Exploration of Volumetric Datasets through Interaction in Transfer Function Space

Francisco de M. Pinto and Carla M. D. S. Freitas
Instituto de Informática – Universidade Federal do Rio Grande do Sul
fmpinto@inf.ufrgs.br, carla@inf.ufrgs.br

Abstract

Direct volume rendering techniques allow visualization of volume data without extracting intermediate geometry. The mapping from voxel attributes to optical properties is performed by transfer functions which, consequently, play a crucial role in building informative images from the data. One-dimensional transfer functions, which are based only on a scalar value per voxel, often do not provide proper visualizations. On the other hand, multi-dimensional transfer functions can perform more sophisticated data classification, based on vectorial voxel signatures. The transfer function design is a non-trivial and unintuitive task, especially in the multi-dimensional case, and its controlled modification allows the user to selectively enhance different structures in the volume. In this paper we discuss the interactive approach of a transfer function design technique that allows the user to explore volumetric datasets by interacting with a derived space as well as with voxels in the volume space.

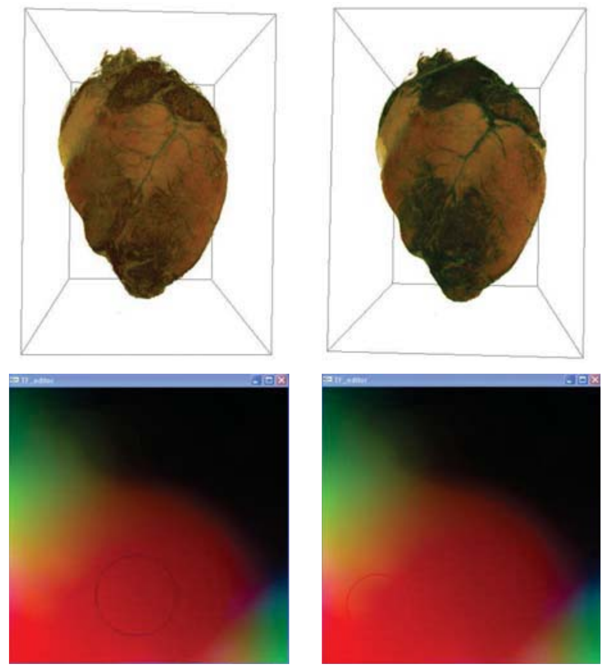

Figure 1. The sheep heart dataset viewed with two TFs, represented as circle in the color map. The circle represents a Gaussian opacity function with peak – maximum opacity – at the center. The color maps are fixed, with red, green and blue representing scalar value, gradient magnitude and directional second derivative, respectively.

1. Introduction

In direct volume rendering (DVR), transfer functions (TFs) are used for emphasizing regions of interest inside the volumes. The most common type of transfer function is the one-dimensional TF, which assigns optical properties (usually color and opacity) to voxels based only on their scalar value. Notwithstanding, one-dimensional TFs have a very limited classification power because they can not make distinction between volume regions defined by scalar values within the same range. On the other hand, multi-dimensional transfer functions can perform better classification because they can take into account not only the scalar value of a voxel [8], but also other attributes like gradient magnitude, directional second derivative, curvature [7] and statistical measures [23].

Designing an appropriate transfer function, even a one-dimensional TF, is a difficult task and much attention has been given to this issue in the literature after Pfister et al.[15]. As the domain dimension increases, the visualization of the transfer function becomes more difficult, and so the interaction with it. Controlled modifications of color and opacity transfer functions allow enhancing parts of the datasets, thus hiding features and revealing others that the user is trying to isolate. This way, the representation of the

This paper is based on the M.Sc. dissertation named "Especificação de Funções de Transferência Unidimensionais e Multidimensionais para Visualização Volumétrica Direta" by the first author.

transfer functions and the ways the user can interact with them play an important role in the exploration of inner structures in volumetric datasets.

In this paper we present an interactive technique for the design of multi-dimensional transfer functions. With our approach the user can interact with a simplified representation of the TFs obtaining different informative visualizations in a very easy and fast way (Figure 1). We discuss the resulting method compared to a previous interface for specifying one-dimensional transfer functions.

The paper is organized as follows. Next sections present the closest related works, including a brief review of our previous work. Section 3 describes our approach in detail, while main implementation aspects are discussed in Section 4. Section 5 discusses the results obtained with our method, and finally, in Section 6 we draw some conclusions and point out future work.

2. Related work

2.1. One-Dimensional transfer function specification

Traditional approaches for TF specification rely on the user's effort in adjusting control points of a graphic plot mapping voxel values to opacity level and/or color tone. The control points are then interpolated in order to build the TF. But, with no clues or prior knowledge about the data, this is a "blind process". Some data-driven approaches provide to the user higher-level information [2][14] that helps in obtaining insight about the data distribution as well as supports the manual TF design. Other methods build abstractions of the TF specification process - the transfer functions can be hidden from the user [24] or a simplified space can be presented [6].

Kindlmann and Durkin [6] proposed a derived space for specification of opacity transfer functions in which the user specifies opacities for voxels as a function of the distance between the voxel and the nearest boundary. Informative histograms are built relating voxel values with the first and second derivative values associated to each voxel in the volume. From these histograms, the mean first and second derivative values associated to each voxel value are used to estimate the distance to the nearest border. Since the boundaries must be emphasized, voxel values with small estimated distances should receive larger opacity values. Prauchner et al. [18] used Kindlmann's method to classify the voxel values by the estimated distance to the nearest border. A set of voxel values with the smallest distances is elected and random subsets are then built. The values of each subset are used as control points for the TF specification. Each of these points receives a random color and a random opacity value different from zero. The transfer functions are obtained by interpolating the control points. Consequently, each subset of the "best" voxel values derives a transfer function to be presented in a gallery of thumbnails, similar to the Design Galleries method [13]. This is the first level of the two-level interaction interface proposed by Prauchner et al. In the second level, the user can visualize a selected thumbnail in better resolution and refine its TF by adjusting the control points manually. The thumbnails can be randomly re-generated any time at interactive rates.

Following this two-level interaction interface approach, we [16] also adopted the gallery to present several thumbnails generated initially through a boundary emphasis technique, following Kindlmann and Durkin [6]. At this level, the user can generate new thumbnails (TFs) by either reapplying the boundary emphasis or selecting thumbnails as parents of a next generation of TFs, which are generated using an stochastic approach by He et al.[4]. The user can also select a specific thumbnail, and go to the second-level of interaction. At this level, looking at the rendered volume in high resolution, the user can modify the TF manually either by interacting with the TF graphic plot (Figure 2) or by picking voxels – to be emphasized by increasing the opacity for its scalar value – from a cutting plane (Figure 3). This work was published in SIBGRAPI 2006 and its extended version is to appear in a Computer and Graphics special issue.

2.2. Multi-dimensional transfer function specification

The design of multi-dimensional transfer functions brings challenges regarding both the visualization of the TF as well as the exploration of the TF domain.

It is possible to explicitly define a multi-dimensional transfer function by interacting in its domain with proper tools. Kniss et al. [8] proposed a volume rendering environment containing a set of direct manipulation widgets for volume inspection, visualization of data distribution and design of three-dimensional transfer functions, using dual domain interaction.

However, the difficulty of exploring the transfer function domain increases with its dimensionality; therefore some approaches for transfer function design provide interfaces based on interaction in a simplified space. Region growing techniques were used by Huang and Ma [5] to segment volume data from seed points specified by the user; voxel signatures of the segmented region were used to automatically design a transfer function.

Tzeng and Ma [25] clusterized voxel's signatures by similarity allowing the user to specify the desired classification by successively splitting and merging the clusters. The user sees the results by associating visual properties to each material class. The same authors [26] implemented multi-dimensional transfer functions using neural networks and support vector machines. They evaluate a classification function learned from training sets selected through a slice painting interface. The user paints the voxels of interest with a specific color, and the undesired ones with a different color. This way they implement a binary classification scheme.

Sêreda et al. [20] used hierarchical clustering to group voxels according to their LH signatures [21]. The user navigates through the hierarchy searching for the branches corresponding to regions of interest. Takanashi et al. [22] used independent component analysis (ICA) of multi-dimensional voxel signatures in order to represent them in a space where the classification is performed by moving axis aligned separation planes. Rezk-Salama et al. [19] created models of transfer functions that are carefully adjusted by specialists for several data sets of the same type in order to reveal the desired structures. Then, they applied PCA to represent the parameter set of each model by a single variable with an associated semantic. The models can be reused for new data sets by setting only that variable.

In order to have a generic design technique, we can consider designing a one-dimensional transfer function as a case of designing a multi-dimensional TF where the user selects only one variable to represent the voxel signature. In [17], we reported a method for designing nD-TFs, where voxel signatures are extracted from the volumetric dataset to be visualized, 2D or spherical self-organizing maps are built from the voxels signatures, and a dimensional reduction step results in voxels signatures being replaced by their coordinates in map space. These processes are performed off-line and perform non-linear dimensional reduction of voxel signatures. The result can be thought as a non-discrete, non-linear voxel classification scheme that group voxels by similarity and map them to a two-dimensional space: a square or a spherical surface. During rendering the user can specify color and opacity transfer functions by navigating on the map with a cursor that is the peak of a Gaussian opacity function. Next section presents this method in detail.

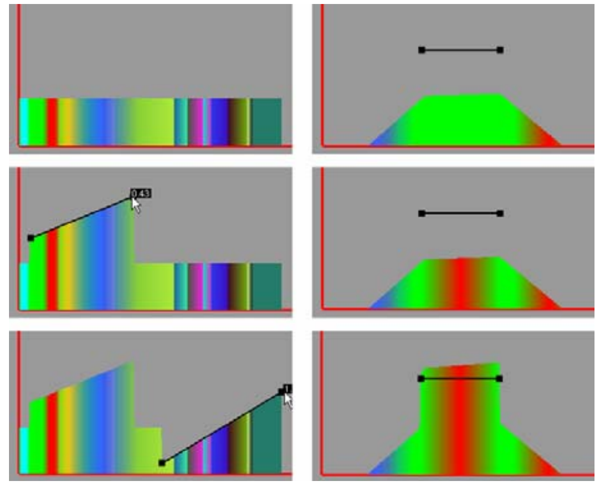

Figure 2: Manual design of one-dimensional opacity and color transfer functions.

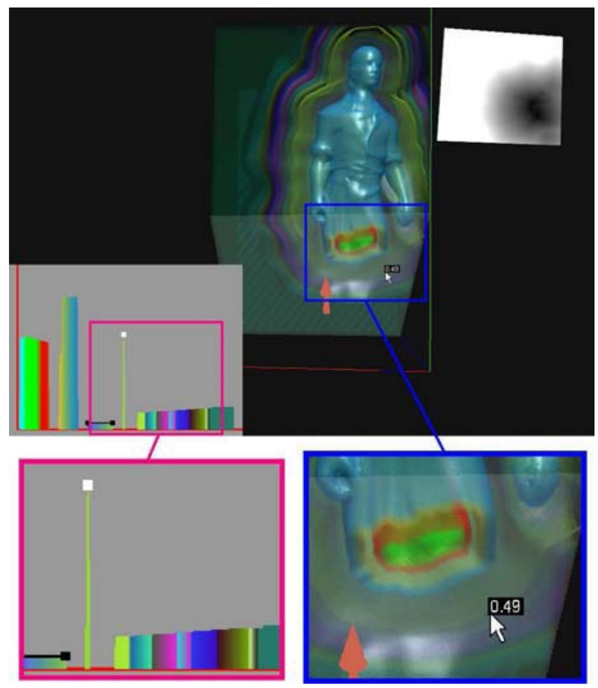

Figure 3. Top image: dataset[1] rendered using the TF shown at the left side. Bottom images: the voxel pointed by the cursor has the value marked as a white square in the TF plot. Opacity associated with this value can be interactively increased by the user.

3. Design of multi-dimensional transfer functions

Figure 4 shows the process of obtaining meaningful transfer functions for nD signatures. Our work on this subject [17] was published in EuroVis 2007.

[1] Dataset “Laçador” kindly provided by LACEM-UFRGS.

3.1. Map building process

The map building process starts with a preprocessing phase, when complex voxel's signatures (like derivative values, statistical measures, etc.) are extracted from the volume data and normalized. This way, each voxel has an nD signature (a set of scalar values represented as a vector) that can be used as a training case for the self-organizing (Kohonen) map building algorithm. It is important to mention that, depending on the source of volume data, there are many background voxels which do not carry useful information (air around scanned objects in CT/MRI volume data, for example), and would influence the map due to their high occurrence. Upon user decision, they can be partially removed from the input set of the training process by a very simple region growing technique using as seeds the voxels identified as background in the most exterior regions of the volume.

The signatures of all non-background voxels are employed as training cases presented in random order to the self-organizing map, and two types of neighborhood functions are applied. In a first stage we define the overall aspect of the map by training it using a Gaussian neighborhood function, and then, we continue the map training with a modified neighborhood function that depends not only on the topological distance, but also on the distance between the training case and the weight vector of the winner cell (refer to [17], for further details). This modified neighborhood function is designed in order to allow a voxel with a signature far from the weight vector of the corresponding winner cell (according to the distance metric) to have more influence on the map. Without this strategy, large homogeneous regions of the volume would tend to dominate the map, while important regions with fewer voxels would be badly represented (signatures far from their respective winner cells).

We use as topological distance the Euclidean distance between the integer 2D coordinates of two cells in the map grid. For spherical maps, the topological distance is the number of edges in the shortest path connecting two cells.

At the end of this process, we have a Kohonen (or spherical) map where each cell has an associated weight vector that represents a class of voxels, being the most similar weight vector for all elements in that class.

3.2. Dimensional reduction

Dimensional reduction was motivated by the need of providing a simplified space for the user to interact with the multi-dimensional transfer functions. When using Kohonen maps, two-dimensional map space coordinates in the interval from zero to one can be associated to cells according to their position in the 2D grid. Dimensional reduction can be performed by replacing each voxel signature by the coordinates of its respective winner cell. However, this would cause unnecessary discretization. To avoid this, we create two multiquadric radial basis functions (multiquadric RBFs), for x and y map space coordinates, based on the weight vectors of all cells. For spherical maps we adopt x, y and z position coordinates ranging from -1 to 1, and use three RBFs to obtain the coordinates of voxel's signatures. Thus, the RBFs supports the final step in dimensional reduction of voxel signatures by producing, through interpolation, the proper x and y (and z) map coordinates for each nD voxel signature.

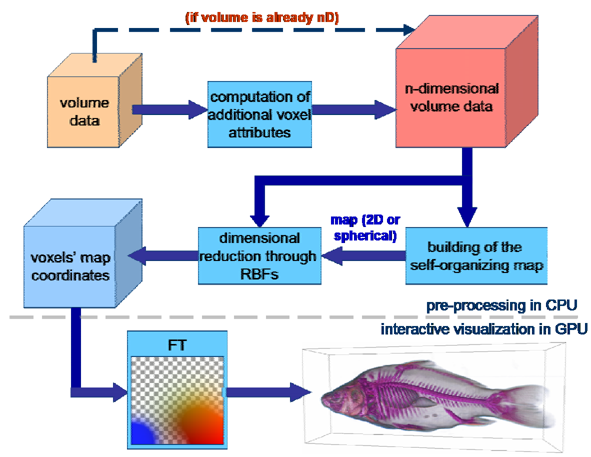

Figure 4. Distribution of processes between CPU and GPU for obtaining opacity and color TFs from nD voxel signatures.

Dimensional reduction normally implies loss and distortion of information, but volumetric data usually have properties that reduce this problem. The voxels signatures are usually not uniformly distributed in their domain (they form clusters, which are well represented in the map), and elements of the voxel signatures are often not completely independent [22]. Moreover, voxel signatures that are not present in the training set do not require space in the map.

3.3. Transfer functions specification

After the dimensional reduction step, the continuous map space defined by the RBFs becomes the TF domain. The user can interactively define the mapping from map coordinates (which represent voxel's signatures) to optical properties. We propose an interface for specification of color and opacity transfer functions that provides dual domain interaction [8] as well as visualizations of the transfer function and of the voxel signatures.

3.3.1. Interaction in TF-domain. The visualization of voxel's signatures in our interface is obtained by directly mapping up to three elements of the weight vectors of the map cells (which actually are elements of the voxel signature) to the three color channels. The user decides which element of the nD signatures should be mapped to each basic color. One element can be associated to more than one color channel and a color channel may have no elements mapped to it. This interface feature illustrates the distribution of voxel's signatures on the map and can be used to build color TFs as described below. Figure 5 (a and b) shows the distribution of voxel's signatures of the well known engine data set. The same regions (clusters of signatures) can be found in both maps.

The transfer function is represented as an RGBA image and is displayed by blending it with a checkerboard pattern. The blending function allows TFs with small opacities to be clearly visualized. Figure 5 displays TFs on a Kohonen map (c) and on a spherical map (d) generated for visualizing the engine data set. The volumetric rendering of the data set using the TF in Figure 5c is shown in Figure 7a.

Transfer functions are composed by blending several 2D Gaussian opacity TFs, each one having an associated 2D color TF. We provide three types of color transfer functions that can be associated to a Gaussian opacity function: a constant color chosen from a colorpicker; map coordinates directly mapped to color channels; and elements of weight vectors of map cells mapped to color channels (for each map coordinate the weight vectors of the near cells are interpolated and mapped to colors). At each step a new Gaussian TF is specified and then blended with the current TF, for opacity and color. The result becomes the current transfer function and the composition continues until the desired TF is reached. At start, the current TF has zero opacity and RGB colors for all the map space.

In our interface, by clicking or dragging the mouse on the map representation, the user moves a circle whose center is the peak of a Gaussian function and whose radius is its standard deviation σ. The Gaussian TF is scaled by a constant k between zero and one which is linearly mapped to the circle color, with blue being zero and red, one. The parameters σ and k can be increased or decreased using the keyboard. In order to fully explore the spherical maps, they can be rotated by dragging the mouse using the right button. The Gaussian opacity transfer function is defined in terms of the distance to the center of the circle.

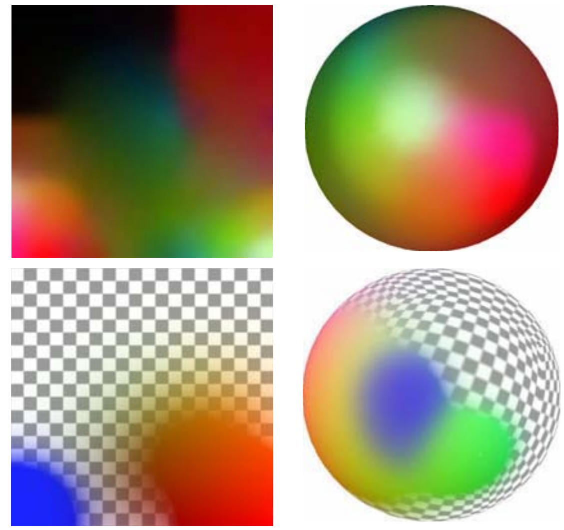

Figure 5. Maps of 3D voxel's signatures (a and b). Scalar value is mapped to red, gradient magnitude to green and second derivative in the gradient direction to blue. Transfer functions displayed on a Kohonen map (c) and on a spherical map (d).

The transfer function used for rendering is the composition of the current TF and the Gaussian function represented by the circle. This scheme provides interactive previewing of the effect of the composition while the user explores the map by moving the circle on it. When the desired effect is reached, the user can set the composition as the current TF using the space bar, and other Gaussian function can be further experimented.

Our interface (Figure 8) keeps track of all transfer functions defined during a session, and provides a tree representation of this evolution using static thumbnails of the volume rendered with the corresponding TF. This allows simple recovering of previous TFs by clicking on the thumbnails.

3.3.2. Interaction in TF-domain. At any time the user can rotate and translate the volume and place a clipping plane to better explore inner structures. The volume slice defined by the clipping plane is textured by mapping to color channels the map coordinates of the voxels sampled by the slice. This causes an interesting coloring effect that helps in inspecting the volume. The slice is blended with the rendered volume using an opacity value controlled by the user, as shown in Figure 6. When Kohonen maps are employed, the x and y map space coordinates of the voxels are mapped to red and green. When using a spherical map the x, y and z map space coordinates are mapped to RGB colors.

The user can also click on the clipping plane to set the position of the Gaussian opacity function peak to the map coordinates of the voxel pointed by the mouse cursor, emphasizing that region. By moving the mouse on the clipping plane, the user can see the position of the pointed voxel depicted as a white cross in the map graphical representation (Figure 1). This spatial domain interaction mapped to TF domain helps in understanding the relationship between both domains.

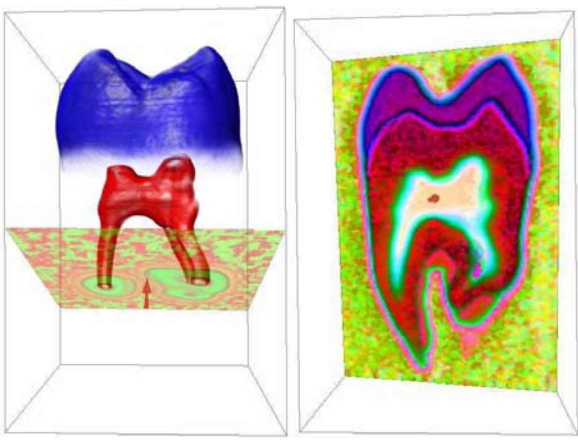

Figure 6. Visualizations of the tooth data set: a semi-transparent slice blended with the tooth image rendered using a transfer function specified in a 2D space built from a Kohonen map (a); and a fully opaque slice of the tooth colored according to voxel coordinates in a spherical map (b). The noisy regions can be clearly seen. The red arrow is the plane normal.

4. Implementation aspects

We implemented map training and dimensional reduction as offline processes, but rendering and transfer function specification demand interactive rates, which are achieved through an intensive use of the GPU (see Figure 4).

The map coordinates of the voxels are stored in a 3D RGB texture, which is sampled using view-aligned slices as proxy geometry. When using a Kohonen map, the transfer function is stored in a 2D RGBA texture which is accessed by using the R and G components (the x and y map coordinates) of the sampled 3D RGB texture. The blue component is used to identify background (zero) or non-background (one) voxels. Background voxels must receive zero opacity during rendering since they are not well represented in the map. Nevertheless, due to hardware interpolation, the blue component can assume values between zero and one. With this in mind, the opacity is actually modulated by a smoothed step function of the blue component. When using a spherical map, the TF is stored in the GPU memory as an RGBA cube map and is accessed using the RGB values of the 3D texture, taken as vectors (the value of each color channel is first converted to the interval [-1, 1]. Background voxels have null vectors and the opacity is modulated by a smoothed step function of the L2-norm of the vectors. The blending of TFs and the evaluation of Gaussian opacity functions also run in GPU.

When sampling the three-dimensional texture for rendering, interpolation must be performed. The hardware can automatically interpolate the map coordinates stored in the 3D texture and generally this produces good results. However, in our approach, it is more correct to interpolate color and opacity associated to voxels (see [3] for better understanding). In our implementation, we use the GPU to create another 3D texture, with the same size, containing the RGBA values that result from the evaluation of the transfer function for each voxel, and this texture is sampled for rendering. When the transfer function changes, this texture must be recomputed, but this strategy is fast enough for our purposes. We also calculate another 3D RGB texture to store the gradient field of the opacity. This is done in GPU by applying central differences on each voxel. The opposite vector of the gradient of the opacity is used as surface normal for shading. Since we are using complex signatures for each voxel, this scheme for normal evaluation is more accurate than to sample a 3D texture containing the precomputed normals of the scalar field. Additionally, the normals of the opacity field do not have ambiguity in their orientation (see [11]). In our implementation, we set hardware interpolation of map coordinates and pre-computed normals as default options, but the user can select color and opacity interpolation and normals computed on the fly as high-quality rendering options.

As for the RBF design, we solve the systems of equations with the Lapack library [1]. GLUT and the GLUI libraries are used for the interface, while the rendering is based on OpenGL and Cg, with the framebuffer objects extension of OpenGL used in hardware accelerated computing (Figure 8 depicts the whole interface).

5. Results and discussion

5.1. Visualization

We tested our method using well-known data sets (see Figure 7), comprising scalar and multivariate volume data. Similar results were obtained using Kohonen and spherical maps. Most of the data sets were successfully visualized using voxels signatures based on the scalar value, gradient magnitude and directional second derivative. For noisy scalar data, however, we achieved better results using statistical signatures like mean scalar value, standard deviation, and cubic root of the third-order statistical moment, taken from a small subvolume centered at the voxel under focus.

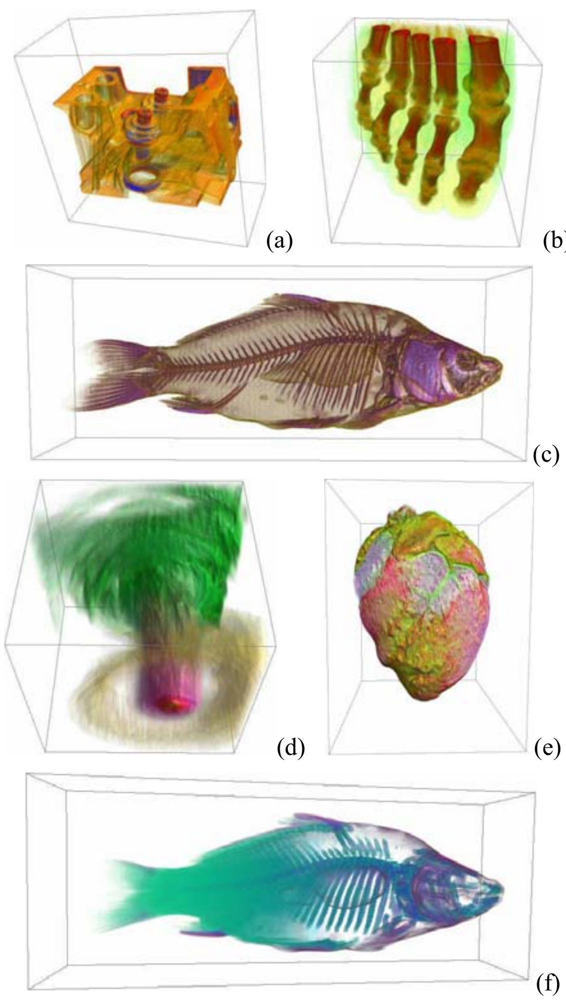

Figure 7. Visualizations obtained using Kohonen maps: (a) engine data set and (c) carp data set, both using scalar and derivative values (gradient magnitude and directional second derivative) as voxel signature; (b) foot data set, using statistical signatures (mean scalar value, standard deviation, cubic root of the third-order statistical moment); and (f) carp data set, using the normalized z coordinate of the voxels and same three statistical signatures, (z axis is horizontally represented); and Spherical maps: (d) hurricane data set at the 24th time step, using wind speed, pressure and temperature as voxel signature; and (e) sheep heart data set, using the same statistical signatures as (b).

Due to the loss and distortion of information usually caused by dimensional reduction, our method can not provide accurate quantitative information about the volume data during the transfer function specification. However, our approach is well suitable for revealing qualitative aspects like shape of structures and dissimilarity between regions.

Figure 7 shows visualizations of test data sets obtained using different sets of voxel attributes as signatures. In all these renderings, we used the automatic generated color transfer functions (see Subsection 3.3.1). For the hurricane data set we used only the colormap which assigns voxel attributes to color channels, since the attributes carry a clear physical meaning: temperature was mapped to red, pressure to green and wind speed to blue. However, the tooth data set (see Figure 6) was visualized using manually chosen colormaps and statistical signatures, achieving a very good separation of the pulp.

Since self-organizing maps group similar voxel signatures, the automatic generated color transfer functions produce very good results because they assign different colors to different regions of the map, which correspond to voxels with considerably different attribute's values.

The importance of each voxel attribute is defined by weights. We suggest associating smaller weights for higher-order voxel attributes. The visualizations presented in this paper were produced using weights of 1.3, 1.0 and 0.7 for the statistical variables formerly mentioned, respectively, and the same weights for scalar and first and second derivative values, respectively. For the hurricane data set the weights were 1.0 for wind speed and pressure, and 0.5 for temperature.

Regarding the shading using normals computed on the fly, results can be seen in Figure 7 (c, d and e). It is worth to mention that for the multivariate data sets, like the hurricane, we can not use meaningful pre-computed normals.

5.2. Interaction

Figure 1 (a and b) shows 2 views of the well-known sheep heart dataset obtained by simply moving the peak of the Gaussian TF on the map space (see Subsection 3.3.1) from the position in Figure 1(a) to that on Figure 1(b). The difference in the color of the circle also indicates a difference in the value of a specific parameter (k) from one opacity function to the other. Automatically generated color transfer functions are usually a good choice, which turns the design process less difficult. This is the case in the examples shown here.

By moving the Gaussian opacity function on the map space, the user quickly obtains an overview of the main structures in the data volume. Users can tune opacity levels and combine TFs with simple (keyboard) input. Careful tuning of parameters of the Gaussian functions and their combination allow building meaningful visualizations.

In comparison to the previous design technique, targeted at one-dimensional transfer functions, the method described herein can be thought as including boundary emphasis since the derived attributes are often intended to capture the boundaries between regions. Thus, the interactive tasks of enhancing boundaries that a user had to accomplish with the previous interface are automatically included in the exploration of the map and setting of Gaussian functions at specific positions on it.

The history tree, briefly described in Subsection 3.3.1, provides a powerful mechanism for exploring the transfer function domain, allowing not only "undo" and "redo" operations, but navigation in the whole history of TF modifications. Both design interfaces have this feature, which proved to be necessary due to the interactivity of the TF design process, which can yield to situations where the user "looses" a good transfer function.

6. Conclusions

In this paper we presented a new approach for the design of multi-dimensional transfer function that uses self-organizing maps to perform dimensional reduction of the voxel attributes. The strongest points of our technique are simplicity and flexibility. Our approach allows building multi-dimensional transfer functions through the exploration of a simplified (reduced) space where traditional interaction techniques can be employed. A simple and effective interface for transfer function design is provided, and the user can interact with the system in both spatial and TF domains.

Self-organizing maps have the ability of representing clusters of voxel's signatures in a compact way, and this helps to understand the data distribution. All relevant voxel's signatures are represented in the map and every region of the map has voxels mapped to it. Moreover, exploring two-dimensional maps is easier and faster than navigating through a class hierarchy. The proposed dimensional reduction scheme requires a preprocessing step, but it has clear advantages in relation to volume segmentation techniques because it performs a non-discrete classification which can represent uncertainty. In addition, with simple interaction, the user can change the transfer function defined in the map space, interactively obtaining new visualizations in real-time.

As future work, regarding interaction with the TF, we want to transport transfer functions designed in map space to the actual multi-dimensional space using the Gaussian multi-dimensional TFs proposed by Kniss et al. [9]. Another promising future work is the semi-automatic search for important structures in the map. This search could be aided by an interface that would provide additional information about the spatial distribution of voxels.

Wherever Times is specified, Times Roman or Times New Roman may be used. If neither is available on your word processor, please use the font closest in appearance to Times. Avoid using bit-mapped fonts if possible. True-Type 1 fonts are preferred.

7. Acknowledgements

We acknowledge the financial support from CNPq, the Brazilian Scientific Research Funding Agency.

8. References

[1] Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., Mckenney, A., Sorensen, *D. LAPACK Users' Guide, third ed*. 1999.

[2] Bajaj, C. L., Pascucci, V., Schikore, D. R., "The contour spectrum", *Proceedings of the 8th conference on Visualization '97*, Los Alamitos, CA, USA, 1997, pp. 167–174.

[3] Hadwiger, M., Berger, C., Hauser, H., "High quality two-level volume rendering of segmented data sets on consumer graphics hardware", *Proceedings of IEEE Visualization*, 2003, pp. 40–47.

[4] He, T., Hong, L., Kaufman, A., Pfister, H., "Generation of transfer functions with stochastic search techniques", *Proceedings of the 7th conference on Visualization '96*, Los Alamitos, CA, USA, 1996, pp. 227–235.

[5] Huang, R., Ma, K.-L., "RGVis: Region growing based techniques for volume visualization", *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, 2003, pp. 355–363.

[6] Kindlmann, G. Durkin, J.W., "Semi-automatic generation of transfer functions for direct volume rendering", *Proceedings of the 1998 IEEE symposium on Volume visualization*, New York, NY, USA, 1998, pp. 79–86.

[7] Kindlmann, G., Whitaker, R., Tasdizen, T., Möller, T., "Curvature-based transfer functions for direct volume rendering: Methods and applications", *Proceedings of IEEE Visualization*, 2003, pp. 513–520.

[8] Kniss, J., Kindlamnn, G., Hansen, C., “Multidimensional transfer functions for interactive volume rendering”, *IEEE Transactions on Visualization and Computer Graphics* 8, 3, 2002, pp. 270–285.
[9] Kniss J., Premoze, S., Ikits, M., Lefohn, A., Hansen, C., Praun, E., “Gaussian transfer functions for multi-field volume visualization”, *Proceedings IEEE Visualization*, 2003, pp. 497–504.
[10] Kniss, J. M., Uitert, R. V., Stephens, A., Li, G.- S., Tasdizen, T., Hansen, C., “Statistically quantitative volume visualization”, *Proceedings of IEEE Visualization*, 2005, pp. 37–44.
[11] Lum, E. B., Ma, K.-L., “Lighting transfer functions using gradient aligned sampling”, *Proceedings of IEEE Visualization*, 2004, pp. 289–296.
[12] Ma, F., Wang, W., Tsang, W. W., Tang, Z., Xia, S., Tong, X., “Probabilistic segmentation of volume data for visualization using SOM-PNN classifier”, *Proceedings of IEEE Symposium on Volume visualization*, 1998), pp. 71–78.
[13] Marks, J., Andalman, B., Beardsley, P. A., Freeman, W., Gibson, S., Hodgins, J., Kang, T., Mirtich, B., Pfister, H., Ruml, W., Ryall, K., Seims, J., Shieber, S., “Design galleries: a general approach to setting parameters for computer graphics and animation”, *Proceedings of SIGGRAPH ’97*, New York, NY, USA, 1997, pp. 389–40.
[14] Pekar, V., Wiemker, R., Hempel, D., “Fast detection of meaningful isosurfaces for volume data visualization”, *Proceedings of the conference on Visualization ’01*, Washington, DC, USA, 2001, pp. 223–230.
[15] Pfister, H., Lorensen, B., Bajaj, C., Kindlmann, G., Schroeder, W., Avila, L. S., Martin, K., Machiraju, R., Lee, J, “The transfer function bake-off”, *IEEE Computer Graphics and Applications*, 21, 3, 2001, pp. 16–22.
[16] Pinto, F.M., Freitas, C.M.D.S, “Two-level interaction transfer function design combining boundary emphasis, manual specification and evolutive generation”, *Proceedings of XIX SIBGRAPI - Brazilian Symposium on Computer Graphics and Image Processing* , IEEE Press, 2006.
[17] Pinto, F.M., Freitas, C.M.D.S, “Design of multidimensional transfer functions using dimensional reduction”, *Proceedings of Eurographics/IEEE-TVCG Symposium on Visualization*, 2007.
[18] Prauchner, J.L., Freitas, C.M.D.S., Comba. J. L. D, “Two-level interaction approach for transfer function specification”, *In Proceedings of XVIII SIBGRAPI - Brazilian Symposium on Computer Graphics and Image Processing*, IEEE Press, 2005.
[19] Rezk-Salama, C., Keller, M., Kohlmann, P., “High-level user interfaces for transfer function design with semantics”, *IEEE Transactions on Visualization and Computer Graphics*, 12, 5, 2006, pp. 1021–1028.
[20] Šereda, P., Bartroli, A. V., Gerritsen, F. A, “Automating transfer function design for volume rendering using hierarchical clustering of material boundaries”, *Proceedings of IEEE/Eurographics Symposium on Visualization*, 2006, pp. 243–250.
[21] Šereda, P., Bartroli, A. V., Serlie, I. W. O., Gerritsen, F. A, “Visualization of boundaries in volumetric data sets using LH histograms”, *IEEE Transactions on Visualization and Computer Graphics*, 12, 2, 2006, pp. 208–218.
[22] Takanashi, I., Lum, E. B., Ma, K.-L., Muraki, S, “ISpace: Interactive volume data classification techniques using independent component analysis”, *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications*, 2002), pp. 366–374.
[23] Tenginakai, S., Lee, J., Machiraju, R, “Salient isosurface detection with model-independent statistical signatures”, *Proceedings of IEEE Visualization*, 2001, pp. 231–238.
[24] Tzeng, F.-Y., Lum, E. B., Ma, K.-L, “A novel interface for higher-dimensional classification of volume data”, *Proceedings of the 14th IEEE Visualization*, Washington, DC, USA, 2003, pp. 66-73.
[25] Tzeng, F.-Y., Ma, K.-L, “A cluster-space visual interface for arbitrary dimensional classification of volume data”, *In Proceedings of the Symposium on Data Visualization*, 2004, pp. 17–24.
[26] Tzeng, F.-Y., Lum, E. B., Ma, K.-L, “An intelligent system approach to higher-dimensional classification of volume data”, *IEEE Transactions on Visualization and Computer Graphics*, 11, 3, 2005, pp. 273–284.

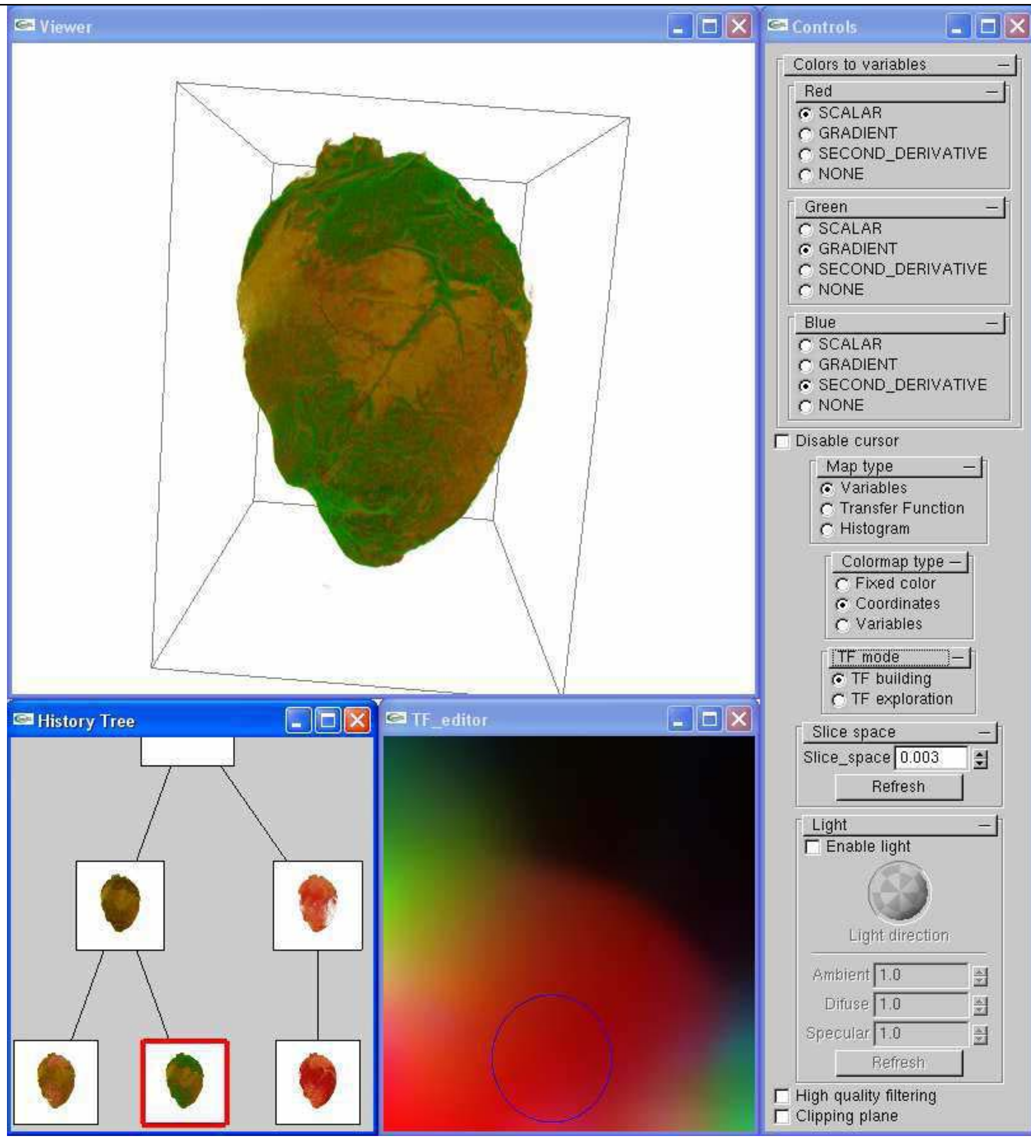

Figure 8. Visualization tool interface showing the main visualization window, the color map with a Gaussian TF marked as a circle and the history tree that maintains the TFs defined along the design process. At the right the controls for specifying the mapping between voxel attributes to colors, in the color map; the map types, and other configuration parameters.