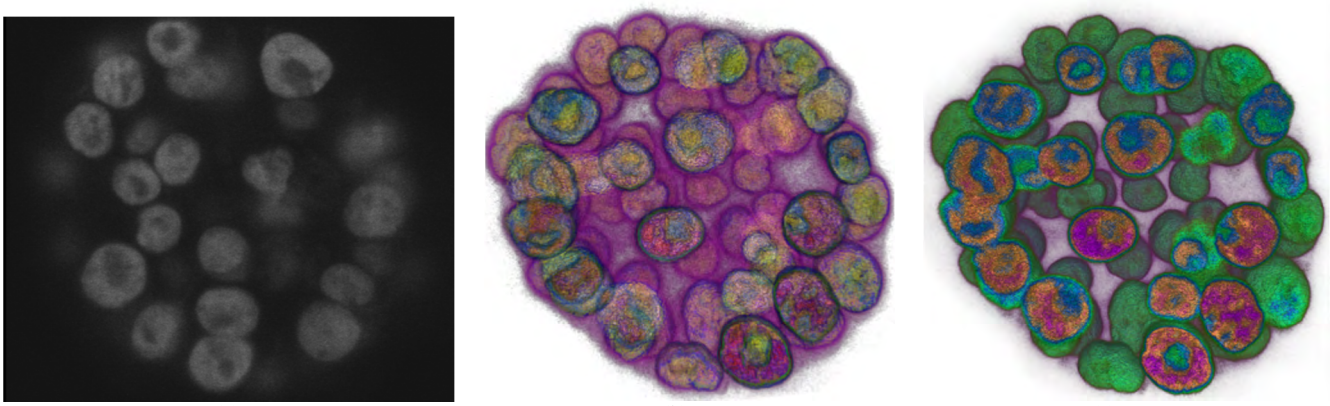
Quick2Insight: A User-Friendly Framework for Interactive Rendering of Biological Image Volumes

Yanling Liu[1]
SAIC-Frederick, Inc.

Curtis Lisle[2]
KnowledgeVis LLC

Jack Collins[3]
SAIC-Frederick, Inc.



(a) (b) (c)

Figure 1. Automatic visualization on confocal microscopy cell images[4]. (a) Slice view of the original dataset. (b) The dataset rendered using transfer functions generated automatically using the proposed approach. (c) The dataset rendered in 3D Slicer (http://www.slicer.org) with transfer functions generated manually by a visualization specialist using the automatic visualization as a reference. Fine tuning of the transfer functions by hand were required to generate comparable rendering results. Furthermore, the automatic rendering result still exhibits better detail of sub-cellular structures and boundaries.

ABSTRACT

This paper presents a new framework for simple, interactive volume exploration of biological datasets. We accomplish this by automatically creating dataset-specific transfer functions and utilizing them during direct volume rendering. The proposed method employs a K-Means++ clustering algorithm to classify a two-dimensional histogram created from the input volume. The classification process utilizes spatial and data properties from the volume. Then using properties derived from the classified clusters, our method automatically generates color and opacity transfer functions and presents the user with a high quality initial rendering of the volume data. Our method estimates classification parameters automatically, yet users are also allowed to input or override parameters to utilize pre-existing knowledge of their input data. User input is incorporated through the simple yet intuitive interface for transfer function manipulation included in our framework. Our new interface helps users focus on feature space exploration instead of the usual effort intensive, low-level widget manipulation. We evaluated the framework using three-dimensional medical and biological images. Our preliminary results demonstrate the effectiveness of our method of automating transfer function generation for high quality initial visualization. The proposed approach effectively generates automatic transfer functions and enables users to explore and interact with their data in an intuitive way, without requiring detailed knowledge of computer graphics or rendering techniques.
Funded by NCI Contract No.HHSN261200800001E.

KEYWORDS: Direct volume rendering, transfer function automation, K-Means++.

INDEX TERMS: I.4.10 [Image Representation]: Volumetric; I.3.3 [Picture/Image Generation]: Display algorithms; I.3.6 [Methodology and Techniques]: Interaction techniques.

1 INTRODUCTION

Direct volume rendering is one of the common methods to visualize 3D (three-dimensional) volume data. A key component toward comprehensible rendering is the generation of effective transfer functions, which map data values to colors and opacities. Interactive transfer function design is a popular method to determine voxel color and opacity for direct volume rendering. Often interactive transfer function design tools provide various interactive selection widgets with a 2D (two-dimensional) histogram for user to select of regions in the histogram. The axes of the histogram represent a feature space of the input data. The interactive selection widgets provide a way to select features in the feature space. However, it is still remains a difficult task to appropriately identify and select features in multi-dimensional feature spaces. Previous work by Maciejewski et al. demonstrated features in multi-dimensional feature spaces may be not visible clearly as peaks and valleys [9]. Thus the feature space histogram

[1]e-mail: liuy5@mail.nih.gov
[2]e-mail: curtislisle@knowledgevis.com
[3]e-mail: collinja@mail.nih.gov
[4]Dataset courtesy of Dr. Karen Meaburn and Dr. Tom Misteli, Cell Biology of Genomes, NCI, NIH

IEEE Symposium on Biological Data Visualization
October 23 - 24, Providence, Rhode Island, USA
978-1-4673-0004-9/11/$26.00 ©2011 IEEE

lacks appropriate guidance for effective feature selection, requiring users to have an underlying knowledge about the input volume data. Furthermore, the large degree of freedom provided by interactive editing widgets makes transfer function editing even harder. Extensive interaction is often required in order to select the features in a multi-dimensional feature space, as they may have arbitrary shapes and sizes. Consequentially, transfer function editing can become a time consuming and often burdensome, trial-and-error process.

In this work, we propose to use the K-Means++ [1] algorithm to enhance conventional 2D histogram transfer function editing approaches. We apply K-Means++ to classify a 2D feature space histogram into arbitrarily sized and shaped clusters. Initial seeds for K-Means++ are estimated from the histogram using peak detection. We create a 1D (one-dimensional) histogram from the classified clusters such that each bin in the histogram corresponds to a single delineated region from the feature space. Should the histogram need adjustment, users are able to select multiple bins in the 1D histogram and join corresponding regions together in the feature space. Users are also able to split bins, thereby reclassifying a selected region into smaller sub-clusters. In this way, users can change the shape and the size of regions within the feature space to reveal the underlying complex structures hidden within the feature space. Finally, color and opacity transfer functions, used by the rendering process, are created from properties of the classified regions within the feature space. In this manner, users are able to easily explore the given feature space and extract and visualize their regions of interest within the input volume.

Figure 1 shows an example of our framework performing automatic visualization on a confocal microscopy image. Figure 1(b) shows the automatic rendering result using our approach. Figure 1(c) shows the rendering done using 3D Slicer by a visualization specialist. The specialist spent approximately 10 minutes in 3D Slicer to fine tune the color and opacity transfer functions and generated a comparable rendering. It is recognized that the automatic rendering framework provided guidance for the visualization specialist in the development of the rendering transfer functions. Without the guidance, our specialist would have spent even more time generating comparable transfer functions using the trial-and-error approach. This reveals a practical limitation in visualizing biological data with many current tools: visualization experts and biologists must currently "sit down together" in order to perform a meaningful visualization. While a beneficial result often comes from such an in-person exchange, it is generally impractical to do this for every dataset. Our framework addresses this limitation by automatically generating a high quality initial visualization - assisting users who are not visualization experts.

The paper is organized as follows. In Section 2, we introduce previous research works on transfer functions. In Section 3, we describe our method in detail. Experimental results on various volumes are presented in Section 4. Finally, Section 5 summarizes our work and draws conclusions.

2 Related Work

Much previous work on transfer functions is focused on enabling efficient exploration of the parameter space defined by transfer functions in order to simplify the transfer function generation process. Fang et al. applied 3D image processing tools into the volume visualization pipeline [3]. In this case, a transfer function is defined as a sequence of 3D image processing procedures. Users are able to adjust a set of qualitative and descriptive parameters to achieve their subjective visualization goals. Design Gallery interfaces automatically generate and organize transfer functions and present all possible transfer function to users simultaneously [11]. Each generated transfer function represents a different configuration of the parameter space. Users then select a transfer function, which best satisfies their goals, from these representatives.

Multi-dimensional transfer functions were initially proposed by Levoy [8]. Following this, Kindlmann [5] and Kniss [6, 7] applied the idea of the multi-dimensional transfer function in their early work. Kindlmann et al. use the first and second directional derivatives along the gradient direction for semi-automatic transfer function generation. Their work demonstrated that boundaries of structures in the input volume data can be represented by the arc-like regions on a histogram created from voxel values and the directional derivatives. Kniss et al. designed an interactive multi-dimensional transfer function editing tool. The tool adds derivative information as parameters to transfer functions and presents a set of direct manipulation widgets for specifying transfer functions. Their work shows that many interactive transfer function creation tools lack appropriate guidance for the user in making selections. Tzeng et al. created a painting user interface for users to specify the region of interest by painting on the input volume [17]. High-dimensional classification functions were then used to classify the volume for transfer function generation.

Based on principal component analysis and computer animation, Salama et al. introduced user interfaces that provide semantic information for specialized visualization problems [12]. An abstract semantic level is used to ease the specification of transfer functions that specify a complex parameter space. The proposed semantic model is based on a set of reference data. A list of the relevant structures contained in the data is generated and each structure is encoded, such that it is composed of one or more transfer function primitives. Finally, a transfer function model is created, based on these primitives, and users manually adapt the model to the input data for visualization.

Zhou et al. adopted a residue flow model based on Darcy's law in transfer function generation [20]. The residue flow model describes water absorbed by each branch in a single rooted water flow network. By applying the residue flow model on the contour tree model of the input volume, Zhou's method generates opacity transfer function for branches in the contour tree. A color transfer function is generated using color harmony concepts and topology information in the contour tree. The generated transfer function reveals the inclusive relationship between structures and maximizes color and opacity differences between them. Rotteger et al. proposed to use spatial information to classify a 2D feature space histogram into a transfer function [13]. However, color and opacity values must be manually specified for each classified cluster in their method.

Selver et al. introduced a semiautomatic method for transfer function automation [14]. A Volume Histogram Stack (VHS) is proposed as a new domain by aligning the histograms of the image slices of a CT/MR series. Then a Self-Generating Hierarchical Radial Basis Function Network (SEG-HRBFN) is used to determine the lobes of a VHS. Their method integrates spatial knowledge, local distribution of the tissues, and intensity information into a transfer function while reserving user control. Maciejewski et al. proposed a non-parametric clustering based transfer function automation method [9]. Their work uses the kernel density estimation to generate a continuous density distribution histogram. In the histogram, bin values correspond to the probabilities a voxel will be found in the feature space. With a user chosen bin schema, the density distribution is divided into different ranges. Then mapping each density distribution range to a discrete color defines the color transfer function. Users must

assign opacity value or pre-defined opacity curve to each cluster to define the opacity transfer function.

In our work, we consider spatial as well as data properties for classification on a 2D feature space histogram created from the input volume data. The classification results provide users structural information of the feature space to ease the exploration of the input data. Users can edit the classification regions on the histogram by joining selected regions or reclassifying one selected region into smaller regions. Finally color and opacity transfer functions are created from properties derived from the voxel sets corresponding to the classified regions.

3 K-Means++ Transfer Function Automation

Given an input volume data set, we generate a 2D histogram H from data scalar value and gradient magnitudes at each voxel. We then classify H into different regions using properties calculated from the voxels of each bin in H. Each region corresponds to a cluster of voxels in the input volume. For each region, we calculate properties from the voxels of the region for transfer function automation. We work in HSV (Hue, Saturation, and Value) color space for our color transfer function generation. Hue stands for the basic pure color, while saturation and value control colorfulness and brightness of the selected color respectively. Based on the concept of color harmony from Cohen et al. [2], we assign hue values to each classified region. Saturation and value are calculated using spatial information of the voxel cluster for the corresponding region. Spatial information of the voxel cluster is also used for generating the opacity transfer function. Interfaces are provided so that user can modify color and opacity for each voxel cluster for the final transfer function.

3.1 Histogram Classification

The K-Means algorithm [10] is one of the popular classification methods. It is attractive in practice due to its simplicity and good performance. However, the K-Means starts with the uniformly random selected initial seeds and it is guaranteed only to find local optimums. Using a simple and randomized seeding technique, Arthur et al. proposed the K-Means++ to augment the K-Means for optimal classification results. We have chosen the K-Means++ to classify the histogram H because it outperforms the K-Means on both accuracy and performance.

The K-Means++ algorithm works as follows: Let $X \subset \mathbb{R}^d$ be a set of n data points. Let $D(x)$ denote the shortest distance from a data point $x \in X$ to the closest seed that has already been chosen. Then the following algorithm is used for optimal seeding

1. Choose an initial seed s_1 uniformly at random from X.
2. Choose the next seed s_i, selecting $s_i = x' \in X$ with probability $\frac{D(x')^2}{\sum_{x \in X} D(x)^2}$.
3. Repeat Step 2 until we have chosen all seeds.
4. Do regular K-Means clustering using selected seeds.

In our approach, a seed in the classification is defined as a six-component vector extracted from a bin. The six components are center of mass x, y, and z coordinates, spatial variance, average scalar value, and average gradient magnitude value of all voxels assigned to the bin. Roettger et al. proposed a distance norm to measure spatial coherence for classifying a 2D histogram created from scalar value and gradient magnitudes. Considering the non-zero count histogram bins as voxel clusters, the distance norm uses two properties, center of mass and spatial variance, to classify these voxel clusters. Given a 2D histogram H plotted using scalar value and gradient magnitude. Let $p_i(T), i = 1..n$, be the normalized positions of the n voxels of the bin T in H. The center of mass of the n voxels of the bin T is defined as $c(T) = \frac{1}{n}\sum_{i=1}^{n} p_i(T)$ and the spatial variance of the n voxels of the bin T is defined as $(T) = \frac{1}{n}\sum_{i=1}^{n}\|p_i(T) - c(T)\|$. Roettger et al. then use the norm

$$N(T, T_0) = \|c(T) - c(T_0)\| + |v(T) - v(T_0)|$$

for a measurement of the spatial coherence between bin T and the reference bin T_0. The norm is used to classify histogram bins with the condition $\forall T: N(T, T_0) < r$, where r is a user defined threshold.

In their experiments, Roettger et al. showed spatial coherence is an effective metric to utilize to classify a volume. However, since the distance norm considers spatial information only, the norm will be less effective at separating features with different scalar values but strong spatial coherence. In other words, distance norm alone will have the tendency to classify unrelated features together just because they are spatially adjacent. We have extended the distance norm to include scalar values, gradient magnitude, as well as spatial coherence information. Taking these additional terms into account, we define a new feature similarity measurement equation as follows:

$$S(T, T_0) = e_0\|c(T) - c(T_0)\| + e_1|v(T) - v(T_0)| + e_2|I(T) - I(T_0)| + e_3|G(T) - G(T_0)|$$

where $S(T, T_0)$ is the feature similarity between the bin T and the reference bin T_0, $I(T)$ is the average intensity value of the bin T, $G(T)$ is the average gradient magnitude of the bin T, e_i is the weighting coefficient, and $\sum e_i = 1$. By default we use a coefficient set $(0.25, 0.25, 0.25, 0.25)$ for equal weighting on the four components. All four components are normalized to range $[0, 1]$. We have created an auxiliary data structure to store the voxel cluster for each non-zero bin in H and we use bins to refer to the voxel clusters for each non-zero bin in H in the rest of the paper.

Both the K-Means and the K-Means++ require an initial number of clusters to start with. While it is both straightforward and tempting to use knowledge of the input dataset to drive the initial number of clusters, we did not want to depend on *a priori* knowledge. Instead, we develop the number of initial clusters algorithmically from the voxel data. To accomplish this, we use a simple histogram peak detection method working on the one-dimensional histogram developed from the input volume's scalar voxel values to estimate the parameter automatically. To formulate the number of seeds, we first apply a Gaussian blur to smooth the 1D histogram, and then slide a four neighbor wide window over the smoothed histogram to find local peaks. Whenever the current bin value is higher than the four neighboring bins, we increase a peak counter by one. Finally, we filter out small peaks to suppress noise. We define a lower peak value threshold τ, and a peak is considered as small if its value is lower than the threshold. By default the threshold is the lower 5% of the all peak value range. We note here that precise estimation of the total number of histogram peaks is not our primary focus. We are using major histogram peaks as a rough estimation for the number of major features on which to start the classification process. Interfaces are provided such that users may override the automatically estimated parameters.

3.2 Optimal Histogram Bin Size

Earlier work has focused on selecting optimal histogram bin size for accurate representation of the underlying distribution. If too small a bin size is chosen, then the voxel count in each bin is too low and the bin suffers significant statistical fluctuation across its member voxels. If too large a bin size is chosen, then the histogram has not enough bins to faithfully represent the shape of

the underlying distribution. In our method, the 2D histogram H classifies the input volume as the first step. Too large a histogram bin size may result in too low a resolution to separate voxels into enough different features. On the other hand, too small a histogram bin size may result in insufficient number of voxels in each bin for effective classification, resulting in too many feature classes being identified initially.

Shimazaki et al. [15] have proposed a MISE (Mean Integrated Square Error) based method for optimal histogram bin size estimation using a cost function

$$C(\Delta) = \frac{2k - v}{\Delta^2}$$

where Δ is the bin width, $k \equiv \left(\sum_{i=1}^{N} k_i\right)/N$ and $v \equiv \left(\sum_{i=1}^{N}(k_i - k)^2\right)/N$ are, respectively, the mean and variance of the number of events, and k_i is the number of events of the ith bin. The cost function is repeatedly applied to the histogram, while reducing bin width from the full range (meaning only one bin per histogram), to a pre-set lower boundary in the number of required bins. The bin width with the lowest score of the cost function is then picked as the optimal histogram bin size. Their method was originally applied to a time histogram but it can also be used on a probability or density histogram. We perform the algorithm on the two axes of the 2D histogram H to begin with nearly optimal histogram bin sizes. In Table 1, we have listed reference running times for histogram size optimization on several sample data sets.

3.3 Color and Opacity Transfer Functions

In direct volume rendering, colors are often determined via personal preferences using an ad hoc or randomized selection process. On the other hand, good visual aesthetics are important in visualization as it reduces stress and improves insight by making the data exploration task a more enjoyable one. Our framework takes color theory into account as colors are chosen from features in our automatically built transfer functions.

Color harmony is a popular design aspect of visual aesthetics. Harmonic colors are sets of colors that are aesthetically pleasing to the human visual perception system [2]. Itten introduced a color wheel in which color harmony is described across color hues [4]. In Itten's theory, color harmony is based on the relative positions of the hues on the color wheel instead of specific colors. While currently there is no formulation for the definition of a harmonic color set, consensus exists on when a color set is harmonic. For example, based on Itten's color wheel, Matsuda introduced a set of 80 harmonic color schemes, defined by combining 8 types of hue and 10 types of tone distributions [18]. Figure 2 illustrates the eight harmonic hue types defined over the hue channel of the HSV color wheel. These types are named for the letter matching the shape of the regions containing harmonized colors in each type. Each type can be rotated arbitrarily for different harmonic color sets. Cohen at al. uses these hue types later for a color harmonization technique to adjust image colors. Recently, researchers have applied color harmony in color transfer function generation in direct volume rendering. Wang et al. provided an interface for users to choose harmonic colors in volume visualization [19]. Their work includes a lightness-preserving color harmonization mechanism to convert non-harmonic colors to harmonic ones, but keeping the original contrast relationships. It is shown in their work that harmonic colors can be used for the highlighting of important features in volume datasets. Zhou et al. adopted a residue flow model based on Darcy's law in transfer function generation. In their work, color transfer function is generated on CIELAB and HSV color space using color harmony and topology information in a contour tree created from the input volume dataset. For each branch in the contour tree, a *hue*, *lightness*, and *vividness* (h, L, V) color triple is selected. Then color hue for each branch is selected using the harmonic hue template. Lightness and vividness are selected on CIELAB color space using topological attributes derived from the contour tree. Finally these (h, L, V) triples are converted to their (h, s, v) equivalents for visualization.

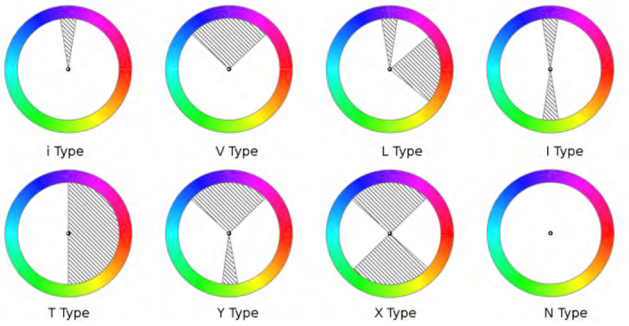


Figure 2. Eight harmonic color hue types [2].

We work in the HSV color space for the color transfer function generation in Quick2Insight. Similar with Zhou et al.'s approach, we use the harmonic hue templates for hue selection for classified clusters. Given a selected hue template, we equally divide the hue range for all classified clusters for hue selection. For volumes with a large number of estimated features, users may choose a hue type with a larger hue range to increase the difference between clusters. In addition, we use spatial information derived from the classified voxel clusters to define the saturation and value components of the color triple defined in defined in HSV color space. In our method, saturation for each voxel is defined as

$$S_i = \frac{1}{1 + v_i}$$

where v_i is the spatial variance of the ith cluster. Value for each cluster is defined as

$$V_i = \frac{1}{1 + dist_i}$$

where $dist_i$ is the distance between the center of mass of the ith cluster and the center of volume. Combining the two definitions, we can use more vivid colors to highlight clusters with smaller spatial variance and shorter distance to the center of volume. The color tuple (h, S, V) for each cluster is later converted to RGB values in the volume rendering step.

We define opacity of each cluster as

$$o_i = 1 - \frac{v_i}{v_{max}}$$

where v_{max} is the maximum spatial variance among all clusters. The effect is to assign higher opacity to clusters with smaller spatial variance as clusters with larger spatial variance are likely to block viewing of clusters with smaller spatial variance. As high gradients often represent boundaries between materials, the cluster opacity is modulated with a boundary enhancement [16] method to bring out the boundaries and more effectively illustrate embedded features. We increase the opacity proportional to the voxel gradient magnitude such that

$$o_e = \alpha_0 o_o \cdot \left|\vec{\nabla}(P)\right|^{\alpha_1}$$

where o_e is the enhanced opacity, o_o is the original sample opacity, $\vec{\nabla}(P)$ is the gradient at the sample point P, α_0 is the global opacity factor, and $\alpha_1$is the boundary exponent for the

sensitivity of the rendering due to gradient magnitude at the feature boundary.

After all transfer function parameters are calculated, to enhance rendering speed, a 2D lookup texture is created based on the 2D histogram H using a one-to-one mapping such that one pixel in the texture corresponds to one bin in H. Pixels in the texture receive color and opacity values according to the color and opacity transfer functions. The 2D lookup texture is then sent to the GPU as a lookup table for volume rendering.

3.4 Transfer Function Manipulation

We have implemented a simple yet intuitive interface for transfer function manipulation. The interface consists primarily of a 1D (one-dimensional) bar histogram with overlayed interaction widgets. The 1D histogram is initialized with a bar representing each cluster from the 2D histogram H. Since each cluster is composed of related voxels, we can also refer to each cluster as a feature class. The 1D bar histogram is rendered with the height of each bar determined from the voxel counts of the classified clusters in logarithm scale. Users can highlight one feature class by clicking the corresponding bar on the 1D histogram. Using the handle provided on each bar, users can refine the opacity values assigned to each class. The user is optionally provided with the ability to display the color-coded 2D histogram, on which each cluster is rendered with its assigned color.

The initial volume rendering is tuned to be specific to the input dataset. However, users may still want to further explore or refine the rendered result by editing the classification results. In this case, we have defined two different operations, *merge* and *split*, for transfer function manipulation. The *merge* operation is used to combine multiple feature classes into one larger class and is useful to combine over-classified feature classes. The *split* operation further divides a feature class to smaller sub-classes. Using the merge and split operations, users can expand or shrink arbitrarily shaped and sized regions on the 2D histogram, which is difficult using conventional 2D histogram manipulation widgets. As users perform the merge or split operations to modify regions in the transfer function, color and opacity values are regenerated for all clusters, and these changes are reflected automatically in the interactive volume rendering window.

In summary, our system performs an initial classification of input volume voxels into multiple feature classes, and the supplied user interface allows human guided manipulation of the color and opacity transfer functions generated based on the feature classification. Figure 3 shows an example of the 1D bar histogram and the corresponding color-coded 2D histogram. We developed the 1D histogram interface because of regions in the 2D histogram are generally hard to define directly using conventional 2D transfer function manipulation widgets.

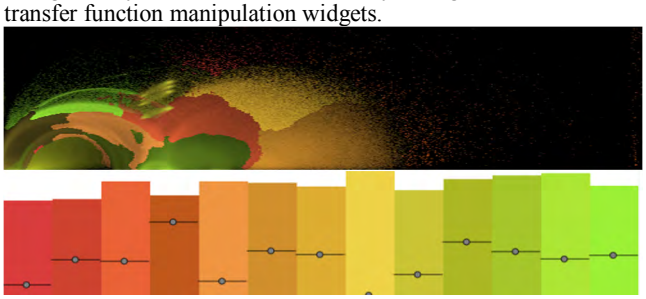


Figure 3. Top: 2D color-coded histogram. Bottom: 1D bar histogram for the clusters.

4 Experimental Results and Discussion

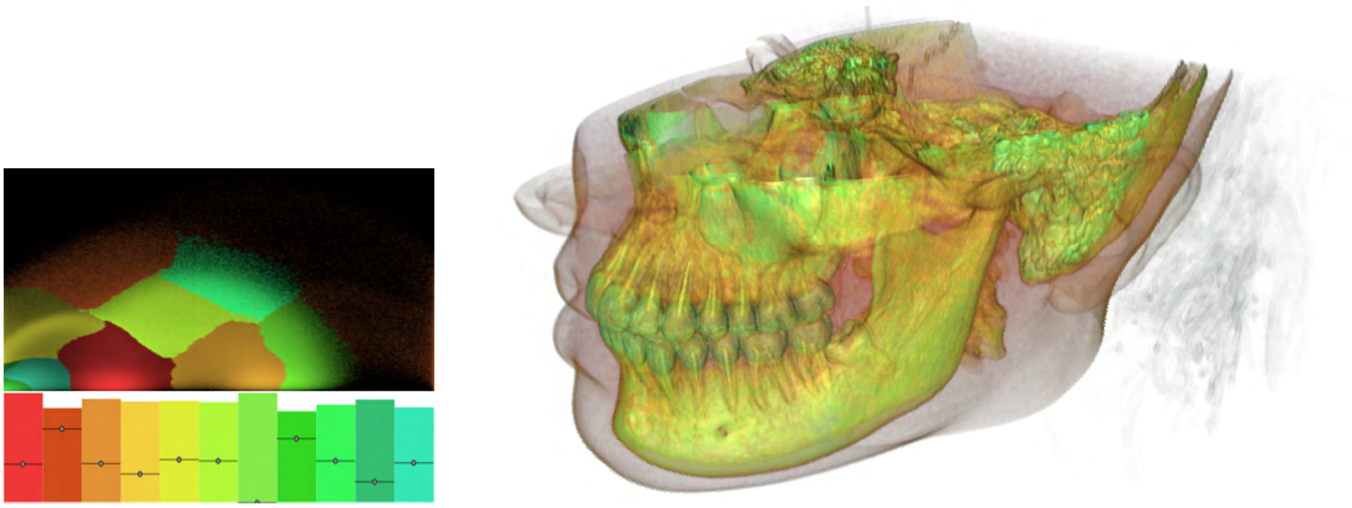
We now present experimental results on various data sets to illustrate the effectiveness and utility of the proposed approach in volume visualization. Our system was run on a Dell Precision 6400 mobile workstation (Intel Core2 Duo T9800 CPU) with an NVIDIA Quadro FX 3700m video card. The operating system was Ubuntu Linux version 10.04.

Figure 4 shows the transfer function automation results using the proposed approach on the Osirix Incisix $512 \times 512 \times 166$ CT dental scan data set. The user is directed to http://pubimage.hcuge.ch:8080/ for more information on this and similar datasets. Figure 4(a) shows the generated transfer functions and (b) shows the resulting volume visualization of the data set using the auto-generated transfer functions. The figure is rendered using the harmonic color hue T type with the starting hue value at 0. On the 1D bar graph, the horizontal line segment on each cluster represents the opacity value assigned to that cluster. In our approach, we assign one distinct opacity value for each cluster, which departs from earlier work where users must manually define curves or ramps for each cluster's opacity transfer function. Our result shows that the proposed method is effective in revealing nested inner features, without totally removing the outer layers, resulting in a high quality initial visualization of the input data set.

Next, we tested the proposed approach on the Osirix Manix human head CT scan dataset. We performed merge and split operations on the classification results to separate different features. Figure 5(a) shows the final 2D histogram. Given the cluster shapes, it is obvious that regions on the histogram would have been difficult to define using conventional transfer function widgets. Figure 5(b) shows a rendering of skull and vessels in the dataset. Figure 5(c) demonstrates the proposed method effectively separates brain, skull and vessels into separate feature classes. The color transfer function is generated based on the harmony color hue V type with starting hue at 0.

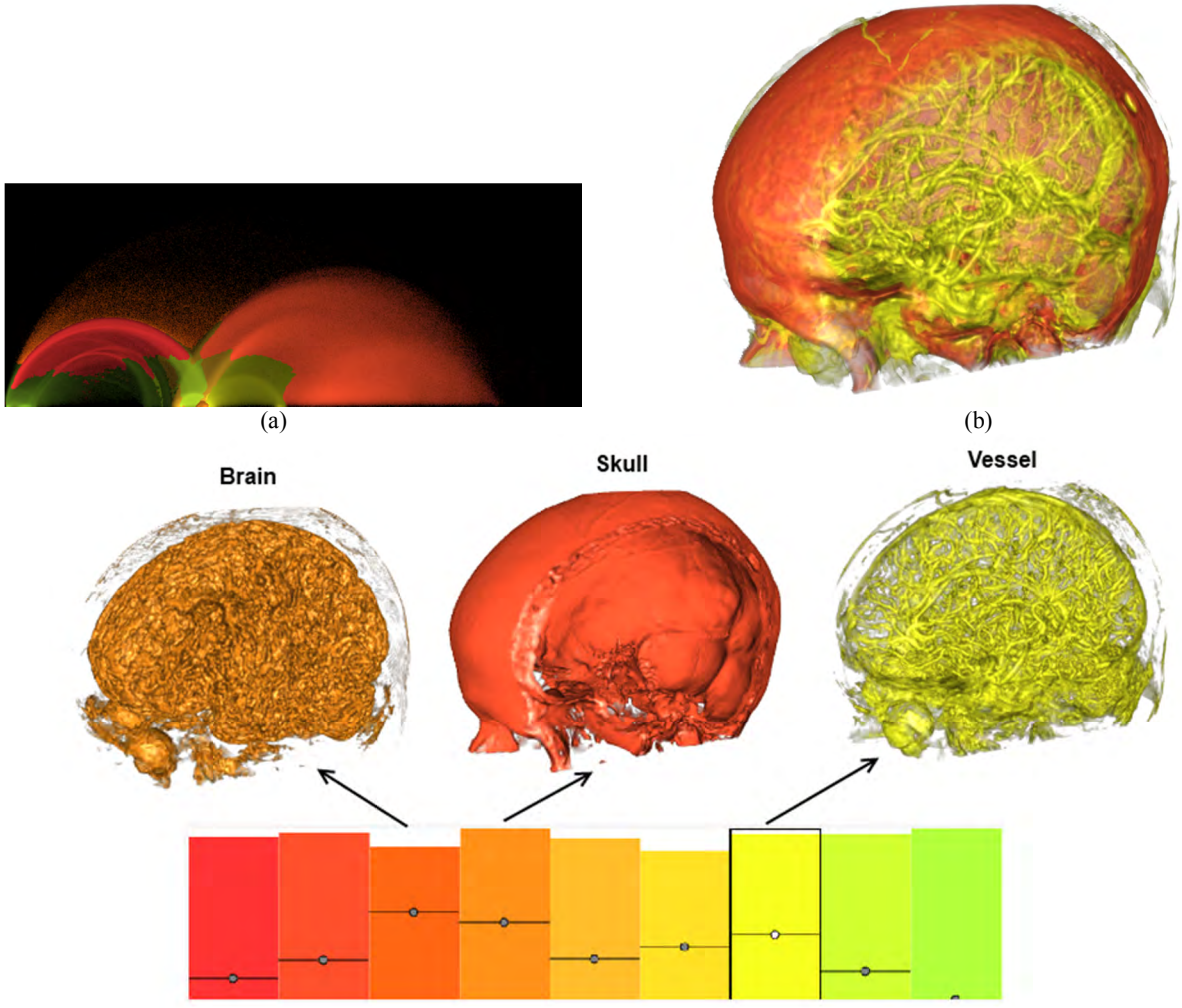
Figure 6 shows experimental results on further volume datasets. Figure 6(a) shows the automatic volume visualization on the Osirix Knee MR data set. The color transfer function is generated using the harmonic color hue V type with starting hue value at 0. Figure 6(b) shows the automatic volume visualization on the Osirix Colonix CT data set. The color transfer function is generated using the harmonic color hue V type with starting hue value at 0. Figure 6(c) shows the results on the Osirix Anonymize cat MR scan data set. The color transfer function is generated using the harmonic color hue V type with the starting hue value at 0.92. In Figure 6(d), we show the automatic rendering of the well-known bonsai data set. To acquire this dataset, the reader is directed to http://volvis.org. The color transfer function is generated using the harmonic color hue N type with the starting hue value at 0.49. The rendering result of the bonsai data set shows that the proposed method is also effective on non-medical images.

Our experiments show that the proposed approach is effective in generating automatic transfer functions. Compared with conventional methods, our approach has the following advantages: 1) It is capable of producing effective initial visualizations on volume datasets by automatically revealing different layers and features without user involvement; 2) It does not require users to have deep understanding of underlying voxel distributions in their volume datasets; 3) When a user desires to tune a transfer function, our method only requires the execution of simple merge and split interactions instead of the time-consuming editing controls of conventional transfer function widgets. In summary, our method greatly improves the transfer function generation process by providing a simple yet powerful transfer function automation framework. Users who are not visualization experts could significantly benefit from this approach.



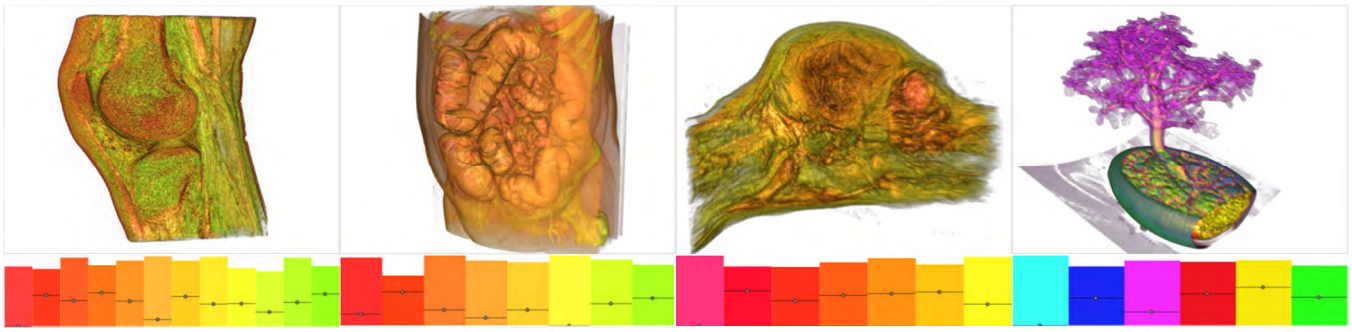
(a) (b)

Figure 4. Automatic volume visualization on the Osirix Incisix CT dental scan data set. (a) Transfer functions generated using the proposed method. (b) Direct volume rendering of the data set using the generated transfer functions.



(a) (b) (c)

Figure 5. Volume visualization on the Osirix Manix human head dataset. (a) 2D transfer function after performing merge and split operations. (b) Volume rendering of skull and vessels. (c) Features separated by the proposed method.



(a) (b) (c) (d)

Figure 6. Automatic visualization results on various data sets. (a) Knee MR. (b) Colon CT. (c) Cat MR. (d) Bonsai CT.

In Table 1, we provide reference running times for classification on different sample datasets using our framework. Times listed in the table are in seconds. In the table, H_t is the histogram size optimization time; C_t is the running time for the first time classification using automatically estimated parameters after loading a volume dataset. We have also utilized GPU acceleration by porting our C++ code to OpenCL. Without detailed optimization, we achieved 30~50% performance gain by using OpenCL acceleration on our mobile GPU. We expect to have even higher performance gain on later, more powerful GPUs. In Table 1, the last column, C_t (OpenCL), shows the running time for the initial classification using OpenCL acceleration.

Table 1: Reference running times on sample datasets

Dataset	Size	H_t	C_t	C_t (OpenCL)
Incisix	512x512x166	2.16	49.71	33.09
Knee	400x400x250	1.79	46.98	26.68
Bonsai	256x256x256	0.75	18.36	10.90
Colonix	256x256x445	1.33	109.62	60.17

5 Conclusion and Future Work

In this work, we have shown that K-Means++ based feature space clustering can provide the basis for automatic transfer function generation. Using spatial and data properties, our proposed method effectively separates volumetric structures in the feature space. Transfer functions are automatically generated based on color harmony and the spatial information of the resulting voxel clusters. Finally, our interface provides users an intuitive approach to further manipulate transfer functions. Users can perform the *merge* and *split* operation on the 1D bar histogram to modify the shape and size of regions in the feature space and improve the illustration of volumetric structures. Preliminary experiments demonstrate that our method is effective for generating high quality initial visualizations on various volume datasets.

In summary, the proposed approach allows efficient transfer function automation as well as intuitive data exploration through K-Means++ clustering based on both voxel spatial and data properties. In the future, more choices of classification properties will be added. We also plan to explore different 2D histograms, such as local texture descriptors, and explore feature-size based histograms to further improve our clustering capability.

Acknowledgment

This project has been funded in whole or in part with federal funds from the National Cancer Institute, National Institutes of Health, under Contract No. HHSN261200800001E. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.

References

[1] David Arthur and Sergei Vassilvitskii, k-means++: The Advantages of Careful Seeding, *In Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA'07, 2007, Society for Industrial and Applied Mathematics, pages 1027-1035

[2] Daniel Cohen-Or, Olga Sorkine, Ran Gal, Tommer Leyvand, and Ying-Qing Xu, Color Harmonization, *ACM Transactions on Graphics*, vol. 25, no. 3, 2006, pages 624-630

[3] Shiaofen Fang, Tom Biddlecome, and Mihran Tuceryan, Image-Based Transfer Function Design for Data Exploration in Volume Visualization, *In Proceedings of the conference on Visualization '98*

[4] Johannes Itten, The Art of Color, New York: Van Nostrand Reinhold Company, 1960

[5] Gordon Kindlmann and James W. Durkin, Semi-Automatic Generation of Transfer Functions for Direct Volume Rendering, *In IEEE Symposium on Volume Visualization*, 1998, pages 79-86

[6] Joe Kniss, Gordon Kindlmann, and Charles Hansen, Interactive Volume Rendering Using Multi-Dimensional Transfer Functions and Direct Manipulation Widgets, *In Proceedings of the conference on Visualization '01*

[7] Joe Kniss, Gordon Kindlmann, and Charles Hansen, Multidimensional Transfer Functions for Interactive Volume Rendering, *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 3, 2002

[8] Marc Levoy, Display Surfaces from Volume Data, *IEEE Computer Graphics and Applications*, 1988

[9] Ross Maciejewski, Insoo Woo, Wei Chen, and David S. Ebert, Structuring Feature Space: A Non-Parametric Method for Volumetric Transfer Function Generation, *IEEE Transactions on Visualization and Computer Graphics*, 2009, vol. 15, no. 6, pages 1473-1480

[10] J. MacQueen, Some methods for classification and analysis of multivariate observations, *In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, pages 281-297

[11] Joe William Marks, and Brad Andalman, and Paul A Beardsley, and William T Freeman, and Sarah F Frisken Gibson, and Jessica Kate Hodgins, and Tae Lk Kang, and Brian Vincent Mirtich, and Hanspeter Pfister, and Wheeler Ruml, and Kathy Ryall, and Joshua E Seims, and Stuart M. Shieber, 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*, 1997, pages 389-400

[12] Christof Rezk Salama, Maik Keller, and Peter Kohlmann, High-Level User Interfaces for Transfer Function Design with Semantics, *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, 2006, pages 1021-1028

[13] Stefan Roettger, Michael Bauer, and Marc Stamminger, Spatialized Transfer Functions, *EUROGRAPHICS – IEEE VGTC Symposium on Visualization*, 2005

[14] M. Alper Selver and Cüneyt Güzeliş, Semiautomatic Transfer Function Initialization for Abdominal Visualization Using Self-Generating Hierarchical Radial Basis Function Networks, *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 3, 2009, pages 395-409

[15] Hideaki Shimazaki and Shigeru Shinomoto, A Method for Selecting the Bin Size of a Time Histogram, *Journal of Neural Computation*, vol. 19, no. 6, 2007

[16] N. Svakhine and D. S. Ebert. Interactive volume illustration and feature halos. *In Proceedings IEEE-VGTC Pacific Visualization Symposium*, October 2003.

[17] Fan-Yin Tzeng, Eric B. Lum, and Kwan-Liu Ma, A Novel Interface for Higher-Dimensional Classification of Volume Data, *In Proceedings of the 14th IEEE Visualization* 2003

[18] Masataka Tokumaru, Noriaki Muranaka, and Shigeru Imanishi, Color Design Support System Considering Color Harmony, *In Proceedings of the 2002 IEEE International Conference on Fuzzy Systems*, pages 378-383

[19] Lujin Wang and Klaus Mueller, Harmonic Colormaps for Volume Visualization, *IEEE/EG Symposium on Volume and Point-Based Graphics*, 2008

[20] Jianlong Zhou and Masahiro Takatsuka, Automatic Transfer Function Generation Using Contour Tree Controlled Reside Flow Model and Color Harmonics, *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, 2009