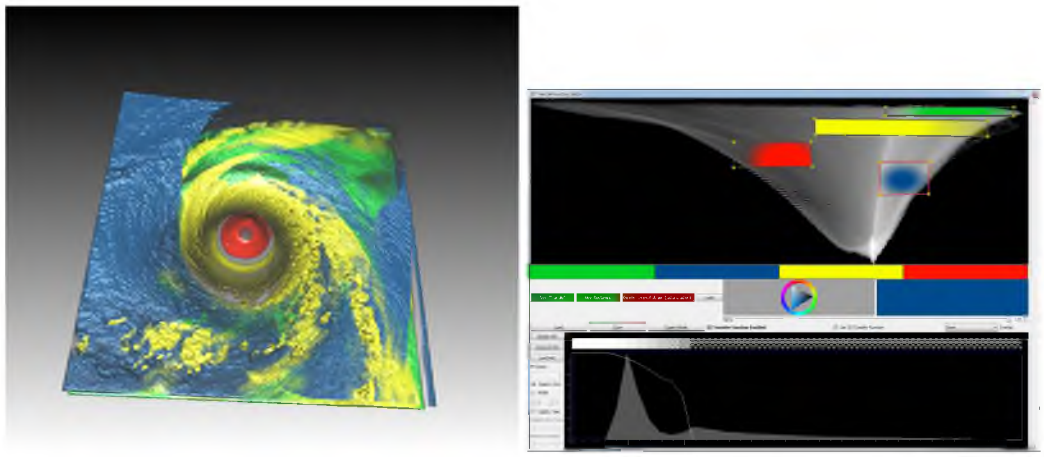
However, a transfer function combination with two tertiary attributes, as shown (Figures 3.5(e) and 3.5(j)), clearly separates both the brain tissue and the tumor. The statistical attributes are used as the primary and secondary attributes, and the occlusion and size information are used as the tertiary attributes.

The widget that classifies the yellow brain tissue uses the occlusion attribute to further classify it with the associated 1D transfer function; however the widget classifying the red tumor uses the size attribute instead to further remove the noise via its associated 1D transfer function.

3.3.5 Multivariate Dataset: Hurricane Isabel

One time step (time step 30) of the VisContest 2004 Hurricane Isabel [47] multivariate dataset is used to demonstrate the generality of our method. The dataset is a simulation of a hurricane from the National Center for Atmospheric Research in the United States. The original dataset contains 100 time steps and each with 12 attributes. Many of these attributes, however, are redundant or contain little amount of information.

The three most salient attributes are selected, namely pressure, temperature and a water vapor mixing ratio measurement, QVAPOR, by evaluating the entropy of each attribute. The 2D transfer function domain is pressure and temperature. We use QVAPOR as the associated 1D transfer function. These attributes are then used to classify significant features in meteorology, including hurricane eye and spiral arms. As shown in Figure 3.6, each colored widget in the 2D domain uses a different QVAPOR 1D transfer function for classification and the classified volume are visualized using volume rendering. The eye of the hurricane (shown in red) has a lower pressure but higher temperature than the blue outer bands and a lower temperature compared to the yellow and green spiraling bands. The QVAPOR attribute allows us to see the spiraling bands with fine details in the dataset.



(a) Hurricane Isabel Visualization (b) Hurricane Isabel UI

Figure 3.6. Classifying a multivariate dataset using the combined transfer function space. Visualization of the multivariate Hurricane Isabel dataset using pressure and temperature in the 2D transfer function with different 1D transfer functions using QVAPOR for each 2D transfer function widget shown in different colors.

CHAPTER 4

TRANSFER FUNCTION DESIGN BASED ON USER-SELECTED SAMPLES FOR INTUITIVE MULTIVARIATE VOLUME EXPLORATION

Multivariate volumetric datasets are important to both science and medicine. We propose a transfer function (TF) design approach based on user-selected samples in the spatial domain to make multivariate volumetric data visualization more accessible for domain users. Specifically, the user starts the visualization by probing features of interest on slices and the data values are instantly queried by user selection. The queried sample values are then used to automatically and robustly generate high-dimensional transfer functions (HDTFs) via kernel density estimation (KDE). Alternatively, 2D Gaussian TFs can be automatically generated in the dimensionality reduced space using these samples. With the extracted features rendered in the volume rendering view, the user can further refine these features using segmentation brushes. Interactivity is achieved in our system and different views are tightly linked. Use cases show that our system has been successfully applied for simulation and complicated seismic datasets.

4.1 Method Overview

The workflow of our proposed method as shown in Figure 4.1 is comprised of three major stages: (A) *data probing*, (B) *qualitative analysis* and (C) *optional feature refinement*.

Data probing is the process where the user discovers regions of interest by examining multivariate data slices. The regions of interest can be conveniently selected using a lasso tool or a "magic wand" tool. Once the regions of interest are selected, a simple, yet efficient, voxel query operation that inquires the multivariate data values is performed. The user then performs a qualitative analysis, i.e.,

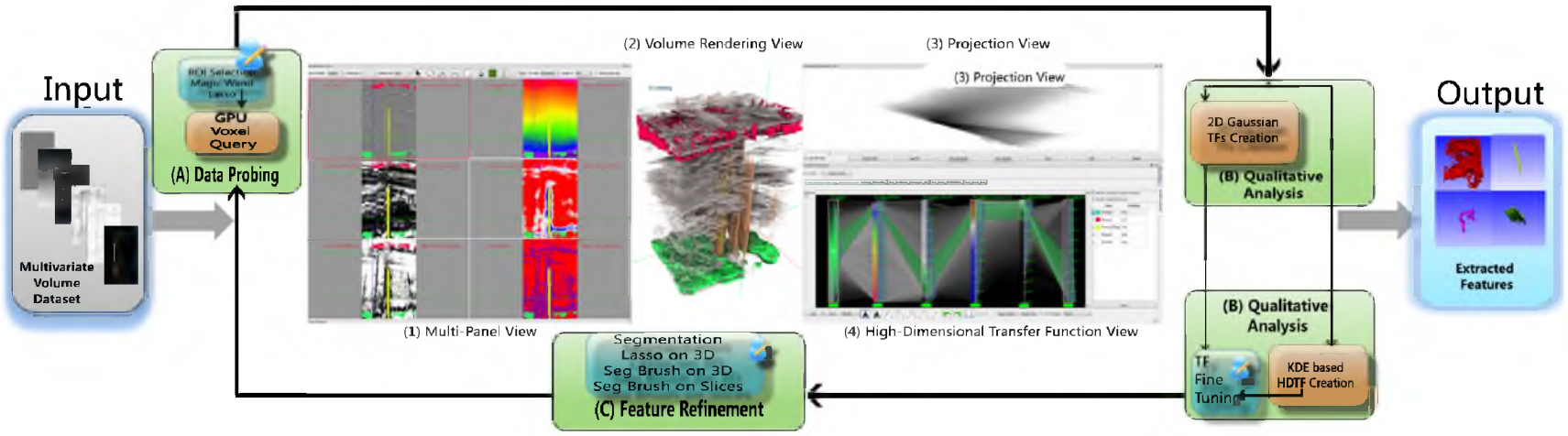


Figure 4.1. The user interface and the work flow of the system implementing our proposed method. Four closely linked views are shown and labeled, namely: (1) multipanel view, (2) volume rendering view, (3) projection view and (4) high-dimensional transfer function view. Three stages: (A) data probing, (B) qualitative analysis and (C) optional feature refinement comprise our work flow. With the proposed method and user interface, domain users are able to explore and extract meaningful features in highly complex multivariate dataset, e.g., the 3D seismic survey shown above.

extracting and rendering volumetric features by means of designing HDTFs or 2D TFs on dimensionality reduced spaces. KDE is utilized to automatically generate the HDTFs and to robustly discard outliers from the queried samples. Similar to previous methods [1, 5, 31], the transfer functions can also be directly modified in the high dimensional transfer function editor. The high dimensional data space as well as the transfer functions are represented by parallel coordinate plots and pairwise scatter plots. In addition, automated 2D Gaussian TFs on the projection view offer a simpler alternative for more distinct features. The HDTFs can then be fine-tuned directly in a PCP-based HDTF editor while the 2D Gaussian TFs can be manipulated by 2D Gaussian TF widgets. On many occasions, however, different features share similar data values and thus an optional feature refinement stage is introduced to refine the features classified by the TFs. Features are refined by the user via segmentation brushes or lassos that are applied directly on the volume rendering view or the multipanel view. After several iterations of the three stages, the user can choose to output the classified result as labeled volume for further processing. We have implemented an interactive multivariate volume visualization system based on the proposed method that has been implemented to allow domain users to extract refined features in very complicated multivariate volume datasets more intuitively.

4.2 Voxel Query and PCP Generation

Our proposed method is based on user-selected multivariate voxel samples through interactive selection, which requires efficient voxel query. The multivariate values of the queried samples should be immediately presented to the user by means of PCPs, and so a fast PCP generation method is needed.

4.2.1 GPU-based Voxel Query via Conditional Histogram Computation

Voxel query can be accelerated by spatial hierarchy structures that group similar neighboring voxels into nodes, e.g., an octree structure adopted by Guo et al. [31]. However, Knoll et al. [58] report that, "Conversely, volumes with uniformly high variance yield little consolidation; due to the overhead of the octree hierarchy they

could potentially occupy greater space than the original 3D array." Our initial experiment on the seismic data with the code from [58] agrees with this statement. We therefore propose to efficiently conduct the voxel query by computing sets of joint conditional histograms via a simple GPU-based volume traversal. A joint conditional histogram $jch(a,b)_f$ of two attributes a and b is a 2D histogram showing the joint distribution of attribute values Y_a and Y_b of voxels V whose evaluated result from a certain boolean function $f(Y(\vec{V}))$ ($Y(\vec{V})$ being the attribute values of V) is true. If f is always true, the joint conditional histogram degenerates to an unconditional joint histogram. Note that the values of user selected samples are queried via an unconditional joint histogram computation over the user-selected region on the given slice.

For a multivariate volume of N attributes, given an N-dimensional TF as the condition, a set of $N-1$ joint conditional histograms can be computed to record the query results. The values of the joint conditional histograms are accumulated by first evaluating the N-dimensional TF for all voxels in the volume, and then transforming the voxels that have positive opacities from the TF into bins in the conditional histogram space, and finally incrementing the joint conditional histogram count at those bins. Specifically, given a voxel v_X of N attributes $Y_1, Y_2, ..., Y_N$ (to be concise, we use y_i to denote the attribute value $Y_i(v_X)$) located at 3D position X in the spatial domain, and an N-dimensional TF `TF`.

$$\begin{aligned} v_X \rightarrow \{&(y_1,y_2),(y_2,y_3),...,(y_{N-1},y_N)\} \\ \text{where } &\text{TF}(y_1,y_2,\ldots,y_N).a > 0 \end{aligned} \tag{4.1}$$

$(y_1,y_2),(y_2,y_3),...,(y_{N-1},y_N)$ being the bins of joint conditional histograms

$$jch(Y_1,Y_2), jch(Y_2,Y_3),..., jch(Y_{N-1},Y_N),$$

respectively.

Equation 4.1 and the accumulation of the conditional histograms, which are stored aggregately as a 2D texture array of $N-1$ slices, can be easily implemented on the GPU via geometry shader and `ADD` blending or read-write textures with atomic operations that are supported on recent GPUs.

4.2.2 Parallel Coordinate Plots Generation

As proposed in [70], Figure 4.2 shows that each nonzero pixel $P(i, j)$ in the joint histogram of attribute x and y yields a quad starting at the position of i on PC axis x and ending at the position of j on PC axis y.

The highly parallel process can be implemented on the GPU using geometry shader and transform feedback buffers. The algorithm loops through all pairs of conditional histograms after setting up the transform feedback buffer for recording the resulting geometry. In each iteration, a regular grid of the same size of a slice of the input conditional histogram texture tex_{cond} is drawn and a geometry shader generates a colored quad for each vertex whose tex_{cond} value is not 0. The dynamic range of the data values is usually high and thus the ratio of the natural logarithm of the data value versus natural logarithm of the total voxel number is computed and then modulated with the input color $C_0(i, j)$ at grid position (i, j) to give the final color $C(i, j)$.

$$C(i, j) = C_0(i, j)\frac{\log(v(i, j))}{\log(\sum v)} \tag{4.2}$$

Finally, all quads are stored in the transform feedback buffer, and they can be rendered directly from the transform feedback buffer without being read back to the CPU.

4.3 Transfer Function Generation from User-Selected Samples

In this section, the actual TF generation method will be explained. Section 4.3.1 introduces the method for interactive voxel sample selection, Section 4.3.2 dis-

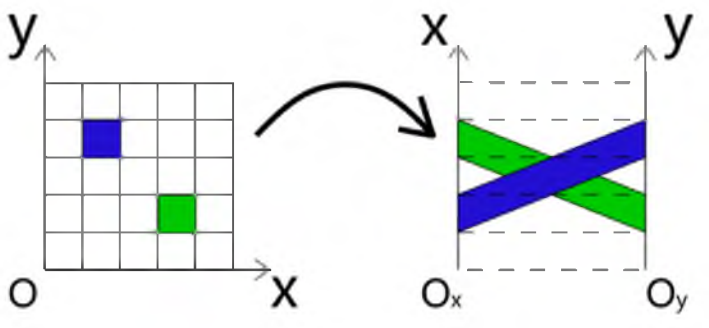


Figure 4.2. Generating a PCP from a joint histogram.

cusses the KDE-based HDTF generation method and Section 4.3.3 presents details on the automated 2D Gaussian TF on the dimensionality reduced space.

4.3.1 Sample Selection in the Multipanel View

The user can interactively select an arbitrary region of interest in any attribute by either drawing a lasso or using the magic wand tool. The lasso tool is a simple free hand drawing tool that allows the user to select regions by manually drawing over the boundary of a feature. Although very flexible, the user has to be very careful when drawing on the boundary using the lasso tool.

To alleviate the difficulty of perfectly drawing over the boundary of a feature, a more intuitive and easier to use magic wand tool is introduced. The magic wand tool is essentially a 2D segmentation tool based on Perona and Malik's anisotropic diffusion [76]. Equation 4.3 describes the diffusion equation where $S(t,x,y)$ is the number of seeds at position (x,y) at time t, with $V(t,x,y)$ being the intensity of the chosen attribute at the same point, $|\nabla V(t,x,y)|$ is its gradient magnitude, and $g(s)$ a conductivity term.

$$\frac{\partial S(t,x,y)}{\partial t} = div(g(|\nabla V(t,x,y)|)\nabla S(t,x,y)) \tag{4.3}$$
$$\text{where } g(s) = v \cdot \exp^{\frac{-s^2}{K^2}}$$

Parameter K governs how fast $g(s)$ goes to zero for high gradients, regular term v is chosen as 1 and normalization term h is set to $\frac{1}{n+1}$ for numerical stability, n being the number of neighbors of a pixel, which is 8 in our case. Equation 4.3 can be solved numerically using the finite difference method with a given iteration number T. The iteration number T, parameter K and seeding brush size are user controllable. Figure 4.3 shows the panel view of a six-attribute seismic volume dataset where attributes are co-rendered with the seismic amplitude volume. Note that a user drawn magic wand in dark blue highlights a potential salt dome structure.

4.3.2 Kernel Density Estimation-based Transfer Function Generation

We would like to generate HDTFs from the samples selected using method described in Section 4.3.1. To reduce the computational complexity, we separate

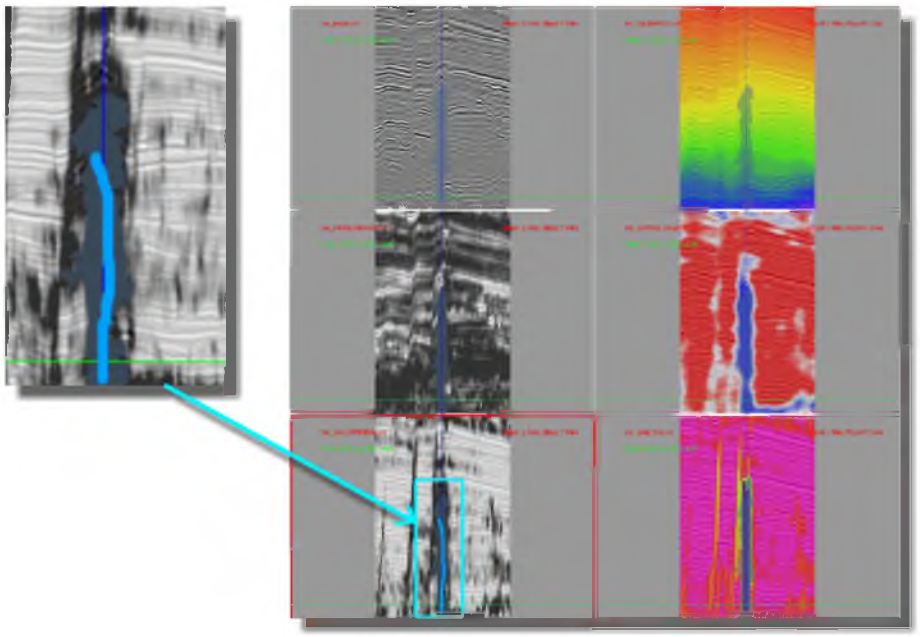


Figure 4.3. Sample selection in the multipanel view. The user draws on a salt dome (stroke shown in light blue) over the fifth attribute in the panel view, resulting in the dark blue region of selection.

the N-dimensional value space into $N-1$ 2D value spaces, i.e., a $2D+2D+\cdots+2D$ ($N-1$ of $2D$) space. A naive approach is to generate a TF by taking the convex hull of these $2D$ sample points. Although useful when the user intends to select exact sample points, it is conceivable that the outliers in the samples can greatly bias the generated TF and result in unwanted regions selected in the value space.

Figure 4.4(a) clearly demonstrates such a situation where a red 2D TF widget is generated as the convex hull of the sample points with the red boundary. Also notable is that the color gradient of the TF widget is arbitrarily defined by the user that may not follow the underlying distribution of data.

Kernel density estimation (KDE) [87] seen in Equation 4.4 is a nonparametric method for estimating the density function $f_h(x)$ at location x of an arbitrary dimensional domain Ω with given samples $\{x_i\}, i \in \{1,2,3,\ldots,n\}$.

$$f_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x-x_i) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x-x_i}{h}), x, x_i \in \Omega \tag{4.4}$$

where $K(x)$ is the kernel function and h is the bandwidth. Thanks to the separation of the value space, instead of computing the KDE for Ω of N dimension, we

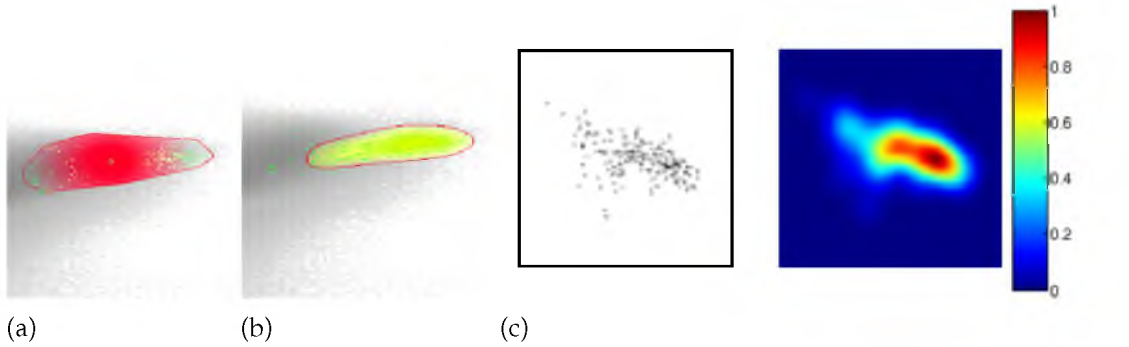


Figure 4.4. Transfer function generated from user-selected sample points. User-selected sample points (shown in green) over a joint histogram. TF widget generated from the samples as (a) convex hull and (b) KDE. In (c): a point cloud (left) and its KDE result color coded with a "jet" color map.

compute $N-1$ KDE for Ω in 2D spaces. In our case, each Ω is set to the same size of the $2D$ joint histogram, which is typically 256×256.

An empirical optimal bandwidth estimator is suggested in [87], which can be extended to 2D:

$$h = 1.06 \sqrt{\det \Sigma} \cdot n^{-\frac{1}{5}} \tag{4.5}$$

where $\det \Sigma$ is the determinant of the $2D$ covariance matrix Σ of current attribute pairs. The kernel function $K(x)$ we used is the $2D$ Gaussian kernel:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\|x\|^2}{2}} \tag{4.6}$$

With the Gaussian kernel, each sample x_i contributes to the estimate in accordance with its distance from x. Therefore, in the region near the intended samples more short distanced samples are contributing to $f_h(x)$ compared to the region near the outliers. As a result, the density value $f_h(x)$ around the outliers is lower than that of the intended samples. Figure 4.4(c) shows the density function generated by the KDE method of the given samples with the above settings. This result verifies our expectation that the outliers have a lower density than the intended sample regions. We can discard the outliers by setting a threshold for the density value $f_h(x)$. Figure 4.4(b) shows the yellow TF widget generated by KDE with a density threshold of 0.15. Noticeable is that the outliers are excluded from the TF widget and the smooth color gradient that actually follows the underlying density. The

resulting TF can be represented by a set of 2D TFs or a PCP created using the method described in Section 4.2.2.

In the presence of multiple HDTFs, ambiguity could arise: different HDTFs can cover the same regions of certain 2D attribute pairs. To differentiate the HDTFs, a unique ID is specified to each HDTF and an ID map of the same size of the $N-1$ 2D TF space is created by conducting bitwise `OR` for all HDTFs on each 2D attribute pair. The ID map is later decoded in the volume rendering shader to correctly select voxels.

4.3.3 Automated Gaussian Transfer Functions on Dimensionality Reduced Space

Dimensional reduction is another popular method for visualizing high-dimensional data due to its ability to intrinsically generate visual representations that are easy to understand and interact with. Instances in an m-dimensional Cartesian space are projected into a lower p-dimensional visual space with preservation of the distances between instances as much as possible. In other words, voxels with similar m-dimensional attribute values are projected to be near each other in the p-dimensional space. With a projected visual space of $p = 2$, the user is able to better identify features by doing visual classification using a 2D TF widget, and moreover, automated clustering methods can be applied for classification. In our proposed method, the high-dimensional value space is projected into a 2D space using Fastmap [24] and then Gaussian TFs are generated via expectation maximization optimization with Gaussian mixture model. The user can choose to use either the 2D Gaussian TF or the HDTF for each feature by switching a button on the user interface. The 2D Gaussian TFs are preferred for more convenient extraction of several distinct features at the same time, whereas the HDTFs are better for features that have subtle differences in the HD value domain.

We employ Fastmap [24] as the dimensional reduction technique since it is fast, stable and easy to implement. Fastmap is a recursive algorithm for multi-dimensional projection with an $O(N)$ time complexity. Given target dimension k, a distance function $D()$ and object array O contains N objects of m dimension, the algorithm `FastMap` computes the k-dimensional projected image X from the N

objects. The algorithm is summarized in Algorithm 4.1

Assuming that all attributes we are handling are continuous measurements, the dimensionality reduced 2D value space can be modeled by a Gaussian mixture model (GMM). GMM models point clouds by assigning each cluster a Gaussian distribution. For a point x in the 2D value space, a Gaussian distribution is shown in Equation 4.7 with mean value μ being a 2D vector and covariance matrix Σ as a 2×2 matrix.

$$N(x|\mu,\Sigma)=\frac{1}{2\pi|\Sigma|^{1/2}}e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \tag{4.7}$$

Therefore, for a GMM with k components, the distribution of the 2D value space can be written as

$$p(x|\theta)=\sum_{j=1}^{k}\alpha^j N(x|\mu^j,\Sigma^j) \tag{4.8}$$

where θ is the parameter set of the k-component GMM $\{\alpha^j,\mu^j,\Sigma^j\}_{j=1}^k$, and α^j is the prior probability of the jth Gaussian distribution. The optimal $\hat{\theta}$ can be found as θ that maximizes the likelihood of $p(X|\theta)$

$$\hat{\theta}=\arg\max p(X|\theta)=\arg\max\prod_{i=1}^{n}p(x^i|\theta) \tag{4.9}$$

where n is the number of input points. Equation 4.9 can be solved by the expectation maximization (EM) algorithm [4]. Given an initial setup of θ, the EM

Algorithm 4.1 FastMap($k,D(),O$)

---

if $k\le 0$ **then**
  return
else
  $col=col+1$ (col is initialized to 0)
end if
Choose and record the pair of pivot objects O_a,O_b.
Project objects on line (O_a,O_b) using the cosine law:
$X[i,col]=x_i=\frac{D(O_a,O_i)^2+D(O_a,O_b)^2-D(O_b,O_i)^2}{2D(O_a,O_b)}, i\in\{0,1,2,\ldots,N-1\}$
Call FastMap($k-1,D'(),O$).
Where
$D'(O_i',O_j')^2=D(O_i,O_j)^2-(x_i-x_j)^2, i,j\in\{0,1,2,\ldots,N-1\}$

---

algorithm iterates between two steps: expectation step (E step) and maximization step (M step) until the log likelihood

$$\ln p(X|\theta) = \log(\prod_{i=1}^{n} p(x^i|\theta)) = \sum_{i=1}^{n}\{\sum_{j=1}^{k} \alpha^j N(x^i|\mu^j, \Sigma^j)\}$$

converges. We initialize the EM algorithm using the K-means algorithm [40], which quickly gives a reasonable estimation of θ. With an initialization of k mean values $\{\mu^j\}_{j=1}^{k}$, K-means algorithm iteratively refines $\{\mu^j\}_{j=1}^{k}$ until convergence through assignment and update steps. The assignment step assigns each sample to the cluster with the closest mean, and the update step calculates the new means to be the centroid of each cluster. In our case, the initial means are k random samples in the input dimensional reduced 2D point cloud. Once the K-means algorithm terminates, $\{\Sigma^j\}_{j=1}^{k}$ can be easily computed with the result means, and prior probabilities $\{\alpha^j\}_{j=1}^{k}$ is given by the proportion of total samples inside each cluster.

We use a modified TF generation scheme as in [96] but ours differs in that 1) the value space we use is the 2D dimensionality reduced space of high-dimensional attribute compared to the 2D intensity versus gradient magnitude space as in [96], and 2) we use the user-selected samples as the input point clouds, whereas they use all voxels in a volume.

Given some user-provided sample data points and a class number k (which is set to 3 by default from our experiments), the EM algorithm computes the Gaussian distribution parameters $\hat{\theta}$. Each Gaussian distribution is managed by a Gaussian TF widget with a user-defined color C and opacity function α of location x:

$$\alpha = \alpha_{\max} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{4.10}$$

The Gaussian TF widget is centered at the mean value μ of the Gaussian distribution and its boundary is generated by transforming a unit circle with the square root matrix $\Sigma^{1/2}$ of covariance matrix Σ. $\Sigma^{1/2}$ is calculated via eigen decomposition of Σ:

$$\Sigma = VDV^{-1} \tag{4.11}$$

$$\Sigma^{1/2} = VD^{1/2}V^{-1} \tag{4.12}$$

where D is a diagonal matrix holding the eigenvalues and V contains the eigenvectors as columns. V is an orthogonal matrix, i.e., $V^{-1} = V^T$, since Σ is symmetric. The eigenvalues σ_1, σ_2 are the radii of the principal axes of the ellipse, whereas the eigenvectors a, b are the unit vectors of the principal axes.

Transformations of the Gaussian widgets, i.e., translation, rotation and scaling, can be achieved using the eigenvalues and eigenvectors. The translation is done by shifting the μ with an offset $\Delta\mu$ given by user dragging. The rotation of the widget is achieved by rotating the eigenvectors in V with an angle β. Finally, multiplying the eigenvalues σ_1, σ_2 with a scaling factor (s_a, s_b) results in the scaling of the widget.

4.4 Feature Refinement in the Spatial Domain

The feature refinement stage is introduced to allow the user to directly manipulate the features in the spatial domain. Various refinement tools have been implemented to handle different situations. All tools support three refinement modes: new, add and remove.

4.4.1 Screen Space Brush in the 3D View.

The tool as seen in Figure 4.5(a) allows the user to draw strokes on the 3D view screen to set seeds in the visualization results, and then a GPU-based region growing is conducted to set the connected voxels to a given tag number. The seeding location is determined by casting rays from brush strokes on the image

(a) (b) (c)

Figure 4.5. Feature refinement tools. Three feature refinement tools are included in our method: (a) 3D brush, (b) 3D lasso and (c) 2D brush.

plane to the volume extracted by current TFs. A voxel along the ray is seeded when its opacity is greater than a user-defined threshold.

4.4.2 Screen Space Lasso in the 3D View.

Alternatively, the user can directly indicate features of interest on the 3D view using a lasso as shown in Figure 4.5(b). A lasso is a simple tool that selects all voxels from the TF extracted volume that are inside the back projected volume of the screen space lasso covered area.

4.4.3 Refinement Brush in the Panel View.

The refinement can also be done by seeding on the panel view via drawing strokes (Figure 4.5(c)), and this is useful when the features of interest are occluded in the 3D view or readily visible in a slice. A morphological closing, i.e., dilate the volume by one voxel and then erode the volume by one voxel, is performed after refinement in order to fill small holes and bridge tiny gaps. Note that all refined feature groups are managed in the group manager in the HDTF editor introduced in Section 4.6.2, and thus similar to TF groups, their colors can be changed, they can be deleted and their visibility can be toggled.

4.5 Rendering

We employ the directional occlusion shading (DOS) [84], which is an efficient approximation to ambient occlusion as the rendering technique because the DOS is gradient-free and provides the user more insights into the dataset than local shading models as shown on seismic datasets [73]. A user study conducted by [61] shows that DOS outperforms other state-of-the-art shading techniques in relative depth and size perception correctness. Hardware supported trilinear interpolation cannot be used for tag volume rendering because false tag values will be generated. Instead, nearest neighbor sampling has to be used to correctly render the tag volume. However, a simple use of nearest neighbor sampling yields blocky looking results because of the voxel level filtering. Instead, using a manual trilinear 0-1 interpolation gives pixel level filtering. From our observations, the cases where multiple tags appear in a single 8 voxel neighborhood rarely occur

and so a simplified method of [34] is utilized. The largest tag value in the eight neighboring voxels around current pixel is mapped to 1 and all others to 0 and then a trilinear interpolation is conducted on these 0/1 values. The interpolated result is then compared against 0.5. If greater, the final tag value of the pixel is set to the pixel's nearest neighboring voxel's tag value, otherwise the tag value is set to 0.

4.6 User Interface

The user interface of our system is seen in Figure 4.1 where a multipanel slice view for data probing is shown to the left (1), an interactive 3D view that shows volume rendering results and allows post feature manipulation is seen in the middle (2), a projection view shown to the upper right (3) and a high-dimensional transfer function view appears to its bottom (4). These four views are tightly linked and any updates in one view will be reflected in others.

4.6.1 Multipanel Viewer

We have developed a multipanel view that shows all attributes of a slice by placing attributes into individual panels as seen in the left part of Figure 4.1 as well as in Figure 4.3. The multipanel viewer synchronizes user interactions across all attribute views, including: mouse positioning, panning, zooming, scrolling and aspect changing. To enhance the perception of attributes, each attribute can have a specifically designed color map that highlights features of interest. In order to better use the dynamic range of the color maps, the contrast of the attributes can be conveniently changed using the mouse wheel. Furthermore, a background volume can be co-rendered with the current attribute volume using transparency. This rendering mode is especially helpful for seismic volumes as our collaborating geologists suggest that it provides more insight into the attributes when the seismic amplitude volume is co-rendered as a context.

4.6.2 HDTF Editor

The user can interact with the HDTF editor to manually modify the HDTFs. Figure 4.6 shows the HDTF editor where the PCP axes reorder button and attribute-

wise control panel can be seen on the top, the PCP TF editor is seen in the upper left, a group manager is shown to its right and the pairwise TF editor is shown in the lower part.

The attribute-wise control buttons allow the user to specify a color map, toggle sampling between linear and nearest neighbor, and toggle lock/unlock for each attribute. A locked attribute is essentially an attribute with its entire value range used in TFs. In other words, it can be visualized in the panel view but is not contributing to classification. This setting is useful since not all attributes provide positive assistance in the extraction of specific features and this knowledge is usually not known beforehand. Also, there are cases when one needs an attribute to provide only context for data probing, e.g., the seismic amplitude attribute, which will be discussed in Section 4.7. The group manager manages all TF and segment groups. One is able to toggle the visibility or remove an individual or a batch of groups conveniently.

As seen in Figure 4.6, the PCP axes are co-rendered with the 1D histograms of attributes shown to the right and color map to the left. Since the color map is synchronized with the one that appears in the panel view, the user is able to instantly know how to set the TF widgets. The user interacts directly with the parallel coordinate axes to design an HDTF using one of the three interaction widgets, namely, *brush widget*, *tent widget* and *Gaussian widget*. The brush widget enables the user to arbitrarily interact with the TF domain. Tent and Gaussian widgets are essentially sets of 1D TF widgets residing on each attribute axis of the HDTF domain, and they differ only in their shape of the opacity gradient. In addition to the PCP TF editor, a pairwise 2D TF editor is used to aid the exploration of pairwise features. The pairwise 2D TF editor allows the user to interact with $N-1$ 2D TF space to fine tune the HDTFs to match irregular shaped features in specific pairs of attributes using 2D rectangle, triangle or lasso widgets.

4.6.3 Projection Viewer

A projection viewer has been implemented in our proposed system by combining the Fastmap dimensional reduction technique with GMM 2D Gaussian TFs. The projection viewer extends the traditional 2D TF editor with Gaussian

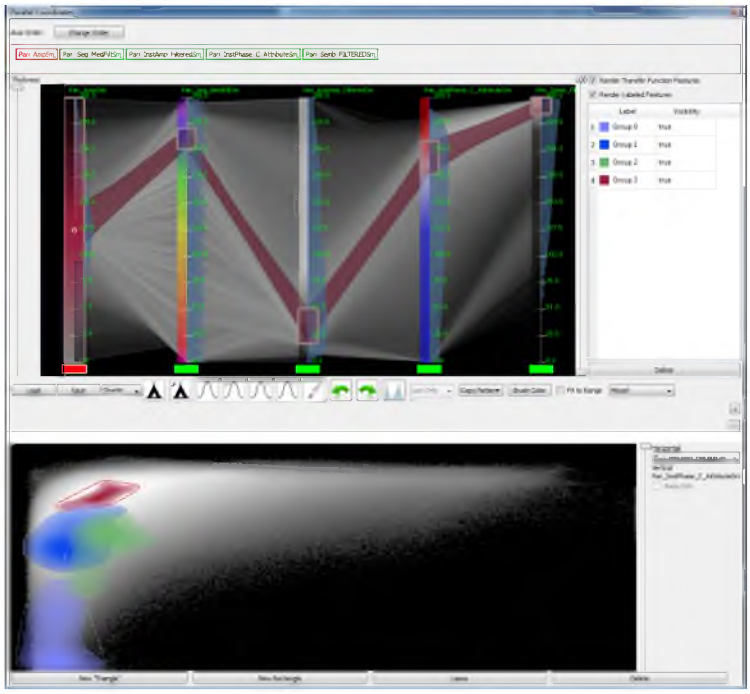


Figure 4.6. The high-dimensional transfer function editor. Note that the first attribute, seismic amplitude, is locked.

TF widgets, but preserves familiar 2D TF widgets: rectangle, triangle and lasso. Closely linked with the panel view and the HDTF editor, the projection viewer shows the dimensional reduction view of user-selected samples.

4.7 Use Cases

Two use cases from different application domains will be shown to demonstrate the usefulness of our proposed method. The first case is a commonly used hurricane simulation dataset and the second case is a 3D seismic survey data with several derived attributes, which will be used to extract geological features that are important in the petroleum industry since they indicate potential oil and gas reservoirs.

4.7.1 Hurricane Isabel Simulation

We have experimented with the proposed system on the simulation dataset: hurricane Isabel. The hurricane Isabel dataset [47] is a multivariate multiple time step atmospheric simulation. Eight attributes of time step 25 are used to generate

the result in Figure 4.7, namely the pressure, the temperature, the total precipitation mixing ratio (PRECIP), the graupel mixing ratio (QGRAUP), the water vapor mixing ratio (QVAPOR), the total cloud moisture mixing ratio(CLOUD) and the speed.

The simulation dataset contains no noise and since each attribute represents a clear physical meaning, it is relatively easy to classify. A good classification can be achieved by HDTFs or alternatively by automated 2D Gaussian TFs on the projection view.

Joint histograms could be generated with continuous scatter plots [2]. The user can generate the result in Figure 4.7 by placing several large lassos on slices

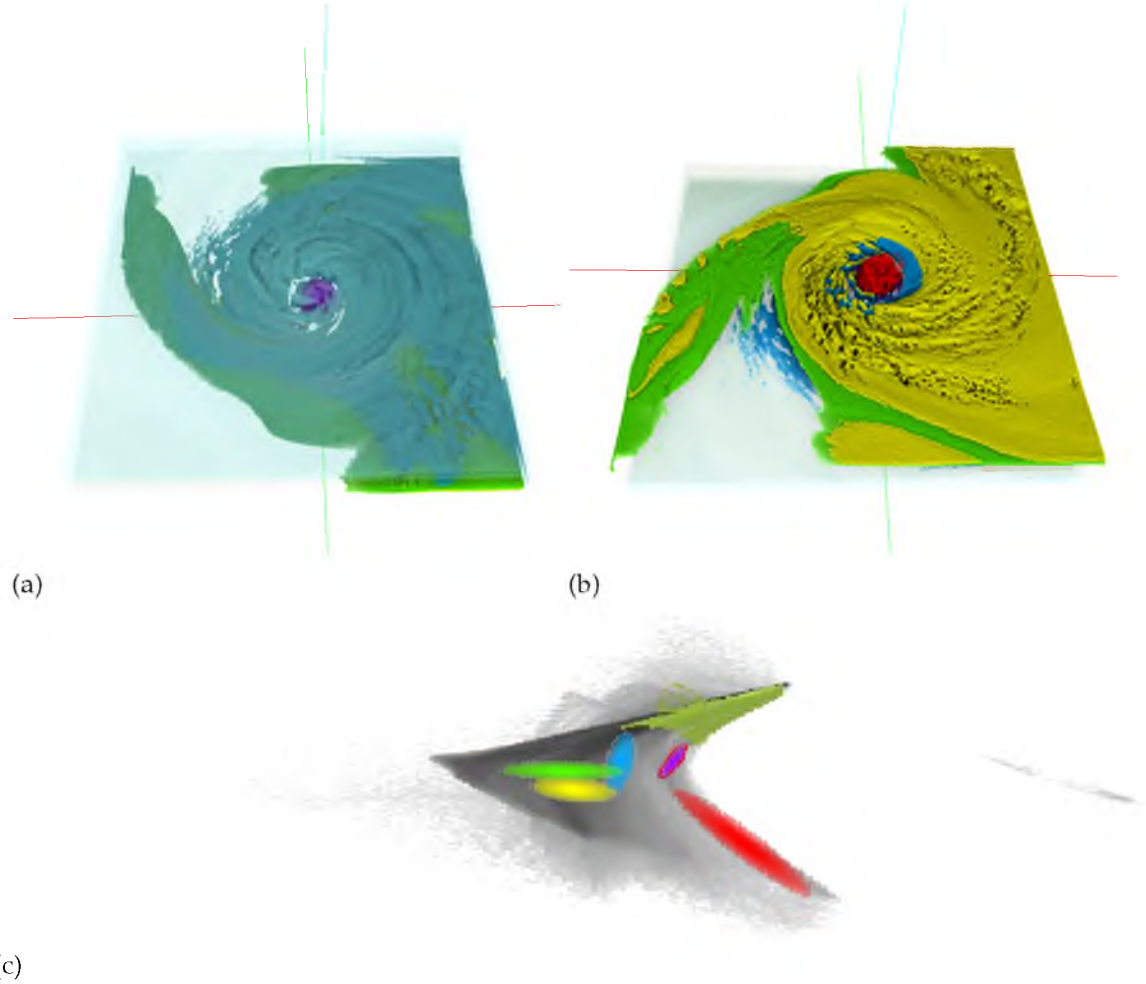


Figure 4.7. Results of a hurricane simulation dataset. The extracted features shown in (a) the top view and (b) the bottom view. Seen in (c) is the corresponding projection view with automated Gaussian TFs that produce the classification result.

in the axial view (slices indexed by the z axis) on the multipanel viewer in the data probing stage. The GMM-EM algorithm explained in Section 4.3.3 then automatically generates the TFs for classification in the qualitative analysis stage. The hurricane eye, spiral arms and the top of the atmosphere are clearly seen in Figure 4.7. Due to the nature of these data, no feature refinement is required.

The results are similar compared to previous methods. With previous methods [21, 1, 5, 31], one has to carefully design the TFs one by one for each feature, either by editing pairs of histograms [21], or PCP-based HDTF [1, 5] or high-dimensional Gaussian TF and MDS-based TF [31]. Our method, however, allows the user to extract the same features by simply drawing several large lassos across the features on the multipanel viewer, which is significantly easier.

4.7.2 3D Seismic Dataset

3D seismic imaging has been the standard for oil and gas exploration for decades, and more recently, multiattribute volumes derived from the seismic amplitude volume have been used to aid the understanding of the seismic surveys [12]. However, these derived volumes are visualized individually in current seismic data analysis tools and therefore the relationships between attributes are lost. With the proposed methods and our system, our collaborating geophysicists successfully extract refined geological features from the dataset and can export the results as a labeled volume for further processing.

The data used are a part of the public 3D seismic survey dataset "New Zealand" of size $213 \times 276 \times 426$, in which different geological features exist, including channels, faults and a salt dome, that can be potential reservoirs of oil and gas. Five attributes have been derived from the original seismic amplitude data (`Amp`). Using the six attributes, namely `Amp`, `Seg_MedFilter`, `Inst_Amp`, `Inst_Phase_Entropy`, `Semb` and `Semb_Thick`, geophysicists are able to clearly extract meaningful features as shown in Figure 4.8.

Note that for all features, `Amp` provides only context and is not clamped in order to select complete geological structures. The geophysicist starts the exploration by scrolling through the slices in the inline direction (slices indexed by the x axis) and finds a shallow channel complex in the `Amp`.

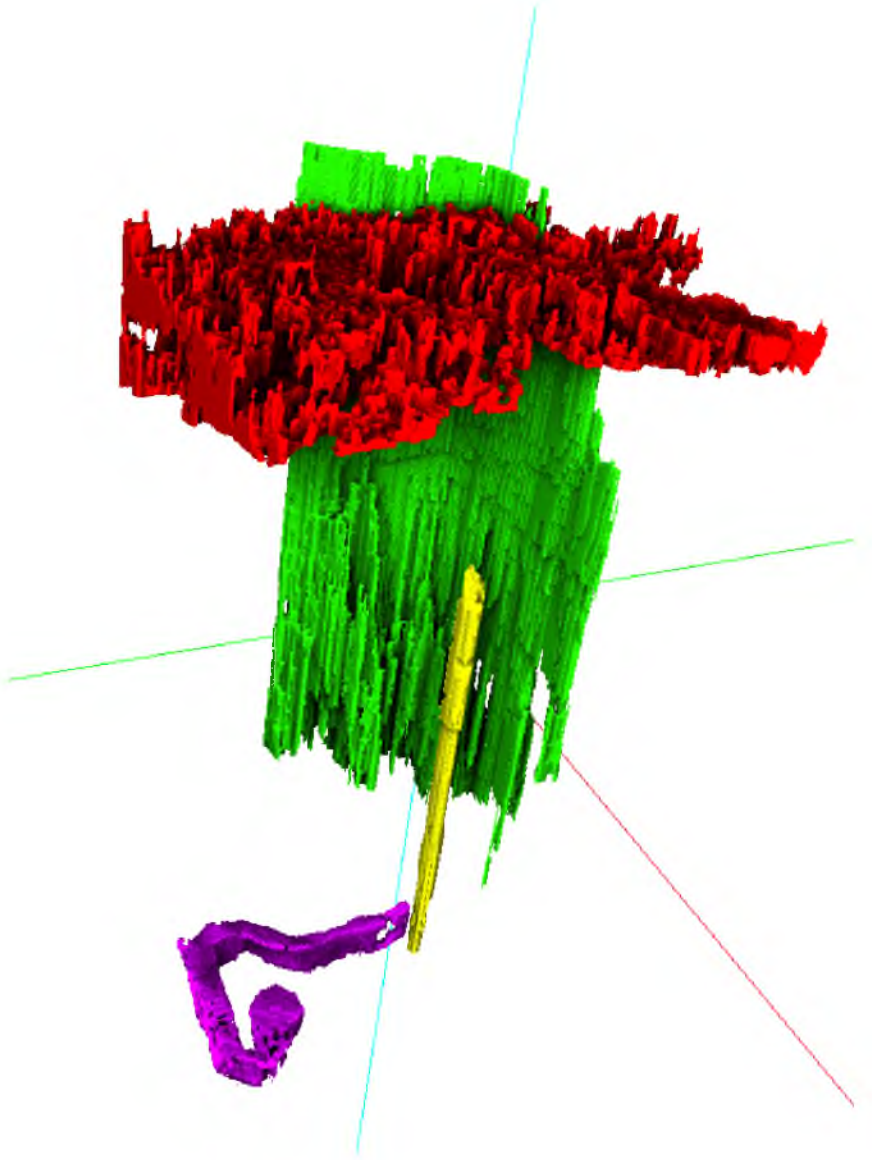


Figure 4.8. Extracted geological features in a seismic dataset. Features extracted from the New Zealand dataset: a shallow channel complex in red, a salt dome shown in yellow, a deeper channel shown in purple and the largest fault in green.

In the data probing stage, a lasso around the channel complex is drawn on the `Amp` attribute seen in Figure 4.9(a), and from this an HDTF is generated with the KDE method described and fine tuned in the qualitative analysis stage as shown in Figure 4.9(c). The main connected component as shown in Figure 4.9(b) is extracted in the feature refinement via segmentation brushing in the 3D view.

The salt dome appears to be a distinct feature on slices in the cross line direction

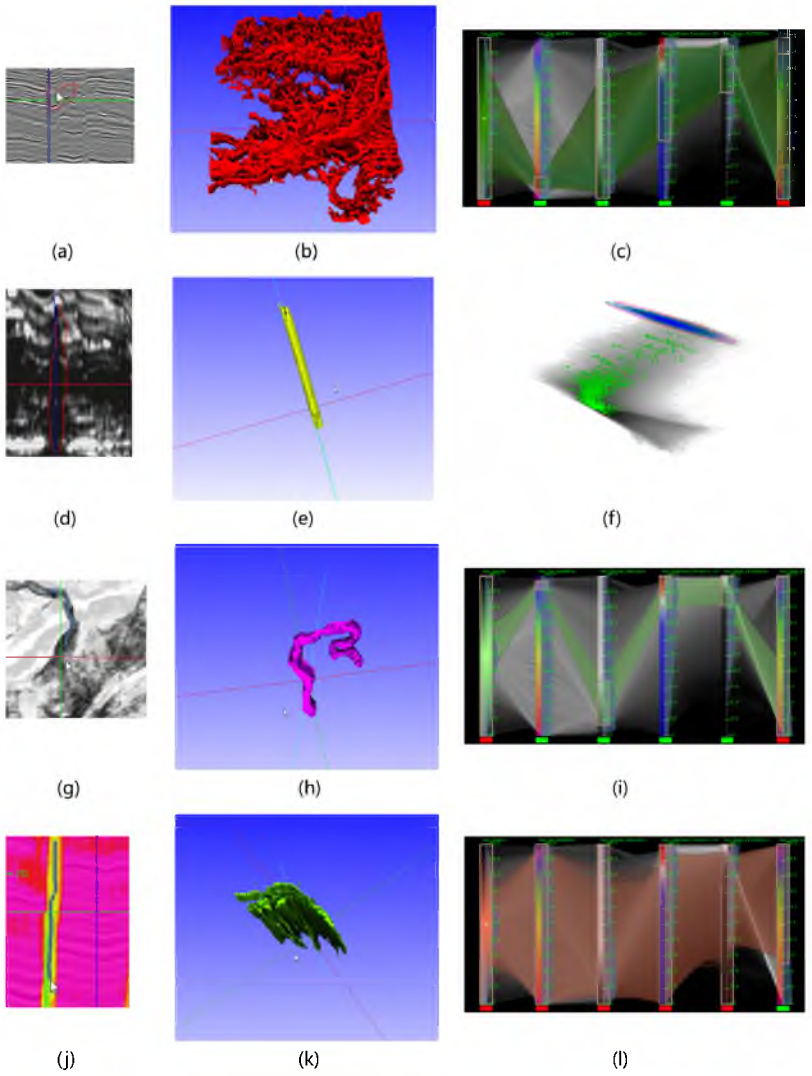


Figure 4.9. User selected samples and classified features of a seismic dataset. Refined features shown in the middle column with the user's selection of regions of interest shown in the left column and the TFs shown to the right. Note that the color of the refined features are independent of their TF colors.

(slices indexed by the y axis) and so the automated Gaussian TFs in the projection view are utilized. By drawing a lasso around the salt dome on the `Inst_Amp` attribute, as shown in Figure 4.9(d), Gaussian TFs are automatically generated in the projection view. The visualization of the isolated salt dome seen in Figure 4.9(e) is created by enlarging the Gaussian widget (Figure 4.9(f)) that highlights the salt dome and drawing a region-growing brush stroke on the salt dome.

Scrolling down through the time direction (slices indexed by z axis), a smaller

channel is discovered at the bottom of the volume. The lower channel is clearly visible in the `Inst_Amp` and `Semb` attributes. Using the magic wand tool inside the channel on the `Inst_Amp` attribute (Figure 4.9(g)), and fine tuning the HDTF as seen in Figure 4.9(i), the channel can be extracted. Due to its connection to the surroundings, we use the lasso tool to manually extract only the channel as shown in Figure 4.9(h).

Finally, when the geophysicist switches back to the inline direction, the faults are easily recognized in the `Semb_Thick` attribute and are partly extracted via magic wand brushing (Figure 4.9(j)). Since the faults depend only on the `Semb_Thick` attribute, this attribute is fine tuned to cover the entirety of the faults (Figure 4.9(l)). The largest fault as seen in Figure 4.9(k) is extracted via region-growing brushing in the 3D view. In theory, previous methods that use only the value domain TF widgets are able to extract the features. However, our collaborating geophysicists have found that in practice, it becomes overwhelmingly laborious.

4.8 Implementation

The system is implemented in C++ with OpenGL and Qt. The magic wand tool, conditional histogram generation, PCP creation and region-growing-based segmentation are accelerated using GLSL shaders. Directional occlusion for volume rendering and PCP rendering are implemented on the GPU as well. The `figtree` package [69] is utilized for efficient kernel density estimation. The linear algebra operations are aided by the `Eigen` library [29].

CHAPTER 5

GUIDEME: SLICE-GUIDED MULTIVARIATE EXPLORATION OF VOLUMES

Multivariate volume visualization is important for many applications including petroleum exploration and medicine. State-of-the-art tools allow users to interactively explore volumes with multiple linked parameter-space views. However, interactions in the parameter space using trial-and-error may be unintuitive and time consuming. Furthermore, switching between different views may be distracting. We propose GuideME, a novel slice-guided semiautomatic multivariate volume exploration approach. Specifically, the approach comprises four stages: attribute inspection, guided uncertainty-aware lasso creation, automated feature extraction and optional spatial fine tuning and visualization. Throughout the exploration process, the user does not need to interact with the parameter views at all and examples of complex real-world data demonstrate the usefulness, efficiency and ease-of-use of our method.

5.1 Method Overview

Our approach utilizes automated methods to replace a laborious user workflow. A guided uncertainty aware lasso that snaps to feature boundaries is proposed to assist region selection, automated transfer function tuning is applied to avoid trial-and-error transfer function design and finally a 3D connected component is automatically extracted. The result of the method is a 3D connected component that best represents the intention of the user. As shown in Figure 5.1, our approach comprises four conceptual stages: attribute inspection, uncertainty aware lasso drawing, feature extraction based on automated transfer function tuning and volume visualization with optional spatial fine tuning.

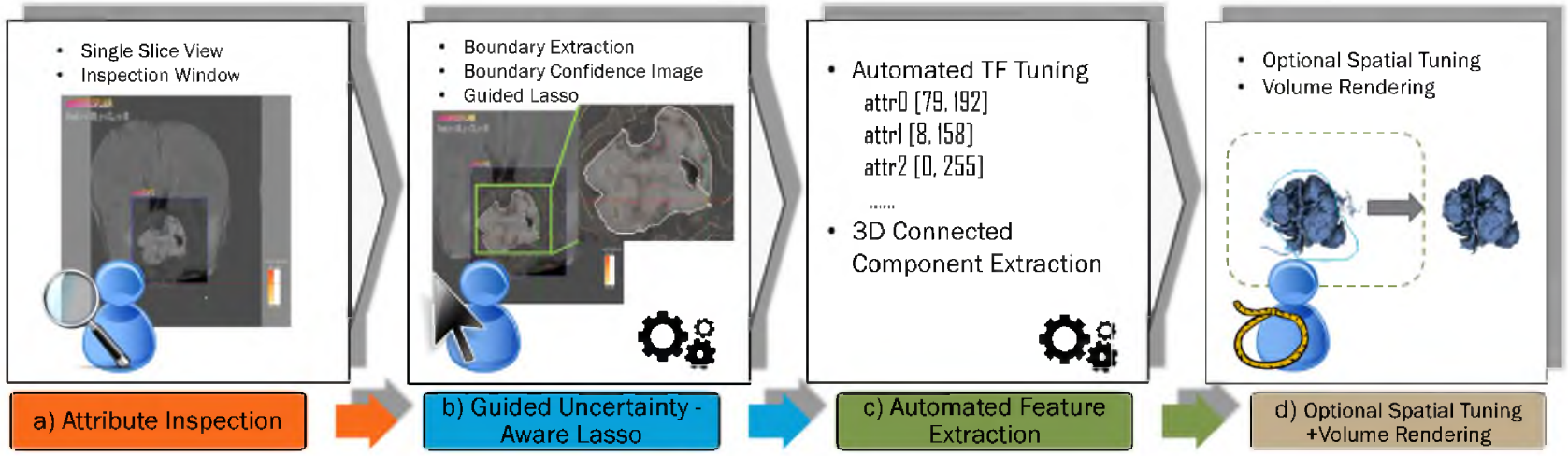


Figure 5.1. The workflow of GuideME. Four stages are included in our proposed method: attribute inspection, guided uncertainty-aware lasso for defining features, feature extraction through automated transfer function tuning and finally, spatial fine tuning and visualization. Shown in this figure is the example of extracting the tumor core in a multimodal MRI brain scan data.

In the following, we explain attribute inspection in this section, detail the uncertainty aware lasso in Section 5.2, and the automated feature extraction in Section 5.3, and briefly describe the volume rendering and spatial fine tuning in Section 5.4.

During attribute inspection, the user inspects one attribute at a time through a focus window on slices. The focus window serves as a "magic lens" [3] to overlay the chosen attribute with the contextual background. Then, the user selects one or more attributes that can properly represent the feature boundaries. The selected attribute(s) are then used to generate uncertainty information in terms of boundary confidence as shown in Figure 5.2 as the color coded curves . With the uncertainty information, the user is able to draw guided uncertainty-aware lassos that snap to feature boundaries via a few mouse clicks as the white curves seen in Figure 5.2. Next, the feature is extracted using an automated feature extraction approach that minimizes false positives outside the lasso region while preserving true positives inside, and finds the dominant 3D connected component within the lasso region. Finally, further spatial fine tuning can be conducted in the 3D view.

5.2 Guided Uncertainty-aware Lasso

In this stage, we first extract feature boundaries using edge detection on an anisotropic diffused image of the data slice. A boundary confidence image describing the uncertainty can then be derived from the feature boundaries of the selected attributes. Next, the system calculates an optimal path between user clicks based on the uncertainty information to create a guided uncertainty-aware lasso. The details of each component will be described in the next subsections.

5.2.1 Boundary Extraction

The feature boundaries are extracted via edge detection on an anisotropic diffused image of current slice I_a of attribute a. We apply anisotropic diffusion [76] to I_a to remove noise while preserving edges. The flow function $g(\nabla I_a)$ shown on the following page is used, where K is a constant that is empirically set to 30, which gives a good diffusion stopping effect, and the partial differential equation is numerically solved with a small number of iterations.

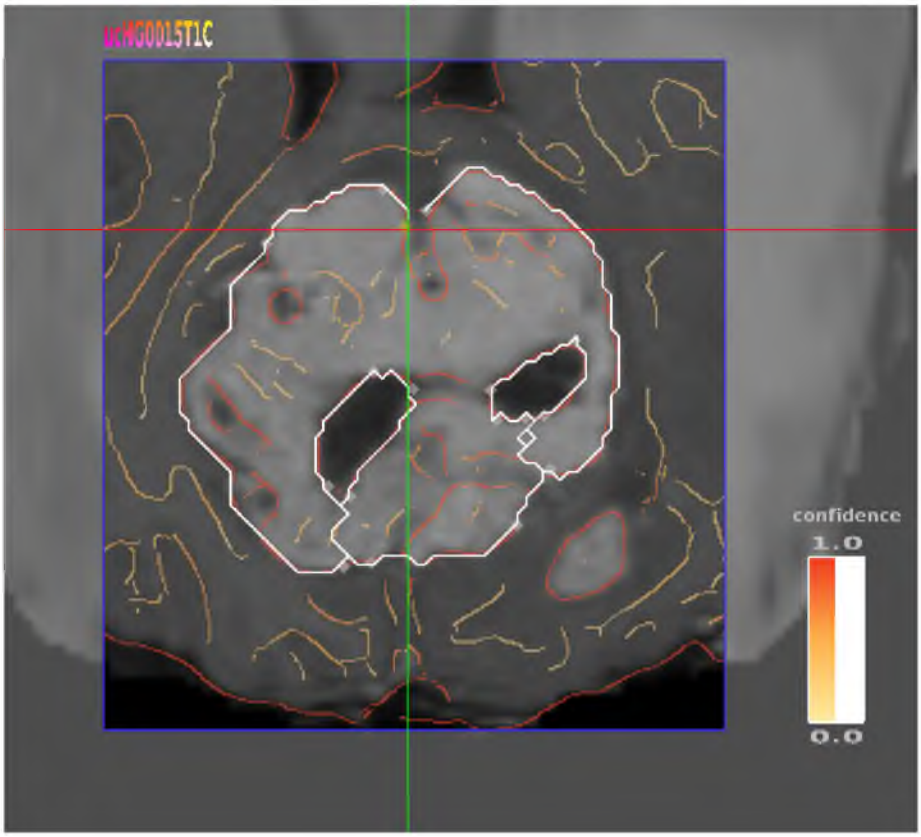


Figure 5.2. The inspection window and the boundary confidence image. On an MRI brain scan dataset, the inspection window with attribute, T1C is shown over a tumor region with the FLAIR attribute as background. The boundary confidence derived from T1C, is rendered overlaying the data slice with a color map shown to the right.

$$g(\nabla I_a) = e^{-(\frac{\|\nabla I_a\|}{K})^2}$$

Then, the edges in the filtered image are extracted by Canny edge detection [11], which is simple and has good accuracy. The gradient field is first derived, and we then compute the direction of the gradient and classify it into four cases: horizontal, vertical and two diagonals. We remove pixels that are not maximal in the pixel's classified direction in the nonmaximal suppression step. Finally, we conduct the hysteresis step via recursive edge tracing. To avoid user involvement in the setup of the lower and upper thresholds, we compute the histogram of the gradient magnitude and accumulate histogram bins until the sum is equal to or greater than a certain portion T_{gm} of the count of voxels on the given slice and take the gradient magnitude value of that bin as the upper threshold t_{up} [11]. The lower

threshold t_{lo} is then computed by multiplying the upper threshold with a constant k_l. We adopt the settings of $T_{gm} = 0.7$ and $k_l = 0.4$ from Matlab and find they work well on all datasets we use.

5.2.2 Boundary Confidence Image

A boundary confidence image can be derived from the extracted boundary images of user-chosen attributes $\bar{A}_s$ from the pop-up menu in the inspection window to indicate the uncertainty. As an uncertainty measurement, the boundary confidence should be in the range $[0,1]$, which is defined by Equation 5.1.

$$I_b = \begin{cases} 1, & \text{if } \|\nabla I_a\| > t_{up} \\ \max_{\bar{A}_s} \frac{\|\nabla I_a\| - t_{lo}}{t_{up} - t_{lo}}, & \text{if } t_{lo} \leq \|\nabla I_a\| < t_{up} \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

The boundary confidence of each attribute is computed by normalizing the gradient magnitude of the extracted boundaries. The normalization uses the upper and lower thresholds defined in the edge extraction process, and values greater than the upper threshold are mapped to one. Pixels that are not detected as edges are simply mapped to zero. Next, the boundary confidence value for all selected attributes is calculated by blending individual boundary confidence using the *MAX* operator, which keeps the blended value inside the range $[0,1]$. An equal weight is assigned to each attribute so as to avoid having the boundary confidence of one attribute reduce the importance of others, and to remove the requirement of user involvement. A sequential color map scheme suggested by Color Brewer [39] is used for the rendering of the boundary confidence image as seen in Figure 5.2. The color map range and opacity can be interactively modified to remove or highlight certain confidence value ranges.

5.2.3 Guided Uncertainty-aware Lasso

Given the boundary confidence image I_b with its pixels P, and two user defined end points u and v, an uncertainty-aware lasso that snaps to feature boundaries can be thought of as finding an optimal path that minimizes the transition energy between each pixel as shown in Equation 5.2.

$$E(I) = \sum_{p \in P} E_s(p) \tag{5.2}$$
$$E_s(p) = 1 - ||I_b(p)|| \text{ where } p \in P$$

The energy in Equation 5.2 can be efficiently optimized using Dijkstra's algorithm [20] from end point u to v.

To compute the optimal path using Dijkstra's algorithm, we convert image I_b into a bidirected graph where each pixel p is assigned a node and the edge from p to its neighboring pixel p_n has energy $E_s(p_n)$ as weight. Thanks to the efficiency of Dijkstra's algorithm, the user is able to interactively set the end points u, v by clicking on the boundary confidence image inside the inspection window to setup, and edit the end points by a click and drag interaction.

5.3 Automated Feature Extraction

In this stage, we extract the feature based on the lasso region via an automated feature extraction procedure. An initial transfer function is generated and tuned using our novel automated transfer function tuning method. The resulting transfer function gives minimum false positives outside the lasso region while preserving true positives inside the lasso. Then, the dominant 3D connected component in the classified volume is extracted.

5.3.1 Automated Transfer Function Tuning

The core of our feature extraction approach lies in automated transfer function tuning. By watching the domain experts manually fine-tuning the transfer functions using existing tools, we observed that they focus only on the lassoed region and try to minimize false positives outside the lasso while preserving true positives inside the lasso. Therefore, we mimic this procedure by formulating an optimization problem. For a multivariate volume of M attributes, $\bar{A} = (A_1, A_2, \ldots, A_M)$, we model the M-dimensional transfer function space by conducting `AND` operation between the M 1D spaces. This avoids erroneous classification caused by a separable M-D transfer function composed of M 1D transfer functions multiplied together as shown on page 258 in [36]. We use only binary values 0 and 1 to indicate the selection of attribute values, and denote such a binary transfer function as f

and its i-th 1D subspace f^i. An initial transfer function can be set up, and then optimized by maximizing a response function.

Given the lasso region, an initial transfer function, f_0, can be created by querying the attribute values of the pixels inside the lasso. We conduct the query by a simple traversal over the slice and tested if the current pixel on the slice falls inside the lasso. If it does, we record the pixel's M-queried results into the corresponding locations of the histogram array H, where H is a 1D histogram array of M-layers, and each layer H_i is a 1D histogram associated with an attribute A_i. Then, an initial binary transfer function f_0 is generated by setting nonzero histogram locations to one.

We formulate a response function $R(I_s, I_c)$ of two binary images: the user lasso image I_s and a connected component image I_c of the transfer function classified image I_f. Since we focus on the lasso region only, we take the dominant connected component of the classified image inside the lasso. Specifically, we extract all connected components in the classified image and create a histogram of tag values inside the lasso. Then we keep only the connected component with the most frequent tag in this histogram and discard other connected components.

As shown in Figure 5.3, the relationship between I_s, I_c and I_f is clearly demonstrated. For a multivariate transfer function f, a slice of M-attribute volume with pixels p, I_f can be denoted as:

$$I_f = \{p | f(\vec{v_p}) > 0\} \tag{5.3}$$

where $\vec{v_p} = (v_p^1, v_p^2, \ldots v_p^M)$ is the multivariate value of pixel p. In practice, whenever the transfer function changes, we update I_f and extract the dominant connected component to get image I_c. After defining the terms, we are able to describe the response function.

The response function $R(I_s, I_c)$ can then be written as:

$$\begin{aligned} &R(I_s, I_c) = w_r \cdot r(I_s, I_c) + (1 - w_r) \cdot s(I_s, I_c) \\ &\text{subject to } N_c \geq N_{min} \\ &\text{where } w_r \in [0, 1] \end{aligned} \tag{5.4}$$

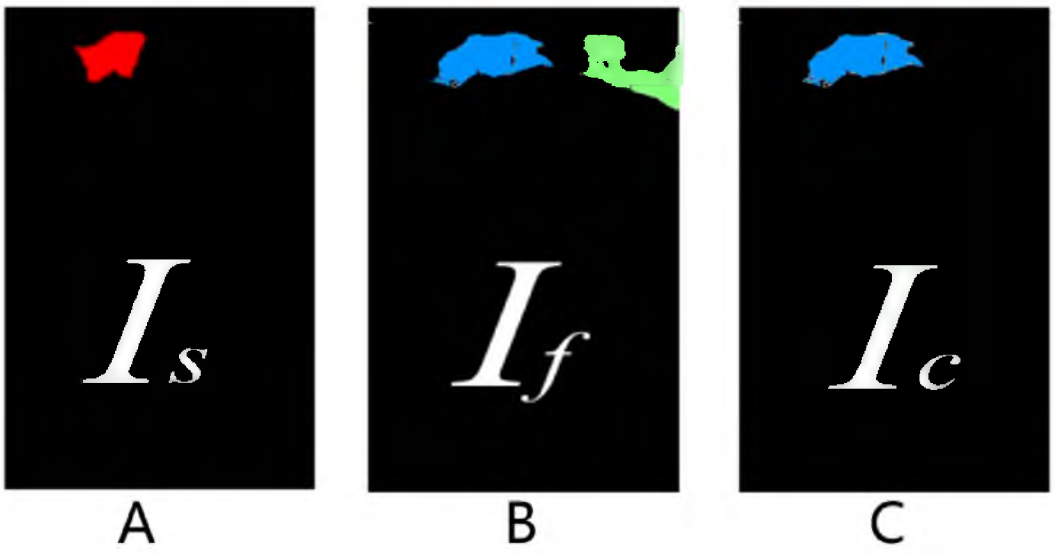


Figure 5.3. Illustrations of images involved in the automated transfer function tuning process. A shows the lasso image I_s where red indicates the lasso region. The transfer function classified image I_f is seen in B where the blue and green regions are classified by the transfer function. C shows the connected component image I_c where blue is the dominant connected component.

where $r(I_s, I_c)$ is the cross-correlation coefficient of images I_s and I_c, $s(I_s, I_c)$ is a smoothness term; w_r is a tunable weight that is empirically set to 0.7 by default; and the nonzero pixel count N_c of image I_c has to be greater or equal to N_{min}, which we empirically set to be 90% of the nonzero pixel count of the lasso image I_s. The cross-correlation coefficient is computed by treating the images as arrays of binary pixels as seen in Equation 5.5.

$$r(I_s, I_c) = \frac{\sigma_{I_s I_c}}{\sigma_{I_s}\sigma_{I_c}} = \frac{\sum_{i=1}^{N}(I_{s_i} - \bar{I}_s)(I_{c_i} - \bar{I}_c)}{\sqrt{\sum_{i=1}^{N}(I_{s_i} - \bar{I}_s)^2}\sqrt{\sum_{i=1}^{N}(I_{c_i} - \bar{I}_c)^2}} \tag{5.5}$$

The smoothness term $s(I_s, I_c)$ measures the normalized differences of nonzero pixels p_c and the neighborhood p_{cn} of I_c inside the lasso, in our case eight neighbors n, in the classified region inside the lasso:

$$s(I_s, I_c) = -\frac{\sum_{P_s}\sum_{n}(p_c - p_{cn})}{s_{max}} \tag{5.6}$$

where P_s are nonboundary pixels of I_s and s_{max} denotes the maximum possible differences inside I_s. Specifically, we derive s_{max} by considering the extreme

case that all nonzero pixels are surrounded by zero pixels, which gives $s_{max} = 8 \times \frac{1}{1+(4\times0.25+4\times0.5)} \times |P_s| = 2 \times |P_s|$ as the corner pixels are shared by four neighboring stencils and the middle pixels on each side are shared by two stencils. Maximizing the response function R encourages higher correlation between the classified region and the lasso, while penalizing the elimination of true positives inside the lasso. As a result, maximizing Equation 5.4 minimizes false positives outside the lasso while preserving true positives inside the lasso.

Since there is no direct link between the transfer function and R, Equation 5.4 is hard to optimize using methods like gradient descent or conjugate gradient. We therefore propose the following greedy algorithm to approximately maximize $R(I_s, I_c)$. As seen in Algorithm 5.1, we first determine the order for optimizing the 1D subspace of individual attributes of the transfer function. This step is necessary because this ordering affects the final result. We assume that an attribute that has higher R than others is likely to require fewer changes for optimization than others, which is confirmed by experiments on the datasets we used. Therefore, we use a conservative heuristic that optimizes the 1D subspaces from more contributing ones (higher R) to less contributing ones (lower R) for the feature of interest. The reason is two-fold: first, this heuristic may lead to minimal iterations of optimization. Second, if we start with less contributing attributes, it is likely to overly eliminate true positives inside the lasso and other attributes may never have the chance to remedy such an error. We first get the binary images classified by the initial transfer function of individual attribute f_0^i for all attributes $\bar{A}$. The response function value R is evaluated for each binary image, and then we sort the attributes by R. Next, we select the attribute that has the highest response

Algorithm 5.1 TF_Opt(I_s, f_0)

```
for Attribute A_i in all M attributes do
    Generate I_c^i with f_0^i
end for
Sort all attributes Ā with descending order of R(I_s, I_c^i)
f = f_0
for Attributes A_j in sorted Ā do
    ModifyTF(f^j, H_j)
end for
```

function value, and maximize the response function R by optimizing the transfer function's j-th subspace.

To optimize the individual subspace of the transfer function, we propose a simple yet efficient transfer function bin dropping approach as seen in Algorithm 5.2. In Figure 5.4, the steps of the bin dropping method are clearly illustrated on the top.

The method starts with the computation of the mean value μ of the associated queried histogram of the given attribute. Then, the farther end of the attribute to μ is chosen as direction d as it is likely to contain more false positives. The algorithm finds the optimal point that maximizes R by dropping bins from the transfer function in the direction d, and then performs the same operations on the other direction until converges. The effect of the automated transfer function tuning process is shown below in the figure.

5.3.2 3D Connected Component Extraction

Transfer function does not contain any spatial information, and therefore even an optimized transfer function may contain false positives in 3D. Therefore, the last step of feature extraction is to apply connected component finding to extract the intended feature in the transfer function classified volume. We first extract all connected components in the classified volume, and then query tag values inside the user lasso on the slice. The connected component whose tag value is most

Algorithm 5.2 ModifyTF(f^i,H_i)

```
Compute μ of H_i
Select direction d whose bin is farther to μ
for true do
  if R^{n+1}(I_s, I_c) > R^n(I_s, I_c) then
    Drop bin from f^i in direction d
  else
    if Both directions have been tried then
      break
    else
      Switch direction d
    end if
  end if
end for
```

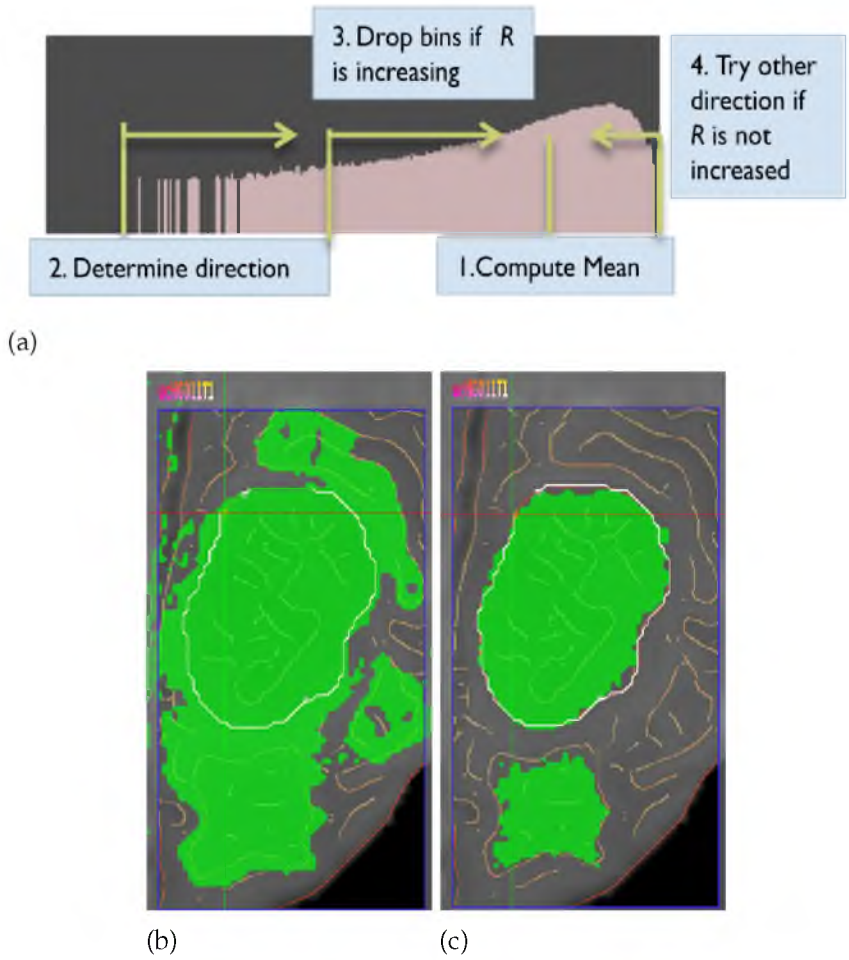


Figure 5.4. Illustrations of the transfer function modification process and effects of the process. (a) shows the steps involved in transfer function modification by bin dropping. (b) is the initial transfer function classified result (green) using the queried values from the lasso (white) on the MRI brain scan HG11. (c) shows the optimized transfer function classification result.

frequent is then selected. Next, the selected connected component is given a color and opacity and other components are discarded.

5.4 Volume Rendering and Spatial Fine Tuning

Once the 3D connected component has been extracted from the automated feature extraction stage, the classified result is stored as a tag volume and visualized using volume rendering. We adopt the directional occlusion shading method from Schott et al. [84], which provides better depth cues than local shading models as demonstrated by [61] and has been shown to provide more insights into seismic

datasets [73]. To provide smooth tag volume rendering, we utilize a simplified version of [34], which can be efficiently computed in the GPU shader.

In some cases, even the user intended connected component contains false positives. As such, we provide the user a simple yet flexible means of spatial fine tuning: volume-rendered image space lasso. This image space lasso allows the user to select voxels inside the back projected volume of the volume rendered image. Two modes are provided: the keep mode keeps voxels inside the lasso while removing others, and the remove mode does just the opposite, which is similar to [95].

5.5 Implementation

Our proposed method has been implemented in C++, with OpenGL and CUDA for rendering and computation. The user interface has been implemented using Qt. Most image processing procedures and value querying tasks, slice rendering and volume rendering have been implemented on the GPU using GLSL shaders with the `GL_EXT_shader_image_load_store` extension. The rendering of lassos and texts is accelerated with NVidia's `NV_path_rendering` SDK [71]. Correlation coefficient computation is implemented with thrust CUDA library [42]. Graph creation from the slice and Dijkstra's algorithm are implemented on the CPU. Efficient connected component extraction is realized with CONNEXE library [66].

5.6 Examples

To demonstrate the usefulness and efficiency of our method, we apply it to complex multivariate datasets in two disciplines: multivariate seismic data in the petroleum industry as shown in Figure 5.5 and multimodal brain scans from the 2013 Medical Image Computing and Computer Assisted Intervention (MICCAI) Conference challenge. To validate our method, we compare the method against previously extracted features by domain experts for the seismic example and hand-segmented ground truths for the MRI brain example.

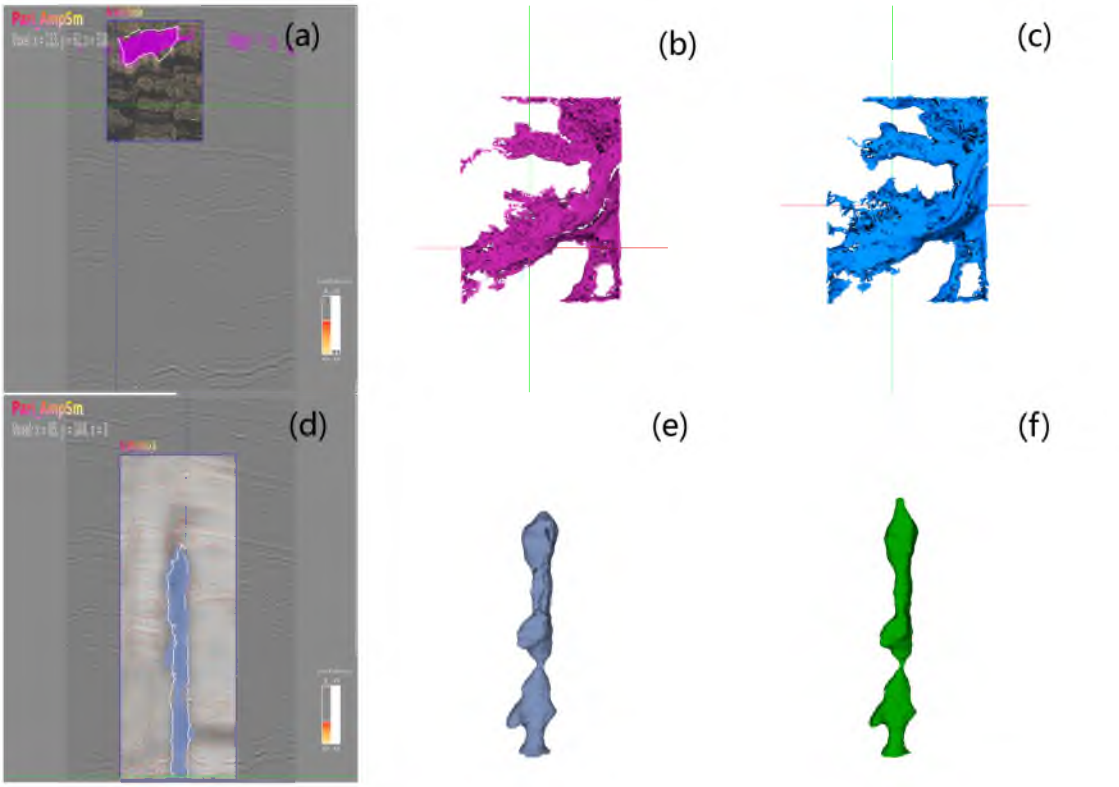


Figure 5.5. Results of the New Zealand seismic dataset. The first row shows the upper channel in the dataset. Shown in subfigure (a) is the lasso that extracts the feature, (b) is the result using GuideME, and (c) is the result generated by a domain expert using [103]. To the bottom, the salt dome feature is shown. Subfigure (d) shows the lasso region drawn for feature extraction and in (e), shows the result using GuideME, and in (f), the result extracted by the domain expert.

5.6.1 Seismic Dataset

The seismic dataset we used is a part of the public New Zealand seismic data. Six attributes have been computed from the original seismic amplitude: instantaneous amplitude `InstAmp`, instantaneous phase `InstPhase`, entropy of instantaneous phase `InstPhase_Entropy`, horizon layers `Layer_Seg`, semblance `Semb` and thickness of semblance `Semb_Thick`. The user starts the exploration on slices in "inline" direction, which in our case is the slices on the "`YZ`" plane. A potential channel structure draws the user's attention, and the user zooms in and places the inspection window over this feature of interest.

As seen in Figure 5.5(a), after inspecting different attributes, it is decided that `Inst_Amp` best represents the boundary of this channel structure.

Next, the user creates an uncertainty-aware lasso by placing several anchor points on the feature's boundary with relatively high boundary confidence and fine-tunes it by dragging the anchor points. The feature is then extracted as shown in Figure 5.5(b). Compared to Figure 5.5(c), where the feature is extracted by our collaborating geophysicists using [103], our automated method provides a similar result, which captures the connected body and its meander details. However, the proposed method greatly reduces the time to achieve such a result. With the same feature lassoed, our method takes around a second as shown in Section 5.6.3 to extract the feature, whereas pure interactive tuning takes minutes. Moreover, the lasso drawing process is guided, which may also be faster than free-hand drawing. Next, we would like to extract the salt dome structure found near the center of the volume. Again, the user utilizes the inspection window to examine attributes that emphasize this feature, and finds that in addition to the `InstAmp` attribute, the `InstPhase_Entropy` attribute best illustrates the boundary. Then, a lasso is drawn with boundary confidence information calculated from these two attributes as shown in Figure 5.5(d). The result as seen in Figure 5.5(e) is comparable to the one from domain expert interactions as seen in Figure 5.5(f).

To make a quantitative comparison, we compute the dice score, i.e., twice the number of overlapping voxels from two datasets divided by the sum of all voxels from the two datasets, for our proposed approach against the results conducted by the domain expert. The dice score for the upper channel is 0.84 and the score for the salt dome is also 0.84 as shown in Table 5.1. Both cases demonstrate that our method is able to extract features that are similar to interactively extracted and fine-tuned features generated by domain experts, but is faster and easier. Furthermore, the entire user interaction in our method happen on slices and the 3D view, which may be more familiar and intuitive to domain users.

5.6.2 Brain Scan

To demonstrate the generality of GuideME, brain tumor image data from the NCI-MICCAI 2013 Challenge on Multimodal Brain Tumor Segmentation [25] (BRaTS) were used. The data consist of multicontrast MRI scans of 30 glioma

Table 5.1. Quantitative comparisons and timing results for the features extracted in the example datasets. For the timings, the first numbers in the parenthesis are the automated transfer function tuning time and the second numbers are the 3D connected component finding time.

Dataset	Size	#Attr	Feature	Dice	Timing
Seismic	x 274 y 426 z 245	7	Upper Channel Salt Dome	0.84 0.84	1.231s (741+490ms) 1.373s (888+485ms)
MRI HG15	x 161 y 216 z 177	4	Tumor Core Edema Both	0.85 0.81 0.87	0.833s (411+422ms) 0.817s (402+415ms) —
MRI HG11	x 161 y 216 z 177	4	Tumor Core Edema Both	0.84 0.76 0.87	0.730s (350+380ms) 0.815s (405+410ms) —

patients with expert annotations for the tumor core and the edema as ground truths. The datasets in the challenge all contain four channels: FLAIR, T1, post-Gadolinium T1 (T1C) and T2. We chose high-grade subject HG15 and HG11, as shown in Figure 5.6, for which the methods in the proceedings of the BRaTS challenge gave good agreement with the ground truths.

We describe detailed operations to extract the tumor in HG15. Setting the FLAIR as the context attribute, and browsing the slices on `XY` direction, a large tumor region is observed. First, we extract the tumor core. Visualizing different attributes inside the inspection window, it is apparent that the T1C attribute is the best candidate for boundary confidence for the tumor core. A lasso is then drawn around the tumor to extract it, and a volume-rendered image space lasso is used to fine tune it. The extracted tumor core is seen in blue in Figure 5.6(a). Next, we extract the edema. Checking with different attributes inside the inspection window, the FLAIR attribute best describes the edema. Clicking along the edema boundary, the feature is then extracted and fine-tuned using the volume rendered image space lasso. The final classification of the edema is shown in red as seen in the figure. To validate the result, we compare our classification against the ground truth segmentation using the dice score. The dice score for the tumor and the edema together is 0.87, and 0.85 for the tumor and 0.81 for the edema as seen in Table 5.1. The result is also compared to a method [13] proposed in the BraTS

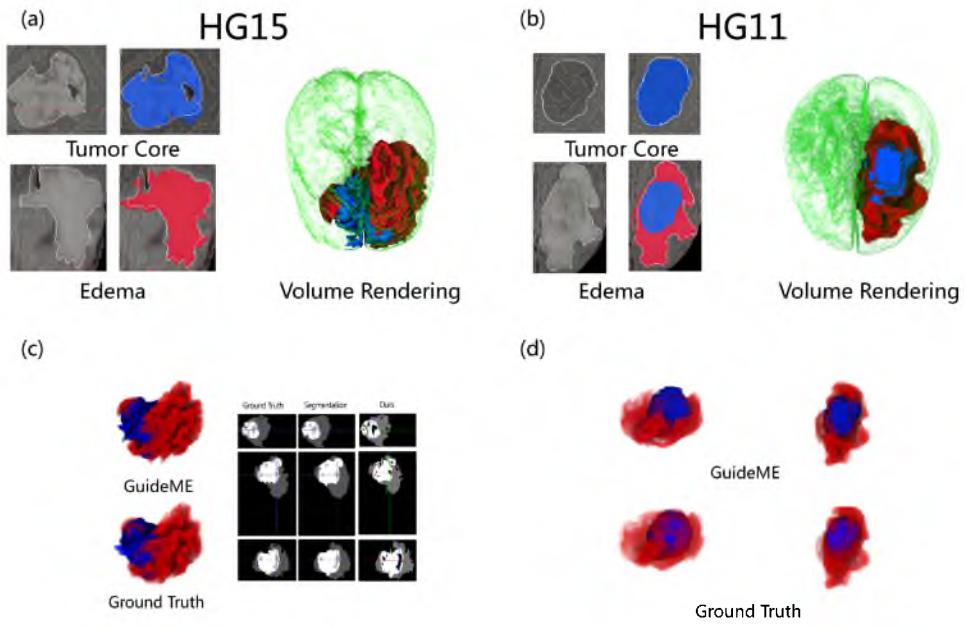


Figure 5.6. Visualization results of multimodal brain scans using GuideME. GuideME is applied to multimodal brain scans of four modals from the BRATS 2013 challenge. Subfigures (a) and (b) show the visualization result and the lassos on the slice for HG15 and HG11. The extracted active tumors are rendered in blue, the edemas in red and the context of brain tissue in green. In (c) and (d), we compare the volume rendering of the tumor and the edema of the GuideME result against the ground truth. In (c) we include a slice comparison against the ground truth as well as the method proposed in [13].

challenge as seen in Figure 5.6(c), in which the method gives slightly above the 0.91 dice score for the whole tumor region, and around 0.90 for the tumor while around 0.86 for the edema.

Similarly, we extract the tumor core and the edema for HG11 as seen in Figure 5.6(b). A comparison can be seen in Figure 5.6(d). The resulting dice score for the core and the edema is 0.87, while the core has a dice score of 0.84 and the edema has 0.76 as shown in Table 5.1. In comparison, the scores for the method in [13] are just above 0.90 for the whole tumor, around 0.82 for the core and 0.70 for the edema.

5.6.3 Performance

All the performance timings are conducted on a workstation with a single Intel Core i5 3.30GHz CPU, 16GB of main memory and an Nvidia GeForce GTX 480 with

1.5GB memory running 64-bit Windows 7 system. The creation of the boundary confidence image is around 200 ms, and the update of guided uncertainty-aware lasso typically takes below 20 ms. The timing results of features in both examples can be found in Table 5.1. For the seismic dataset, the classification of the upper channel takes 714 ms for 43 response function iterations, and it takes 55 iterations (888 ms) for the salt dome. For the brain scan datasets, both subjects have the same size and thus show similar timings. Specifically for subject HG15, the time for the automated transfer function tuning for the tumor takes 411 ms with a total number of 53 iterations. For the edema region, the timing is 402 ms for 49 iterations.

5.6.4 Discussion

Although we have not conducted a formal user study, our collaborating experts from the petroleum industry found GuideME an improvement over previous tools and provided informal comments. As in their traditional workflow, the datasets are examined and analyzed using slices. Since our collaborators have the expertise to identify a certain feature on slices, interactively selecting feature boundaries is not an imposition to them. As they are familiar with free-hand drawing on seismic slices, selecting an appropriate slice and view angle is naturally part of their workflow. While selecting features from multiattribute slices is interactive and thus done through trial and error, they have the expertise to identify a feature on slices. A previous method [103] required the user to use a free-hand lasso tool to select features of interest on slices. The free-hand lasso was cumbersome to use. GuideME guides the user through the uncertainty-aware lasso interactions where boundaries can be more rapidly and concisely defined by the drawing interaction. The domain experts commented that the GuideME system is faster and easier to use than previous tools. The experts also complained about the trial-and-error transfer function tuning in [103]. Having the automated transfer function tuning freed the experts from this tedious step. Given that the extracted features are comparable to the previous method as shown in Section 5.6.1, the domain experts found that with our proposed method, they can be more focused on their geological interpretation tasks.

Our approach is not without limitations. Our proposed method is not an automated feature extraction method for multivariate volume datasets. It requires the user to identify features of interest on slices and select them using the uncertainty-aware lasso. The selection of feature boundaries is an interactive process that requires the user's expertise and understanding of the data. The lasso region chosen for features is critical to the final visualization result. Furthermore, our method extracts features that are connected in the 3D data. Our method, therefore, would not work on datasets in where features are not distinguishable on 2D attribute slices or where features are not connected in the 3D space. While the user has to browse through the slices to detect features of interest and the lasso region drawn for features is critical to the final visualization result, we argue that this is where the expertise of the user applies, and is the main focus of our method.

CHAPTER 6

INTERACTIVE RENDERING AND EFFICIENT QUERYING FOR LARGE MULTIVARIATE VOLUMES ON CONSUMER LEVEL PCS

We present a volume visualization method that allows interactive rendering and efficient querying of large multivariate seismic volume data on consumer level PCs. The volume rendering pipeline utilizes a virtual memory structure that supports out-of-core, multivariate, multiresolution data and a GPU-based ray caster that allows interactive multivariate transfer function design. A Gaussian mixture model representation is precomputed and nearly interactive querying is achieved by testing the Gaussian functions against user-defined transfer functions on the GPU in the runtime. Finally, the method has been tested on a multivariate 3D seismic dataset which is larger than the size of the main memory of the testing machine.

6.1 Multivariate Out-of-Core Volume Rendering

Multivariate, multiresolution data blocks are stored in our virtual memory structure. The associated ray caster is able to support multivariate transfer functions (TFs), which are interactively defined by the user.

6.1.1 Virtual Memory Structure for Multivariate Volumes

We share the same virtual memory hierarchy as in the work of Hadwiger et al., namely, in a top-down manner: page table directory, page table and block caches. The difference is that instead of storing a single scalar volume in the block cache, we store data of all attributes at a given block location contiguously in the block cache. The page table entries are set to point to the beginning of the first attribute of each block. When the volume renderer makes paging requests,

the virtual memory system updates all attribute blocks of the requested block location. Also, we store our multiresolution blocks in a block file for each attribute to avoid the block building process from 2D tiles that intersect with the viewport. During initialization, our system fills the block cache by simply fetching blocks from the block cache files and sets flags in the page table and page table directory accordingly.

6.1.2 Multivariate Transfer Functions

Multivariate TFs are supported in our method, and to reduce the computational complexity, we separate the n-dimensional value space formed by n attributes into $n-1$ 2D space. The user designs the transfer function interactively on a parallel coordinate plot (PCP) based editor as shown in Figure 6.1.

We define a so called *visibility* TF, comprised by the $n-1$ 2D space, which determines the visibility of voxels and also defines an *appearance* TF of 1D, which controls the visual appearance of the visible voxels. The user defines the multivariate visibility TF by manipulating TF widgets on the parallel coordinate axes and designs the appearance TF by clicking on a desired axis and editing in a 1D TF editor to set color and opacity. Alternatively, the visibility TF can be modified in a 2D TF editor for a chosen pair of attributes for a more refined result.

In the TF sampling function of the GPU ray caster, we first determine the visibility of a voxel based on current visibility TF using an ID map, which stores

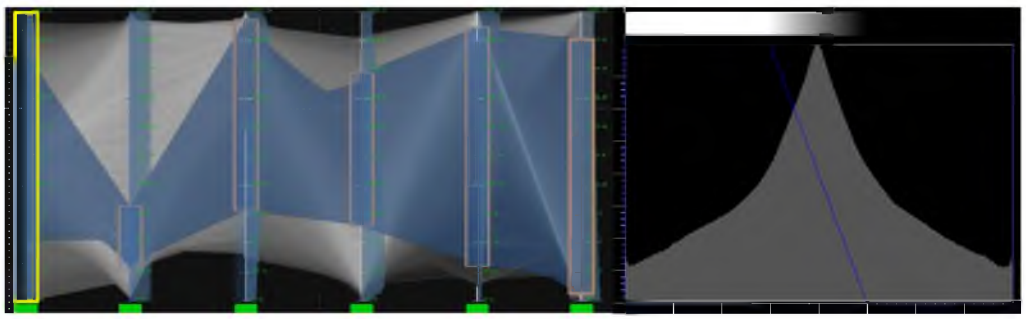


Figure 6.1. Transfer functions for the channel system of a large seismic dataset. The TFs classifying the channel system in Figure 6.2. The *visibilisty* TF is shown to the left where the blue PCP indicates the query result with the user-defined TF widget, while the *appearance* TF to the right sets a gray-level color map for the amplitude attribute.

the coverage of all user-defined visibility TFs by bitwise OR. If any attribute value of current voxel falls outside the coverage of current visibility TF, the voxel is skipped. Otherwise, the visible voxel is rendered with the user-designed appearance TF.

6.2 Efficient Multivariate Query

To allow efficient data query on the noisy seismic datasets, we propose a two-stage approach that utilizes the Gaussian Mixture Model (GMM) to compactly approximate the multidimensional distribution of data. The method first computes GMM for each block in a precomputation stage and then tests ellipse-polygon intersection in runtime to query data values for voxels selected by user-defined TFs. The per-block GMM is required to compute only once for a dataset and the runtime querying achieves near interactive performance.

6.2.1 Per-block Gaussian Mixture Model Computation

Assuming the datasets follow Gaussian distribution, we are able to describe the distribution using GMM, which is very compact in terms of storage. GMMs are computed using the well-known expectation maximization algorithm, and we precompute GMMs for each block at its finest resolution only once. In the same fashion as our TF space, as described in Section 6.1.2, we compute GMMs in the $n-1$ 2D space. The computation is performed using the CUDA thrust library and the result is written to a file that records the mean value and covariance matrix for each Gaussian distribution for each block. We empirically choose the number of Gaussians to be three as it strikes a balance between the closeness of GMM approximation of the original distribution and the compactness of storage.

6.2.2 Runtime Ellipse-Polygon Intersection Test

During visualization, the system queries data values for user-defined TFs on the GPU with the GMM information stored as a texture. For any given pair of attributes, each Gaussian distribution is a 2D ellipse and each user-defined TF is a 2D polygon. We are able to conduct the query using ellipse-polygon intersection detection, i.e., if any part of the ellipse intersects with the TF polygon in any 2D subspace of the $n-1$ 2D TF space, all values in the distribution are selected.

The ellipse-polygon intersection is hard in the original space, and we compute a circle-triangle intersection in a transformed space. It is known that the ellipse can be transformed from a circle using matrix $\Sigma^{1/2}$, which is the square root matrix of matrix Σ, which holds the eigenvectors of the ellipse. Therefore, the ellipse can be transformed back to a circle using the inverse matrix $\Sigma^{-1/2}$. The 2D polygon can be triangulated, and the triangles that form the polygon can also be transformed using $\Sigma^{1/2}$ into the circle's space, and then a much easier circle-triangle intersection test can be performed. Consequently, the query result is rendered using PCP by transforming the data values inside the Gaussian blobs to lines in the PCP.

6.3 Result

The proposed approach has been tested on a machine with Nvidia GTX480 with 1.5GB memory and a single Intel Core i5 processor with 16GB memory. Due to the restriction of usage of the datasets provided by our collaborators, we created a test dataset by repeating a small 100MB public domain seismic dataset with its five derived attributes three times in the x and y axes and four times in the z axis. The total size of the dataset is then 21.6GB, and we achieved frame rates from 2 FPS to 25 FPS with different settings of transfer functions on a frame buffer of 800×800. The querying time varies from 30 ms to 4 s, which is positively correlated with the number of voxels that passed runtime testing. As shown in Figure 6.2, a channel system and a salt dome structure have been classified using the multivariate TFs. The channel system is colored using an appearance TF with a gray-level color map on the seismic amplitude attribute, and the salt dome structure is colored with a red-to-blue color map on the thickness attribute.

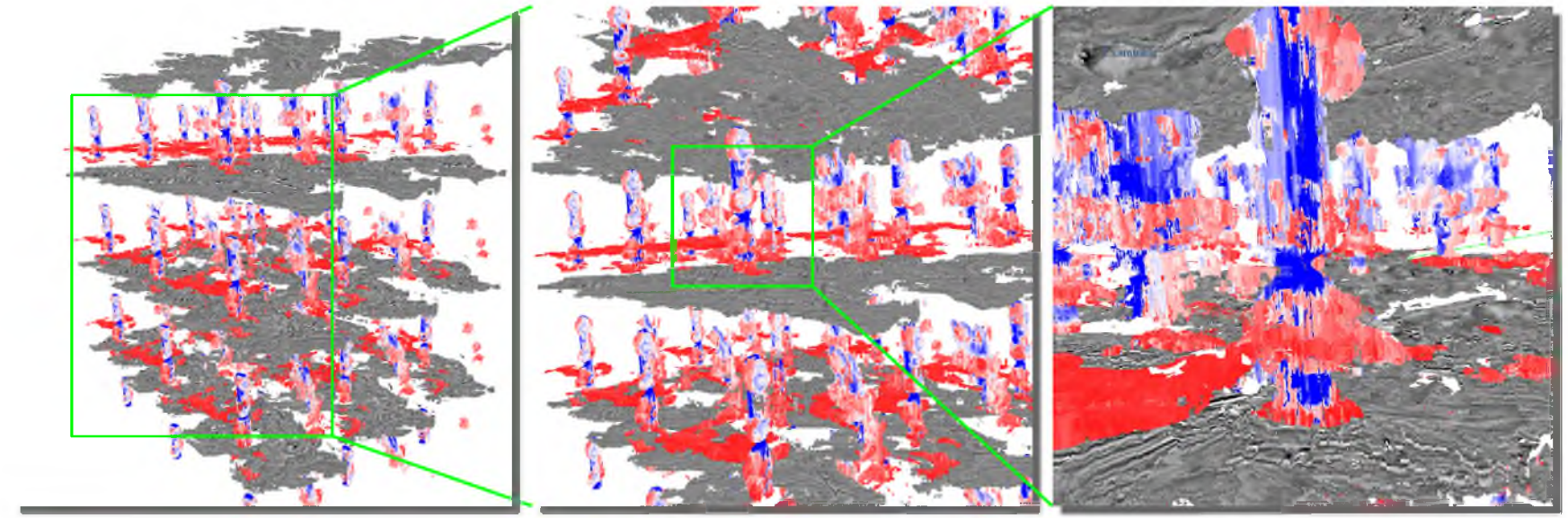


Figure 6.2. Our proposed method allows interactive visualization and efficient query of large multivariate seismic datasets on consumer level PCs. Shown here is a test seismic data of six attributes with size: $1278 \times 1653 \times 1704$.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this dissertation, we have described our progress on the research of multivariate transfer function design. In summary, our contributions are the improvement of the classification ability of transfer functions, intuitive workflows for multivariate volume exploration based on multivariate transfer function design, and interactive out-of-core rendering of large multivariate volumes. The improvement of classification ability of transfer functions is achieved using transfer function combinations. Transfer function combinations use existing transfer function spaces, specifically, the scalar/gradient magnitude transfer function space, the statistical transfer function space, the occlusion transfer function space and the size-based transfer function space. Combinations that have better specificity than the element transfer functions are selected to create a new high-dimensional transfer function space. A moderate amount of precomputation that has been accelerated using GPUs and separable convolution filters allows subsequent interactive design and manipulation of the combined multivariate transfer functions via an intuitive transfer function editor.

A novel multivariate volume exploration workflow has been proposed for more intuitive user interaction with refined feature extraction results. The workflow is designed to facilitate domain users with multivariate transfer function design. Initial transfer function setup is achieved by generating multivariate transfer functions from user-selected samples lassoed directly on slice-based panel views. Transfer functions are then fine-tuned using linked parameter space widgets, including parallel coordinate plots and histograms. The extracted features can be further edited directly on the 2D or 3D view.

GuideME, a slice-guided semiautomatic multivariate volume exploration method,

is a further improvement on the workflow of multivariate volume exploration. In GuideME, the user explores the volume on slices, inspects different attributes via an inspection window and draws guided uncertainty-aware lassos on feature of interest, and then the features are extracted through an automated feature extraction approach whose core is a multivariate transfer function optimization method. More specifically, a boundary confidence measurement that is derived from edge detection provides the user with hints and the uncertainty of feature boundaries. A guided uncertainty-aware lasso that snaps to the feature boundary facilitates region selection. An automated feature extraction method minimizes false positives outside the lasso while preserving true positives inside the lasso. Our experiments have shown that GuideME gives comparable results to those generated by previous methods and expert segmentations, but is more efficient and easier in terms of interaction.

A GPU-based out-of-core method has been proposed to support interactive rendering and efficient query of large multivariate seismic volumes on consumer level PCs. Virtual memory hierarchy is utilized for the realization of interactive rendering. The efficient query is achieved by conducting ellipse/polygon interSections for precomputed Gaussian mixture models of the multivariate data blocks. The method allows the user to efficiently explore large volumes using parameter space multivariate transfer function editors.

All these works share the same research focus: multivariate transfer function design for complex univariate or multivariate datasets. We have demonstrated the usefulness and efficiency of the proposed methods through highly complex real-world datasets, including CT chest scan, MR brain scans and seismic data. Moreover, we have gain positive feedback from our collaborating geophysicists. They find the proposed methods, especially the novel multivariate volume exploration workflows, merit the exploration of complex multivariate seismic data.

Further research on multivariate transfer function design can happen in many ways at different levels. The transfer function combinations are selected using the rules described in Section 3.1.2, and it would be interesting to see robust methods to automatically choose the best transfer function combinations. In

order to improve the classification over a specific region, metric volumes used for further classification steps may be computed locally from the regions already classified instead of being precomputed globally. We would like to investigate how to provide the user more guidance during the volume exploration process by bringing in advanced image processing and machine learning techniques. We might then be able to automatically select an appropriate slice that captures useful features, and furthermore, the features would be automatically highlighted for the user. Time varying datasets are another topic of interest. By exploiting temporal coherence between time steps of simulations or scans, it is possible to automatically propagate and modify already defined multivariate transfer functions. Ultimately, our goal is to develop an interactive, flexible, scalable and intuitive visual analytic environment. Domain users should be able to conduct both qualitative and quantitative analysis on very large and complex volume datasets that have one or more time steps. The visual analytic environment should allow users to focus purely on utilizing their domain knowledge, whereas the laborious or unintuitive procedures, e.g., transfer function design should be conducted automatically by the computer.

APPENDIX

RELATED PUBLICATIONS

- "Transfer Function Combinations"
  Liang Zhou, Mathias Schott and Charles Hansen,
  Computers & Graphics, vol. 36, no. 6, 2012, pp. 596-606.

- "Transfer Function Design based on User Selected Samples for Intuitive Multivariate Volume Exploration"
  Liang Zhou and Charles Hansen,
  In *Proceedings of the IEEE Pacific Visualization Symposium* (PacificVis), 2013, pp. 73-80.

- "GuideME: Slice-guided Multivariate Exploration of Volumes"
  Liang Zhou and Charles Hansen,
  Computer Graphics Forum (EuroVis 2014), vol. 33, no. 3, 2014, pp. 151-160.

- Poster: "A System for More Intuitive Multivariate Volume Exploration and Visualization" (Honorable Mention)
  Liang Zhou and Charles Hansen,
  In *Proceedings of the IEEE Symposium on Large-Scale Data Analysis and Visualization*, 2012.

- Poster: "Interactive Rendering and Efficient Querying for Large Multivariate Seismic Volumes on Consumer Level PCs"
  Liang Zhou and Charles Hansen,
  In *Proceedings of the IEEE Symposium on Large-Scale Data Analysis and Visualization*, 2013.

REFERENCES

[1] Akiba, H., and Ma, K.-L. A tri-space visualization interface for analyzing time-varying multivariate volume data. In *Proceedings of Eurographics/IEEE VGTC Symposium on Visualization* (May 2007), pp. 115–122.

[2] Bachthaler, S., and Weiskopf, D. Continuous scatterplots. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1428–1435.

[3] Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. Toolglass and magic lenses: The see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1993), SIGGRAPH '93, ACM, pp. 73–80.

[4] Bilmes, J. A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Tech. Rep. TR-97-021, ICSI, 1997.

[5] Blaas, J., Botha, C., and Post, F. Extensions of parallel coordinates for interactive exploration of large multi-timepoint data sets. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1436–1451.

[6] Boada, I., Navazo, I., and Scopigno, R. Multiresolution volume visualization with a texture-based octree. *The Visual Computer 17* (2001), 185–197.

[7] Bowman, D. A., Chen, J., Wingrave, C. A., Lucas, J., Ray, A., F, N., Li, Q., Haciahmetoglu, Y., sun Kim, J., Kim, S., Boehringer, R., and Ni, T. New directions in 3d user interfaces. *International Journal of Virtual Reality 5*, 2 (2006), 3–14.

[8] Brandes, U., and Pich, C. Eigensolver methods for progressive multidimensional scaling of large data. In *Proceedings of the 14th International Conference on Graph Drawing* (Berlin, Heidelberg, 2007), GD'06, Springer-Verlag, pp. 42–53.

[9] Bruckner, S., and Gröller, M. E. Style transfer functions for illustrative volume rendering. *Computer Graphics Forum 26*, 3 (2007), 715–724.

[10] Caban, J., and Rheingans, P. Texture-based transfer functions for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1364–1371.

[11] Canny, J. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8*, 6 (Nov 1986), 679–698.

[12] Chopra, S., and Marfurt, K. J. *Seismic Attributes for Prospect ID and Reservoir Characterization (Geophysical Developments No. 11)*. Society Of Exploration Geophysicists, 2007.

[13] Cordier, N., Menze, B., Delingette, H., and Ayache, N. Patch-based segmentation of brain tissues. In *Proceedings of NCI-MICCAI BRATS 2013* (2013), pp. 6–17.

[14] Correa, C., and Ma, K.-L. Size-based transfer functions: A new volume exploration technique. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1380–1387.

[15] Correa, C., and Ma, K.-L. The occlusion spectrum for volume classification and visualization. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov 2009), 1465–1472.

[16] Crassin, C., Neyret, F., Lefebvre, S., and Eisemann, E. Gigavoxels: Ray-guided streaming for efficient and detailed voxel rendering. In *Proceedings of the 2009 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2009), I3D '09, ACM, pp. 15–22.

[17] Cui, W., Zhou, H., Qu, H., Wong, P. C., and Li, X. Geometry-based edge clustering for graph visualization. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov.-Dec. 2008), 1277 –1284.

[18] D'Agostino, R. B., Belanger, A., and D'Agostino, Ralph B., J. A suggestion for using powerful and informative tests of normality. *The American Statistician 44*, 4 (1990), 316–321.

[19] de Silva, V., and Tenenbaum, J. B. Sparase multidimensional scaling using landmark points. Tech. rep., Stanford University.

[20] Dijkstra, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik 1* (1959), 269–271.

[21] Doleisch, H. Simvis: Interactive visual analysis of large and time-dependent 3d simulation data. In *Proceedings of the 39th Conference on Winter Simulation* (Piscataway, NJ, USA, 2007), WSC '07, IEEE Press, pp. 712–720.

[22] Elmqvist, N., Dragicevic, P., and Fekete, J. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov 2008), 1539–1148.

[23] Engel, K. CERA-TVR: A framework for interactive high-quality teravoxel volume visualization on standard pcs. In *Posters at Large-Data Analysis and Visualization 2011 LDAV 2011* (2011).

[24] Faloutsos, C., and Lin, K.-I. Fastmap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. *SIGMOD Rec. 24*, 2 (May 1995), 163–174.

[25] Farahani, F., Reyes, M., Menze, B., Gerstner, E., Kirby, J., and Kalpathy-Cramer, J. NCI-MICCAI 2013 Challenge on Multimodal Brain Tumor Segmentation (BRaTS). The challenge database contains fully anonymized images from the following institutions: ETH Zurich, University of Bern, University of Debrecen, and University of Utah and publicly available images from the Cancer Imaging Archive (TCIA).

[26] Fua, Y.-H., Ward, M., and Rundensteiner, E. Hierarchical parallel coordinates for exploration of large datasets. In *Visualization '99. Proceedings* (Oct. 1999), pp. 43 –508.

[27] Gerig, G., Kubler, O., Kikinis, R., and Jolesz, F. Nonlinear anisotropic filtering of mri data. *Medical Imaging, IEEE Transactions on 11*, 2 (Jun 1992), 221–232.

[28] Gobbetti, E., Marton, F., and Iglesias Guitin, J. A. A single-pass gpu ray casting framework for interactive out-of-core rendering of massive volumetric datasets. *The Visual Computer 24* (2008), 797–806.

[29] Guennebaud, G., Jacob, B., et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[30] Guo, H., Mao, N., and Yuan, X. Wysiwyg (what you see is what you get) volume visualization. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (Dec 2011), 2106–2114.

[31] Guo, H., Xiao, H., and Yuan, X. Multi-dimensional transfer function design based on flexible dimension projection embedded in parallel coordinates. In *Pacific Visualization Symposium (PacificVis), 2011 IEEE* (March 2011), pp. 19–26.

[32] Guo, H., Xiao, H., and Yuan, X. Scalable multivariate volume visualization and analysis based on dimension projection and parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 18*, 9 (Sept. 2012), 1397 –1410.

[33] Guthe, S., and Strasser, W. Advanced techniques for high-quality multi-resolution volume rendering. *Computers & Graphics 28*, 1 (2004), 51 – 58.

[34] Hadwiger, M., Berger, C., and Hauser, H. High-quality two-level volume rendering of segmented data sets on consumer graphics hardware. In *Visualization, 2003. VIS 2003. IEEE* (Oct. 2003), pp. 301 –308.

[35] Hadwiger, M., Beyer, J., Jeong, W.-K., and Pfister, H. Interactive volume exploration of petascale microscopy data streams using a visualization-driven virtual memory approach. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec. 2012), 2285 –2294.

[36] Hadwiger, M., Kniss, J. M., Rezk-salama, C., Weiskopf, D., and Engel, K. *Real-time Volume Graphics*. A. K. Peters, Ltd., Natick, MA, USA, 2006.

[37] Haidacher, M., Bruckner, S., Kanitsar, A., and Gröller, M. E. Information-based transfer functions for multimodal visualization. In *VCBM* (Oct. 2008), W. N. C.P Botha, G. Kindlmann and B. Preim, Eds., Eurographics Association, pp. 101–108.

[38] Haidacher, M., Patel, D., Bruckner, S., Kanitsar, A., and Groller, M. Volume visualization based on statistical transfer-function spaces. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE* (March 2010), pp. 17–24.

[39] Harrower, M., and Brewer, C. A. Colorbrewer.org: An online tool for selecting colour schemes for maps. *Cartographic Journal 40*, 1 (Jun 2003), 27–37.

[40] Hartigan, J. A., and Wong, M. A. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 28*, 1 (1979), pp. 100–108.

[41] Heinrich, J., and Weiskopf, D. Continuous parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov 2009), 1531–1538.

[42] Hoberock, J., and Bell, N. Thrust: A C++ template library of parallel algorithms. http://thrust.github.io/.

[43] Höllt, T., Beyer, J., Gschwantner, F., Muigg, P., Doleisch, H., Heinemann, G., and Hadwiger, M. Interactive seismic interpretation with piecewise global energy minimization. In *Pacific Visualization Symposium (PacificVis), IEEE 2011* (March 2011), pp. 59–66.

[44] Höllt, T., Freiler, W., Gschwantner, F., Doleisch, H., Heinemann, G., and Hadwiger, M. Seivis: An interactive visual subsurface modeling application. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec. 2012), 2226–2235.

[45] Holten, D. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (Sept.-Oct. 2006), 741 –748.

[46] Holten, D., and van Wijk, J. J. Force-directed edge bundling for graph visualization. *Computer Graphics Forum 28*, 3 (2009), 983–990.

[47] Viscontest2004 datasets. http://vis.computer.org/vis2004contest/.

[48] ImageVis3D: A Real-time Volume Rendering Tool for Large Data. Scientific Computing and Imaging Institute (SCI).

[49] Inselberg, A. The plane with parallel coordinates. *The Visual Computer 1* (1985), 69–91.

[50] Ip, C. Y., Varshney, A., and JaJa, J. Hierarchical exploration of volumes using multilevel segmentation of the intensity-gradient histograms. *IEEE Transactions on Visualization and Computer Graphics 18* (2012), 2355–2363.

[51] Jankun-Kelly, T. J., and Ma, K.-L. Visualization exploration and encapsulation via a spreadsheet-like interface. *IEEE Transactions on Visualization and Computer Graphics 7*, 3 (July 2001), 275–287.

[52] Jarque, C. M., and Bera, A. K. Efficient tests for normality, homoscedasticity and serial independence of regression residuals. *Economics Letters 6*, 3 (1980), 255 – 259.

[53] Jolliffe, I. T. *Principal Component Analysis*. Springer, 2002.

[54] Kindlmann, G., and Durkin, J. Semi-automatic generation of transfer functions for direct volume rendering. In *Volume Visualization, 1998. IEEE Symposium on* (Oct 1998), pp. 79–86.

[55] Kindlmann, G., Whitaker, R., Tasdizen, T., and Moller, T. Curvature-based transfer functions for direct volume rendering: Methods and applications. In *Visualization, 2003. VIS 2003. IEEE* (Oct 2003), pp. 513–520.

[56] Kniss, J., Kindlmann, G., and Hansen, C. Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (Jul 2002), 270–285.

[57] Kniss, J., Premoze, S., Ikits, M., Lefohn, A., Hansen, C., and Praun, E. Gaussian transfer functions for multi-field volume visualization. In *Visualization, 2003. VIS 2003. IEEE* (Oct 2003), pp. 497–504.

[58] Knoll, A., Wald, I., Parker, S., and Hansen, C. Interactive isosurface ray tracing of large octree volumes. In *Interactive Ray Tracing 2006, IEEE Symposium on* (Sept. 2006), pp. 115 –124.

[59] LaMar, E., Hamann, B., and Joy, K. I. Multiresolution techniques for interactive texture-based volume visualization. In *Proceedings of the Conference on Visualization '99: Celebrating Ten Years* (Los Alamitos, CA, USA, 1999), VIS '99, IEEE Computer Society Press, pp. 355–361.

[60] Levoy, M. Display of surfaces from volume data. *Computer Graphics and Applications, IEEE 8*, 3 (May 1988), 29–37.

[61] Lindemann, F., and Ropinski, T. About the influence of illumination models on image comprehension in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (Dec 2011), 1922–1931.

[62] LLNL. Visit visualization tool, 2012.

[63] Lum, E. B., Shearer, J., and Ma, K.-L. Interactive multi-scale exploration for volume classification. *The Visual Computer 22*, 9-11 (2006), 622–630.

[64] Lundstrom, C., Ljung, P., and Ynnerman, A. Local histograms for design of transfer functions in direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 12*, 6 (Nov 2006), 1570–1579.

[65] Maciejewski, R., Woo, I., Chen, W., and Ebert, D. Structuring feature space: A non-parametric method for volumetric transfer function generation. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov 2009), 1473–1480.

[66] Malandain, G. Connexe: Connected component extraction. http://www-sop.inria.fr/epidaure/software/basics/connexe.php.

[67] Marks, J., Andalman, B., Beardsley, P. A., Freeman, W., Gibson, S., Hodgins, J., Kang, T., Mirtich, B., Pfister, H., Ruml, W., Ryall, K., Seims, J., and Shieber, S. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 389–400.

[68] McDonnell, K. T., and Mueller, K. Illustrative parallel coordinates. *Computer Graphics Forum 27*, 3 (2008), 1031–1038.

[69] Morariu, V. I., Srinivasan, B. V., Raykar, V. C., Duraiswami, R., and Davis, L. S. Automatic online tuning for fast gaussian summation. In *Advances in Neural Information Processing Systems (NIPS)* (2008).

[70] Novotny, M., and Hauser, H. Outlier-preserving focus+context visualization in parallel coordinates. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (Sept 2006), 893–900.

[71] NVIDIA, C. Nv path rendering OpenGL extension. https://developer.nvidia.com/nv-path-rendering.

[72] Osirix dicom sample image sets. http://pubimage.hcuge.ch:8080/.

[73] Patel, D., Bruckner, S., Viola, I., and Groller, E. Seismic volume visualization for horizon extraction. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE* (March 2010), pp. 73 –80.

[74] Patel, D., Haidacher, M., Balabanian, J.-P., and Groller, E. Moment curves. In *Visualization Symposium, 2009. PacificVis '09. IEEE Pacific* (April 2009), pp. 201–208.

[75] Paulovich, F., Silva, C., and Nonato, L. Two-phase mapping for projecting massive data sets. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov.-Dec. 2010), 1281 –1290.

[76] Perona, P., and Malik, J. Scale-space and edge detection using anisotropic diffusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 12*, 7 (Jul 1990), 629–639.

[77] Piringer, H., Kosara, R., and Hauser, H. Interactive focus+context visualization with linked 2d/3d scatterplots. In *Coordinated and Multiple Views in Exploratory Visualization, 2004. Proceedings. Second International Conference on* (July 2004), pp. 49 – 60.

[78] Prassni, J.-S., Ropinski, T., Mensmann, J., and Hinrichs, K. Shape-based transfer functions for volume visualization. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE* (March 2010), pp. 9–16.

[79] Rezk-Salama, C. *Volume Rendering Techniques for General Purpose Graphics Hardware*. PhD thesis, 2002.

[80] Ropinski, T., Prass ni, J., Steinicke, F., and Hinrichs, K. Stroke-based transfer function design. In *Proceedings of the Fifth Eurographics / IEEE VGTC Conference on Point-Based Graphics* (Aire-la-Ville, Switzerland, Switzerland, 2008), SPBG'08, Eurographics Association, pp. 41–48.

[81] Rübel, O., Prabhat, Wu, K., Childs, H., Meredith, J., Geddes, C. G. R., Cormier-Michel, E., Ahern, S., Weber, G. H., Messmer, P., Hagen, H., Hamann, B., and Bethel, E. W. High performance multivariate visual data exploration for extremely large data. In *Proceedings of the 2008 ACM/IEEE Conference on Supercomputing* (Piscataway, NJ, USA, 2008), SC '08, IEEE Press, pp. 51:1–51:12.

[82] Salama, C., Keller, M., and Kohlmann, P. High-level user interfaces for transfer function design with semantics. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (Sept 2006), 1021–1028.

[83] Sato, Y., Westin, C. F., Bhalerao, A., Nakajima, S., Shiraga, N., Tamura, S., and Kikinis, R. Tissue classification based on 3d local intensity structures for volume rendering. *IEEE Transactions on Visualization and Computer Graphics 6*, 2 (Apr 2000), 160–180.

[84] Schott, M., Pegoraro, V., Hansen, C. D., Boulanger, K., and Bouatouch, K. A directional occlusion shading model for interactive direct volume rendering. *Computer Graphics Forum 28*, 3 (2009), 855–862.

[85] Seo, J., and Shneiderman, B. A rank-by-feature framework for interactive exploration of multidimensional data. *Information Visualization 4*, 2 (July 2005), 96–113.

[86] Sereda, P., Bartroli, A., Serlie, I. W. O., and Gerritsen, F. Visualization of boundaries in volumetric data sets using lh histograms. *IEEE Transactions on Visualization and Computer Graphics 12*, 2 (March 2006), 208–218.

[87] Silverman, B. W. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, 1986.

[88] Tenenbaum, J. B., de Silva, V., and Langford, J. C. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science 290*, 5500 (Dec. 2000), 2319–2323.

[89] Tory, M., Potts, S., and Moller, T. A parallel coordinates style interface for exploratory volume visualization. *IEEE Transactions on Visualization and Computer Graphics 11*, 1 (Jan 2005), 71–80.

[90] Tzeng, F.-Y., Lum, E., and Ma, K.-L. A novel interface for higher-dimensional classification of volume data. In *Visualization 2003. VIS 2003, IEEE.* (Oct 2003), pp. 505–512.

[91] Tzeng, F.-Y., Lum, E., and Ma, K.-L. An intelligent system approach to higher-dimensional classification of volume data. *IEEE Transactions on Visualization and Computer Graphics 11*, 3 (May 2005), 273–284.

[92] Tzeng, F.-Y., and Ma, K.-L. A cluster-space visual interface for arbitrary dimensional classification of volume data. In *Proceedings of the Sixth Joint Eurographics - IEEE TCVG Conference on Visualization* (Aire-la-Ville, Switzerland, Switzerland, 2004), VISSYM'04, Eurographics Association, pp. 17–24.

[93] Volume rendering engine. http://www.voreen.org/.

[94] Vuccini, E., Patel, D., and Gröller, M. E. Enhancing visualization with real-time frequency-based transfer functions. In *Proceedings of IS&T/SPIE Conference on Visualization and Data Analysis* (jan 2011), 7868, IS&T/SPIE, pp. 78680L–1–78680L–12.

[95] Wan, Y., Otsuna, H., Chien, C.-B., and Hansen, C. Interactive extraction of neural structures with user-guided morphological diffusion. In *Biological Data Visualization (BioVis), 2012 IEEE Symposium on* (Oct 2012), pp. 1–8.

[96] Wang, Y., Chen, W., Zhang, J., Dong, T., Shan, G., and Chi, X. Efficient volume exploration using the gaussian mixture model. *IEEE Transactions on Visualization and Computer Graphics 17*, 11 (Nov 2011), 1560–1573.

[97] Weiler, M., Westermann, R., Hansen, C., Zimmermann, K., and Ertl, T. Level-of-detail volume rendering via 3d textures. In *Proceedings of the 2000 IEEE Symposium on Volume Visualization* (New York, NY, USA, 2000), VVS '00, ACM, pp. 7–13.

[98] Welch, B. L. The generalization of 'student's' problem when several different population variances are involved. *Biometrika 34*, 1/2 (1947), 28–35.

[99] Wu, Y., and Qu, H. Interactive transfer function design based on editing direct volume rendered images. *IEEE Transactions on Visualization and Computer Graphics 13*, 5 (sept.-oct. 2007), 1027 –1040.

[100] Yuan, X., Guo, P., Xiao, H., Zhou, H., and Qu, H. Scattering points in parallel coordinates. *Visualization and Computer Graphics, IEEE Transactions on 15*, 6 (Nov. 2009), 1001–1008.

[101] Zhao, X., and Kaufman, A. Multi-dimensional reduction and transfer function design using parallel coordinates. In *Volume Graphics 2010, IEEE/EG International Symposium on* (may. 2010), pp. 69–76.

[102] Zhou, H., Yuan, X., Qu, H., Cui, W., and Chen, B. Visual clustering in parallel coordinates. *Computer Graphics Forum 27*, 3 (2008), 1047–1054.

[103] Zhou, L., and Hansen, C. Transfer function design based on user selected samples for intuitive multivariate volume exploration. In *Pacific Visualization Symposium (PacificVis), 2013 IEEE* (March 2013), pp. 73–80.