Computers & Graphics 32 (2008) 420–429

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Volume visualization and exploration through flexible transfer function design

Francisco de M. Pinto *, Carla M.D.S. Freitas

Instituto de Informática, UFRGS, Av. Bento Gonalves, 9500 - Campus do Vale - Bloco IV, Caixa Postal 15064, CEP 91501-970, Porto Alegre, RS, Brazil

ARTICLE INFO

Article history:
Received 31 October 2007
Received in revised form
10 April 2008
Accepted 13 April 2008

Keywords:
Transfer function
Direct volume rendering
Volume data
Volume visualization

ABSTRACT

Direct volume rendering (DVR) is a well-known method for exploring volumetric data sets. Optical properties are assigned to the volume data and then a DVR algorithm produces visualizations by sampling volume elements and projecting them into the image plane. The mapping from voxel values to optical attribute values is known as transfer function (TF). Therefore, the quality of a visualization is highly dependent on the TF employed, but its specification is a non-trivial and unintuitive task. Without any help during the TF design process, the user goes through a frustrating and time-consuming trial-and-error cycle. This paper presents a useful combination of TF design techniques in an interactive workspace for volume visualization. Our strategy relies on semi-automatic TFs generation methods: boundary emphasis, stochastic evolutive search in TF space, and manual TF specification aided by dual domain interaction. A two-level user interface was also developed. In the first level, it provides multiple simultaneous interactive visualizations of the volume data using different TFs, while in the second one, a detailed visualization of a single TF and the respective rendered volume are displayed. Moreover, in the second level, the TF can be manually refined and the volume can be further inspected through geometric tools. The techniques combined in this work are complementary, allowing easy and fast TF design and data exploration.

1. Introduction

Volume rendering is widely known as a set of methods for visualization of large three-dimensional (3D) scalar or vector fields, mainly in medical and scientific data exploration. In these areas, one often deals with 3D images, such as those obtained from CT and MRI devices or numerical simulation data. Volume rendering techniques and algorithms are well described in the literature [1], and can be classified as isosurface extraction methods and direct volume rendering (DVR) methods. The former methods extract polygonal meshes representing isosurfaces in the volume and then use the traditional rendering pipeline to display the meshes (see [2], for a well-known example). On the other hand, DVR methods display volume data without extracting an intermediate geometry. DVR and its advantages were first described by Levoy [3]. Modern graphics hardware supports volume rendering at interactive rates using either of these approaches.

To obtain useful images through DVR, voxels have to be classified in order to determine which ones must be displayed. This classification is typically performed by transfer functions (TFs), which associate values of optical attributes to voxels based on their values. Opacity and color are the most common optical properties used in TFs. The degree of opacity can make a voxel more or less visible and is normally used to emphasize voxels in boundaries between different homogeneous regions of the volume [4]. Other optical properties may also be used, such as specular reflection coefficients [5], spectral reflectance [6] and light scattering coefficients [7]. More recently, the concept of style TFs was introduced by Bruckner and Gröller [8], who used TFs to define the rendering style of volume regions based on data values and eye-space normals. The information conveyed by an image built from volume data is, therefore, highly dependent on the quality of the TF. However, TF design is a non-trivial and unintuitive task, and has been referred as one of the top 10 problems in volume visualization [9].

One-dimensional (1D) TFs take into account only scalar voxel values, and are the most common TFs, although having a limited classification power. On the other hand, multi-dimensional TFs allow more freedom in voxel classification by taking as arguments vectorial values or combinations of local measures of scalar fields, such as derivative values [10,11], neighborhood, position [12], curvature [13,14] and statistical signatures [15]. Notwithstanding, the design complexity grows with the size of the TF domain [10],

* Corresponding author. Tel.: +55 51 3308 6811.
E-mail addresses: fmpinto@inf.ufrgs.br (F.M. Pinto), carla@inf.ufrgs.br (C.M.D.S. Freitas).

0097-8493/$ - see front matter © 2008 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2008.04.004

and the memory required to implement truly multi-dimensional TFs restricts their application [16]. In this work, we adopted 1D TFs due to their simplicity and low-memory requirements, since they can be implemented as small lookup tables. Furthermore, the pre-integrated volume rendering technique, proposed by Engel et al. [17], allows high-quality DVR at interactive rates using 1D TFs.

Designing TFs with no assistance leads to trial-and-error efforts. Therefore, several automatic and semi-automatic techniques for specifying TFs have been proposed [4,8,15,18–23]. They can be guided by analysis of the volumetric data (data-driven) or analysis of the generated images (image-driven) [9]. In any case, to make the process less frustrating and less time-consuming, the user must be given a rapid feedback with real-time rendering frame rates.

The main contribution of our work is a two-level interface method that combines a set of useful tools for semi-automatic 1D TF design and fast data exploration in an interactive workspace. The first level of our interface presents several thumbnails of the volume data rendered with different TFs, allowing immediate insight of the main structures in the data set. The second level shows a detailed visualization of a single TF as well as the resulting rendering. TFs can be easily generated and refined using semi-automatic boundary emphasis, stochastic evolutive search and manual design aided by dual domain interaction [11]. These three approaches are complementary and were successfully combined in this work, improving the idea of two-level interaction proposed by Prauchner et al. [24].

This paper extends our previous work [25] with an improved interface and an experimental evaluation of our approach. We implemented a history tree that keeps track of the TF evolution and allows the user to go back to a previously specified TF. The evaluation was performed as an experiment with 15 subjects who accomplished two visualization tasks with different data sets. This way we tested the usability of our methods with potential users. We also implemented a set of geometric tools for volume inspection inspired by the work of Dietrich et al. [26], but their description is beyond the scope of this article.

The paper is organized as follows. Section 2 discusses the closest related works. Section 3 describes the proposed interface and the TF design techniques provided within it. Implementation details are addressed in Section 4, while in Section 5 we present the evaluation of our tools. At last, in Section 6, we draw some conclusions and point directions for future work.

2. Related work

The TF specification problem has received much attention from researchers. Traditional approaches rely on user effort in adjusting control points of a graphic plot of the TF [27]. The control points—scalar values associated with values of optical attributes—are then interpolated in order to build the TF. However, with no clues or prior knowledge about the data, this is a "blind process". Some data-driven approaches provide users with higher-level information [18,23] that helps in obtaining insight about the data distribution, thus supporting manual TF design. Some methods hide the TF from the user through abstractions: Tzeng et al. [28] implemented multi-dimensional TFs as neural networks or support vector machines, providing users with a simple interface for selection of voxels as training sets for the learning process; He et al. [21] used genetic algorithms to perform stochastic evolutive TF design, requiring the user to only choose the best rendered images, i.e., the best TFs. Other methods offer a simplified space for TF specification. Rezk-Salama et al. [29] created models of TFs 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. Šereda et al. [30] used hierarchical clustering to group voxels according to their LH signatures [31]. The user navigates through the hierarchy searching for the branches corresponding to regions of interest. LH signatures also constitute a two-dimensional (2D) space for definition of TFs where boundaries in the data set are easily classified. Bruckner and Gröller [8] represent pre-defined shading styles as lit spheres that the user can choose to build rendering style TFs. This technique allows the user to obtain TF-guided non-photorealistic visualizations of volumes. We have also worked on multi-dimensional TFs specification [32], using self-organizing maps to perform non-linear dimensional reduction of multi-dimensional voxel values, creating an abstraction of the nD TF domain in an easy-to-use interface.

Kindlmann and Durkin [4] proposed a derived space for specification of opacity TFs in which the user assigns opacity levels to voxels as a function of the distance between the voxel and the nearest boundary. The authors assume that boundaries are smoothed by a Gaussian filtering process due to the Gaussian frequency response of 3D scanners. Informative histograms are built relating voxel scalar values to first and second derivative values. From these histograms, the mean first and second derivative values associated with each voxel value are used to estimate the distance to the nearest border. Boundaries often need to be emphasized, thus voxel values with small estimated distances are associated with large opacity values.

The design galleries method [22] is based on the generation of a wide set of TFs through a dispersion algorithm, in a pre-processing phase. Then, the obtained TFs are used to render thumbnails that are grouped by similarity and presented in an interface where the user can pick and zoom in the most appealing thumbnails to observe the resulting image. The stochastic approach for TF design proposed by He et al. [21] also uses galleries of thumbnails. The authors represent TFs as vectors of control points and the smooth interpolation of the control points produces the actual TF. The method employs genetic algorithm operators to create new TFs from the previous ones. The mutation operator changes control points, while the crossover operator produces new TFs by concatenating sub-vectors of control points from different TFs. Given a set of TFs, and their respective rendered images presented as thumbnails, the "best" ones are selected as parents, either by the user or through automatic evaluation based on objective image processing metrics. The initial TFs are randomly specified and the TF population evolves as the parents are selected, and new TFs are generated by applying the operators mentioned above to the current parents.

Prauchner et al. [24] used Kindlmann and Durkin's method [4] to classify the voxel values by the estimated distance to the nearest border. The voxel scalar values with the smallest distances are selected and random subsets of these values are then built. The values in the subsets are used as control points which receive random color and random opacity value. Each TF is obtained by interpolating the control points in one of those subsets. The generated TFs are used to render a gallery of thumbnails of the volume data, similarly to the design galleries method. This is the first level of the two-level interaction interface proposed by Prauchner et al. [24]. In the second level, the user can visualize a selected thumbnail in high resolution and refine its TF by manually adjusting the control points. The thumbnails can be re-generated at any time using new TFs.

Our method [25] improves the TF specification process by combining features from the approaches referred above into a general-purpose interactive workspace that implements image-driven and data-driven TF design through a two-level interface.