Transfer Function Design based on User Selected Samples for Intuitive Multivariate Volume Exploration

Liang Zhou*
SCI Institute and the School of Computing,
University of Utah

Charles Hansen†
SCI Institute and the School of Computing,
University of Utah

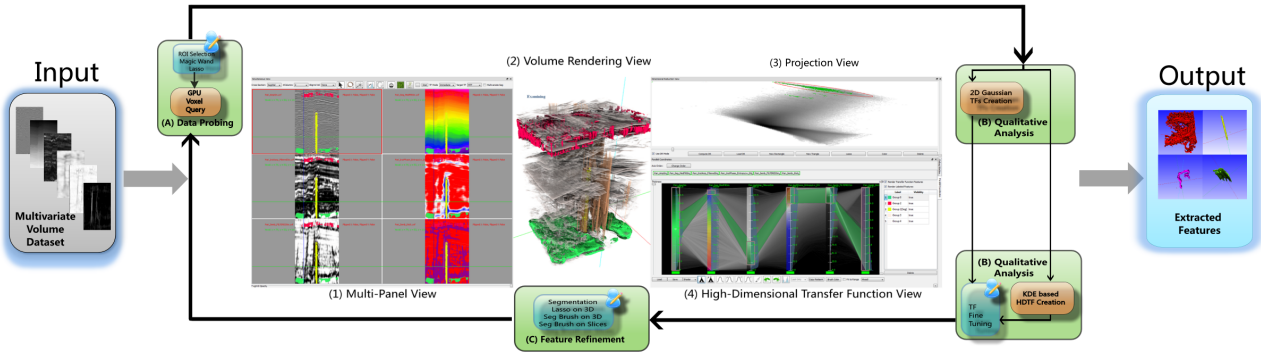


Figure 1: The user interface and the work flow of the system implementing our proposed method. Four closely linked views are shown and labeled, namely: (1) multi-panel view, (2) volume rendering view, (3) projection view and (4) high-dimensional transfer function view. Three stages: (A) data probing, (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.

Abstract

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 data sets.

Index Terms: Computer Graphics [I.3.7]: Three-Dimensional Graphics and Realism—Color, shading, shadowing and texture; Image Processing and Computer Vision [I.4.10]: Image Representation—Volumetric

1 Introduction

Multivariate dataset visualization has been an active research area for the past decade and is still 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 [6, 1, 4, 10]. 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 result 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., multi-variate 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.

In this paper, 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

*e-mail: zhoul@cs.utah.edu
†e-mail: hansen@cs.utah.edu

IEEE Pacific Visualization Symposium 2013
26 February - 1 March, Sydney, NSW, Australia
978-1-4673-4797-6/13/$31.00 ©2013 IEEE

and robustly generated from the queried data samples via the kernel density estimation (KDE) [26] method. The TFs are represented by parallel coordinate plots (PCPs) and can be interactively modified in a 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 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 followings: we propose a transfer function design method for multivariate volume visualization based on user selected samples, specifically: a HDTF generation method based on KDE and a Gaussian mixture model based 2D Gaussian TF generation method. Second, 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.

2 Related Work

Transfer Function Design

Volume datasets can be explored using transfer functions. A 1D TF that uses scalar values of the volume or a 2D TF that has the gradient magnitude of the volume as a second property for better classification [15] are most frequently used. The TFs can be interactively defined by 1D TF widgets or 2D TF widgets proposed by Kniss *et al.* [16]. However, to design a good TF, the user has to manipulate the TF widgets in the value space and check result in the volume rendering view which is laborious and time consuming. To address this issue, researchers have proposed to automate the TF generation process. Maciejewski *et al.* [19] utilize KDE to structure the data value space to generate initial TFs. Note that instead of performing KDE over a 2D value space of the whole data as in [19], our proposed approach applies KDE over samples selected by the user to robustly generate HDTFs. Also focusing on the value space, Wang *et al.* [28] initialize TFs by modeling the data value space using Gaussian mixture model and render the extracted volume with pre-integrated volume rendering. Our automated Gaussian TFs are similar to their work, however, our working space is created from multidimensional reduction while theirs is traditional 2D TF space. Alternatively, the volume exploration system proposed by Guo *et al.* [9] allows the user to directly manipulate on the volume rendering view with intuitive screen space stroking tools similar to 2*D* photo processing applications. In contrast to our work, all methods mentioned above work on volumes with only one or two attributes.

Multidimensional Data Visualization

Visualizing and understanding multidimensional datasets has been an active research topic in information visualization. Scatter plot matrices, parallel coordinate plots [13] and star-glyphs [29] are common approaches for discrete multidimensional data visualization. An efficient rendering [21] method has been proposed to make meaningful and interactive visualization of large multidimensional datasets possible. 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) [14], multidimensional scaling (MDS), isomap [27], and Fastmap [7]. We employ Fastmap due to its speed, stability and simplicity.

Linked View Systems

Multivariate volume datasets can be explored using linked view systems which have shown to be useful for multivariate simulation data exploration. The SimVis system [6, 24] 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. Akiba and Ma [1] propose a tri-space exploration technique involving PCPs together with time histograms to help the design of HDTFs for time-varying multivariate volume datasets. Blaas *et al.* [4] extent parallel coordinates for interactive exploration of large multi-time point datasets rendered as isosurfaces. More recently, Zhao and Kaufman [30] combine multi-dimensional reduction and TF design using parallel coordinates but, their system is only able to handle very small datasets. Guo *et al.* [10] propose an interactive HDTF design framework using both continuous PCPs and multidimensional scaling technique accelerated by employing an octree structure. However, we have observed two limitations in the above systems: 1) the user has to explore the data via interactions on the TF 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 TFs and it is difficult to achieve a more refined result. As such, we have implemented a linked view system that improves on these two issues to allow the domain user to explore complex real-world datasets more intuitively with more refined results.

3 Method Overview

The work flow of our proposed method as shown in Figure 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 lasso tool or "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., 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. In addition, automated 2D Gaussian TFs on the projection view offers 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 which are applied directly on the volume rendering view or the multi-panel view.

4 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 as such a fast PCP generation method is needed.

4.1 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.* [10]. However, Knoll *et al.* [17] 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 [17] agrees with this statement. As such, we 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`.

$$v_X \rightarrow \{(y_1,y_2),(y_2,y_3),...,(y_{N-1},y_N)\}$$
$$\text{where } \mathrm{TF}(y_1,y_2,\ldots,y_N).a > 0 \quad (1)$$

$(y_1,y_2),(y_2,y_3),...,(y_{N-1},y_N)$ 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 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 Parallel Coordinate Plots Generation

As proposed in [21], Figure 2 shows that each non-zero 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

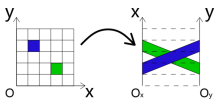


Figure 2: Generating a PCP from a joint histogram.

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 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 $C(i,j)$.

$$C(i,j) = C_0(i,j)\frac{\log(v(i,j))}{\log(\sum v)} \quad (2)$$

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.

5 Transfer Function Generation from User Selected Samples

In this section, the actual TF generation method will be explained. Section 5.1 introduces the method for interactive voxel sample selection, Section 5.2 discusses the KDE based HDTF generation method and Section 5.3 details on the automated 2D Gaussian TF on the dimensionality reduced space.

5.1 Sample Selection in the Multi-panel View

The user can interactively select arbitrary a 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 which 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 anisotropic diffusion [23]. Equation 3 describes the diffusion equation where $S(t,x,y)$ is the number of seeds at position (x,y) at time t, $V(t,x,y)$ being the intensity of the chosen attribute at the same point, $|\nabla V(t,x,y)|$ is its gradient magnitude, and $g(s)$ being a conductivity term.

$$\frac{\partial S(t,x,y)}{\partial t} = div(g(|\nabla V(t,x,y)|)\nabla S(t,x,y)) \quad (3)$$
$$\text{where } g(s) = v \cdot \exp^{\frac{-s^2}{K^2}}$$

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 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 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.

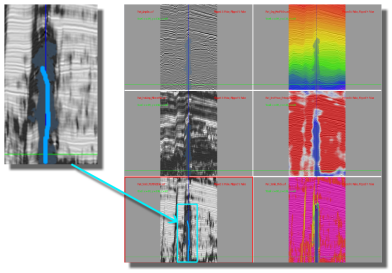


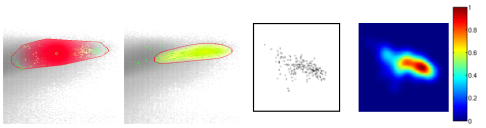
Figure 3: The user draws on a salt dome (stroke shown in light blue) over the fifth attribute in the panel view resulting in the dark blue region of selection.

5.2 Kernel Density Estimation based Transfer Function Generation

We would like to generate HDTFs from the samples selected using method described in Section 5.1. To reduce the computational complexity, we separate 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 intents 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(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) [26] seen in Equation 4 is a non-parametric method for estimating the density function $f_h(x)$ at



(a) (b) (c)

Figure 4: 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.

location x of an arbitrary dimensional domain Ω with given samples $\{x_i\}, i \in \{1,2,3,\ldots,n\}$.

$$f_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x-x_i) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x-x_i}{h}), x, x_i \in \Omega \quad (4)$$

where $K(x)$ being 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 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 [26], which can be extended to 2D:

$$h = 1.06\sqrt{\det\Sigma}\cdot n^{-\frac{1}{5}} \quad (5)$$

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:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{||x||^2}{2}} \quad (6)$$

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 outliers. As a result, the density value $f_h(x)$ around the outliers is lower than that of the intended samples.

Figure 4(c) shows the density function generated by the KDE method of the given samples with the above settings. It verifies our expectation that the outliers have lower density than the intended sample regions. As such, we can discard the outliers by setting a threshold for the density value $f_h(x)$. Figure 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.

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.

5.3 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 [7] 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, while the HDTFs are better for features that have subtle differences in the HD value domain.

5.3.1 Dimensional Reduction using Fastmap

We employ Fastmap [7] as the dimensional reduction technique since it is fast, stable and easy to implement. Fastmap is a recursive algorithm for multidimensional 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 can be summarized as the following:

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

---

if $k \leq 0$ **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:
$X[i,col] = x_i = \frac{D(O_a,O_i)^2 + D(O_a,O_b)^2 - D(O_b,O_i)^2}{2D(O_a,O_b)}, i \in \{0,1,2,...,N-1\}$
Call FastMap($k-1, D'(), O$).
Where
$D'(O'_i, O'_j)^2 = D(O_i,O_j)^2 - (x_i - x_j)^2, i, j \in \{0,1,2,...,N-1\}$

5.3.2 Gaussian Mixture Model with Expectation Maximization

Assuming that all attributes we are handling are continuous measurements, the dimensionality reduced 2D value space can therefore 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 7 with mean value μ being a 2D vector and covariance matrix Σ as a 2×2 matrix.

$$N(x|\mu,\Sigma) = \frac{1}{2\pi|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \quad (7)$$

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

$$p(x|\theta) = \sum_{j=1}^{k} \alpha^j N(x|\mu^j,\Sigma^j) \quad (8)$$

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 $p(X|\theta)$

$$\hat{\theta} = \arg\max p(X|\theta) = \arg\max \prod_{i=1}^{n} p(x^i|\theta) \quad (9)$$

where n is the number of input points. Equation 9 can be solved by the expectation maximization (EM) algorithm [3]. Given an initial setup of θ, the EM 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 [12] 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.

5.3.3 Automated 2D Gaussian Transfer Functions

We use a modified TF generation scheme as in [28] but ours differs in 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 [28], and 2) we use the user selected samples as the input point clouds while 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:

$$\alpha = \alpha_{\max} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \tag{10}$$

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:

$$\Sigma = VDV^{-1} \tag{11}$$

$$\Sigma^{1/2} = VD^{1/2}V^{-1} \tag{12}$$

where D is a diagonal matrix holding the eigen values and V contains the eigen vectors as columns. V is an orthogonal matrix, i.e. $V^{-1} = V^T$, since Σ is symmetric. The eigen values σ_1, σ_2 are the radii of the principal axes of the ellipse, while the eigen vectors 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 eigen values and eigen vectors. 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 eigen vectors in V with an angle β. Finally, multiplying the eigen values σ_1, σ_2 with a scaling factor (s_a, s_b) results in the scaling of the widget.

6 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.

(a) (b) (c)

Figure 5: Feature refinement tools: (a) 3D brush, (b) 3D lasso and (c) 2D brush.

Screen Space Brush in the 3D View. The tool as seen in Figure 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 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.

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 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.

Refinement Brush in the Panel View. The refinement can also be done by seeding on the panel view via drawing strokes (Figure 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 8.2, and thus similar to TF groups, their colors can be changed, they can be deleted and their visibility can be toggled.

7 Rendering

We employ the directional occlusion shading (DOS) [25], which is an efficient approximation to ambient occlusion as the rendering technique because that the DOS is gradient-free and provides the user more insights into the dataset than local shading models as shown on seismic datasets [22]. A user study conducted by Lindemann and Ropinski [18] 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 as such a simplified method of [11] 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.

8 User Interface

The user interface of our system is seen in Figure 1 where a multi-panel 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 as such any updates in one view will be reflected in others.

8.1 Multi-panel Viewer

We have developed a multi-panel view which shows all attributes of a slice by placing attributes into individual panels as seen in the left part of Figure 1 as well as in Figure 3. The multi-panel 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 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.

8.2 HDTF Editor

The user can interact with the HDTF editor to manually modify the HDTFs. Figure 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 while the pair-wise 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 is useful since not all attributes provide positive assistance in the extraction of specific features and this knowledge is usually not known before hand. 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 9. The group manager manages all TF and segment groups. One is able to toggle the visibility or remove individual or a batch of groups conveniently.

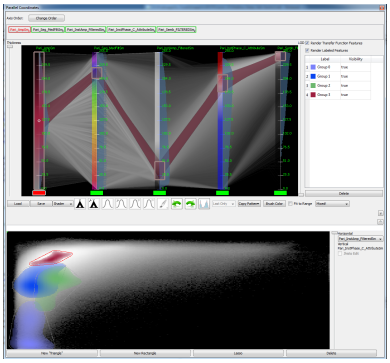


Figure 6: The HDTF editor. Note that the first attribute, seismic amplitude, is locked.

As seen in Figure 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 opacity gradient.

In addition to the PCP TF editor, a pair-wise 2D TF editor is used to aid the exploration of pair-wise features. The pair-wise 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.

8.3 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 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.

9 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 data set 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.

9.1 Hurricane Isabel Simulation

We have experimented with the proposed system on the simulation dataset: hurricane Isabel. The hurricane Isabel dataset, introduced by IEEE visualization contest 2004, is a multivariate multiple time step atmospheric simulation data. Eight attributes of time step 25 are used to generate the result in Figure 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

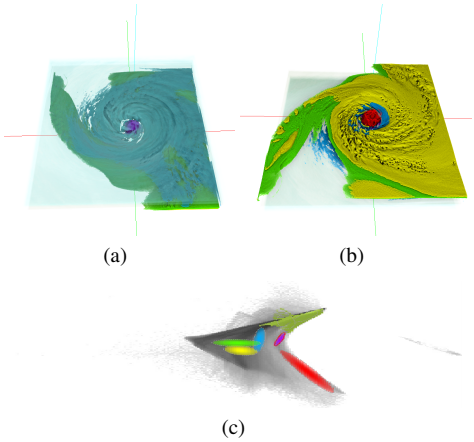


Figure 7: The extracted features shown in (a) the top view and (b) the bottom view. Seen in (c) is the corresponding projection view with automated Gaussian TFs that produce the classification result.

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 as seen in Figure 7. Joint histograms could be generated with continuous scatter plots [2]. The user can generate the result in Figure 7 by placing several large lassos on slices in the axial view (slices

indexed by the z axis) on the multi-panel viewer in the data probing stage. The GMM-EM algorithm explained in section 5.3 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 7. Due to the nature of this data, no feature refinement is required.

The results are similar compared to previous methods. With previous methods [6, 1, 4, 10], one has to carefully design the TFs one by one for each feature, either by editing pairs of histograms [6], or PCP-based HDTF [1, 4] or high-dimensional Gaussian TF and MDS-based TF [10]. Our method, however, allows the user to extract the same features by simply drawing several large lassos across the features on the multi-panel viewer, which is significantly easier.

9.2 3D Seismic Dataset

3D seismic imaging has been the standard for oil and gas exploration for decades, and more recently, multi-attribute volumes derived from the seismic amplitude volume have been used to aid the understanding of the seismic surveys [5]. However, these derived volumes are visualized individually in current seismic data analysis tools and as such 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 is 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 8. Note that for all features, `Amp` provides only context and is

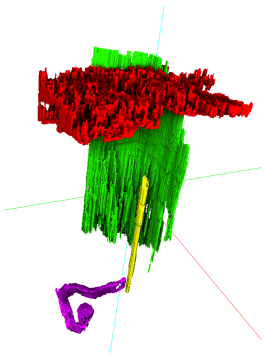


Figure 8: Extracted geological features: a shallow channel complex in red, a salt dome shown in yellow, a deeper channel shown in purple and the largest fault in green.

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`. In the data probing stage, a lasso around the channel complex is drawn on the `Amp` attribute seen in Figure 9(a), from this an HDTF is generated with the KDE method described and fine tuned in the qualitative analysis stage as shown in Figure 9(c). The main connected component as shown in Figure 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 (slices indexed by the y axis) and as such 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 9(d), Gaussian TFs are automatically generated in the projection view. The visualization of the isolated salt dome seen in Figure 9(e) is created by enlarging the Gaussian widget (Figure 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 9(g)), and fine tuning the HDTF as seen in Figure 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 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 9(j)). Since that the faults depend only on the `Semb_Thick` attribute, it is fine tuned to cover the entirety of the faults (Figure 9(l)). The largest fault as seen in Figure 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.

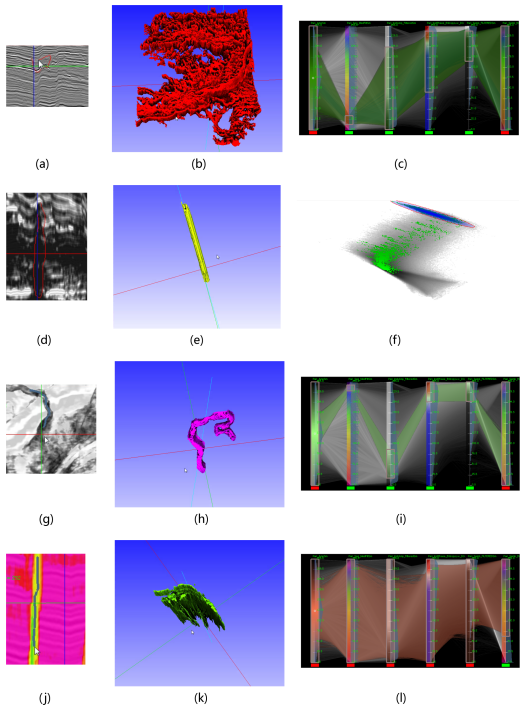


Figure 9: Refined features shown in the middle column with the user's selection of regions of interest shown in the left column, and their TFs shown to the right. Note that the color of the refined features are independent of their TF colors.

10 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 [20] is utilized for efficient kernel density estimation. The linear algebra operations are aided by the Eigen library [8].

11 Conclusion and Future Work

We have presented a TF design method to provide a more intuitive multivariate volume exploration experience with refined feature extraction results. An interactive system is built to realize these proposed methods. Results from the multi-attribute hurricane simulation and 3D seismic survey data sets demonstrate that the system is able to extract features in both commonly used simulation data and highly complicated real-world data. The applications are domain dependent and require domain knowledge, thus our tool is interactive and empirical, and our collaborating geophysicists found merit in the system.

For future work, we would like to further develop our system in three ways: 1) improve the scalability, 2) further reduce the user's workload using advanced machine learning methods, and 3) support time varying datasets. GPU-based out-of-core methods for volume rendering, conditional histogram computation and PCP rendering could support full size 3D seismic data. The user interaction of the system is intuitive as reported by the geophysicists, but is still time consuming for very complicated data sets. By introducing advanced machine learning methods, we hope to make the interaction more automated, e.g. an appropriate slice that captures useful features can be automatically found for the user. Our work can now be applied only to one time step of a simulation. However, thanks to the temporal coherence between the simulation steps, and the user defined TFs on one time step can be propagated to other time steps with incremental update methods.

Acknowledgements

This research was sponsored by the DOE NNSA Award DE-NA0000740, KUS-C1-016-04 made by King Abdullah University of Science and Technology (KAUST), DOE SciDAC Institute of Scalable Data Management Analysis and Visualization DOE DE-SC0007446, NSF OCI-0906379, NSF IIS-1162013, NIH-1R01GM098151-01.

References

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

[2] S. Bachthaler and D. Weiskopf. Continuous scatterplots. *IEEE Transaction on Visualization and Computer Graphics*, 14(6):1428 –1435, nov.-dec. 2008.

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

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

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

[6] H. Doleisch. Simvis: interactive visual analysis of large and time-dependent 3d simulation data. In *Proceedings of the 39th conference on Winter simulation*, WSC '07, pages 712–720, Piscataway, NJ, USA, 2007. IEEE Press.

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

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

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

[10] H. Guo, H. Xiao, and X. Yuan. Multi-dimensional transfer function design based on flexible dimension projection embedded in parallel coordinates. In *Pacific Visualization Symposium*, 2011.

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

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

[13] A. Inselberg. The plane with parallel coordinates. *The Visual Computer*, 1:69–91, 1985.

[14] I. T. Jolliffe. *Principal Component Analysis (Springer Series in Statistics)*. Springer, 2002.

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

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

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

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

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

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

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

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

[23] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):1364–1371, 1990.

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

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

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

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

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

[29] M. O. Ward. Xmdvtool: integrating multiple methods for visualizing multivariate data. In *Proceedings of the conference on Visualization '94*, VIS '94, pages 326–333, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

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