ELECTRÓNICA E TELECOMUNICAÇÕES, VOL. 5, Nº 3, JUNHO 2011

Interaction and visualization techniques for volumetric data of various scalar modalities: integration of visualization and interaction

Vítor Gonçalves

Abstract - **Over the last years, acquisition techniques have improved significantly in several areas such as oil mining industry, medicine, geophysics, weather predictions, etc. resulting in large volumetric data sets. Visualizing and exploring this kind of data introduces some questions: how to select the appropriate method for a particular case? How to represent more than 3 dimensions? How to guide the users to extract valuable information? The interaction with the final visualization is also crucial to gain insight and reach conclusions. Providing the different visualization techniques combined with appropriate methods of interaction seems to be the way to empower the final user. This paper present several visualization and interaction techniques used in volumetric data, as well as some aspects regarding the integration of visualization and interaction.**

I. INTRODUCTION

Volume visualization is one of the fastest growing areas in Scientific Visualization. 3D scanning is a common technique in many areas, using different forms of acquisition and different purposes. It is used extensively in medicine for prevention and diagnosis (PET, CT or MRI), geophysics (subsoil analysis) in oil mining, geology, archeology, etc. [1-3].

There are several areas that have the need to explore this type of data and over the years acquisition equipment have become widespread and with more capability, allowing the capture of larger volumes. Thus are more and more volumetric data and more accurately, which increases the challenge of exploring these data and extract from them valuable information.

Providing the different visualization techniques combined with appropriate methods of interaction seems to be the way to empower the final user to explore and take the most advantages of huge datasets every day larger and more difficult to understand.

Volumetric data contain three dimensions in its structure, they are equivalent to raster images (in two dimensions); however, in 2D the smallest element is the pixel (with two dimensional xy), while the volumetric (scalar) data the smallest element is the voxel (with three dimensional xyz). When the size of a voxel is equal in all dimensions (vozel size x = vozel size y = vozel size z) we say that the volume is isotropic, otherwise it is anisotropic [4].

II. VISUALIZATION TECHNIQUES

The visualization of volumetric data is not a trivial process, since in general we not only want to see only the external surfaces of the volume, but also its interior. Then, there is the problem of data representation in the output devices, as even today most devices produce 2D representation and the vast majority of the so called 3D devices are in fact 2D devices with special features that generate illusion of depth through stereoscopy, so operations are needed to represent the volumetric data (3D) in these devices. Moreover, there are other relevant issues such as exploration of the inside of the volume, image quality (color, transparency, shadow, etc.), algorithms used and performance which are directly linked to interactivity.

There are two ways to represent the volumetric data, by direct representation of the voxels (direct volume rendering), or by extracting the geometry associated to the voxels values [5].

A. Direct volume rendering (DVR)

Computer-based visualization of data is something less perfect from what occurs in reality. The quality of visualization algorithms is intrinsically related to the precision and insight achieved with the minimal use of computational resources (important to achieve sufficient performance to enable interactivity).

DVR visualization techniques consist in representing the data as a "translucent" material and this material has assigned important properties [5]: color and opacity that are in the generality of the graphics libraries allocated through color and opacity "Transfer Functions" (TFs) [6]. The TF performs the mapping of scalar values of data to the color/opacity defined in the TF [7]. For example, if a volume has values between 0 and 100, we can create a color TF defining white for value 0 and black for value 100, intermediate values will be represented in a gray level (linear gradient from white to black). The definition of appropriate values for TFs is a difficult and time consuming task (usually manual), because it is closely related to the type of data and with the goals of the visualization (which data subset(s) we need to give more emphasis?) [7].

Currently there are a number of well known techniques for implementing DVR: ray tracing, ray casting, splatting and shear warp.

These techniques can be implemented by algorithms that fall into two groups [8]:

- image-order – algorithms shoot rays from the image pixels into the volume (back-to-front), in this case is calculated only the color of each pixel;
- object-order – algorithms generally shoot rays from the volume voxels and project that data onto the image plane (front-to-back), however can also be implemented in back-to-front order. In this case each voxel contribute to the final image.

The ray tracing and ray casting are implemented through image-order and splatting and shear warp are implemented through object-order [8, 9].

A.1. Ray tracing

Ray tracing was first used for computer rendering by Arthur Appel in [10]. This technique has presented new variants over the subsequent years, as [11-13] and even different combinations of methods (illumination, shading, etc.), some to improve efficiency, others to improve the degree of realism. Ray tracing reproduces a scene using the inverse ray projection (backward from pixel to object) [6]. Therefore, only the rays that will contribute to a particular pixel are calculated, instead of the contribution of the infinity of rays which actually exist in reality [6]. Thus, considering that the end result is a visualization of the scene projected on a 2D grid of pixels, this technique will calculate only one color component for each pixel of the final image [6]. If the ray intersects objects in the scene, then the distances between the pixel and points of intersection in the scene objects are calculated [6]. The nearest point of the pixel will be the visible surface to the current pixel [6]. Then the ray is reflected in the object and in the case of transparent objects will there still a refraction ray [6]. These rays were known as secondary rays, because they result from a first ray [6]. Next the same process is repeated for the secondary rays producing a next level, resulting in a new reflected ray and possibly also a new refracted ray (in the case of transparent surfaces) [6]. The process only ends when it reaches the maximum number of defined levels or if the ray does not intersects any objects or if the ray intersects a source light [6].

This is a optimal technique to produce high quality images, but with a high computational cost, making it difficult to produce images in real time on most computer systems (off-line technique) [14]. However, this technique is highly scalable, taking advantage of current multi-core processors or multiple graphics processing units (GPUs) of graphics cards which allows real time in some of the newest systems fitted with these components [14].

However, due to these requirements in reality is used a simplified version of this technique, named "ray casting".

A.2. Ray casting

Ray casting is a special case of ray tracing [6], it differs from ray tracing since no secondary rays are projected [15]. This means that the reflections, refractions and shadows are not calculated with the same accuracy [16], however these factors can be calculated in an approximated way, for example through color mapping [17] in addition to other methods, with a much higher efficiency than ray tracing [18].

It is also notorious the amount of work that has recently emerged in adapting ray casting technique to parallel algorithms using processing through multi-core processors and especially through GPUs [19-22]. This is justified by the recent emergence of these multi-processing technologies often enabling to implement interactive volume visualizations using ray casting techniques in midrange systems.

A.3. Splatting

This technique was proposed by Lee Westover in 1989 in [23] and later refined in [24, 25] to solve the critical problem of rendering time in volumetric data, which then could take several hours with ray tracing (even for relatively small volumes, from 64^3 to 256^3 voxels). Since then, this technique has been successively improved with the same goal of obtaining interactivity through the use of less computational resources relatively to other techniques such as ray tracing [24, 25].

Generally this technique classifies each voxel and represents it in a "3D kernel", using transfer functions [24, 25] and each voxel is projected on a 2D plane through "footprint functions" [25]. Then, pre-calculated multiple perspectives through "footprint functions" are computed according to an offset, and the result is stored in a table [26]. Even though the table is discrete, the renderer can build any point of view based on this table [26].

A.4. Shear warp

Analogous to the splatting technique shear-warp is one of the volume rendering techniques that uses less computational resources, achieving this performance by reducing the image quality [27].

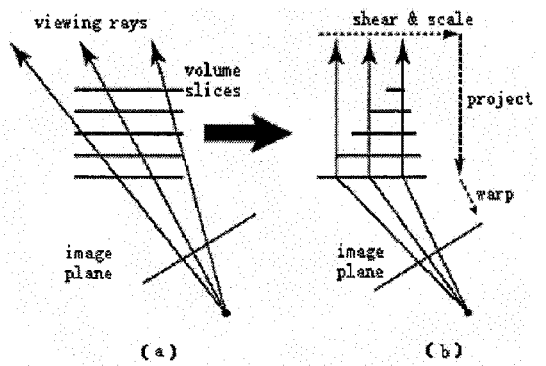

Fig. 1 - Perspective projection in Shear warp [28]

This technique starts by computing an intermediate coordinate system, this step is called "object space to shear object space" that simplifies the calculation of 3D projections, putting the view point parallel to a plane (eg the plane Z) [9, 28]. In the case of parallel projections this step only includes translations of the volume slices, but in the case of perspective projections a scaling of the volume slices is still needed (Figure 1-a) [28]. After these steps only a simple parallel projection of the "shear object space to image space" is needed (Figure 1-b) [28].

B. Other techniques to visualize volumetric data

B.1. Texture mapping

Texture mapping is often used when real-time is strictly necessary [29]. This technique was proposed by Cullip and Neumann in [30], which demonstrated the possibility to implement volume rendering interactively using 3D hardware texture acceleration.

There are two distinct implementation methods for texture mapping, the method of surface rendering that creates a mesh extracted from the volume, similarly to marching cubes method (described in *Section B.4.1.*) and the method of volume rendering which is a DVR, as the ray casting [31]. The surface rendering method has the benefit of being very efficient in current graphic cards, but is a CPU intensive task in the mesh calculation from the volume data. [31]. The volume rendering method using texture mapping has proved to be more flexible, especially with the evolution of graphics cards that have multiple GPUs (graphic processing units) [31].

While techniques such as ray casting consist in calculating the luminosity and opacity through ray shot in software, texture mapping techniques consists in rendering the data as textured parallel slices, leaving luminosity and opacity computation for the hardware, which allows real-time volume rendering, nevertheless with less image quality [32]. To improve the final image appearance several proposals have emerged as [32] and [33] which allow the efficient implementation of shadows and incorporate diffuse and specular illumination, increasing image realism.

This hardware acceleration technique was also used successfully in [34] to accelerate the ray casting technique, but with an excessive memory consumption, which was solved in [35]. To accelerate the ray casting technique using texture mapping is necessary to use an "proxy geometries" algorithm, as described in [31].

B.2. Visualize large volumes

Over recent years the size of volumetric data has grown dramatically and now are very common volumes 1024^3 or higher [36].

While the hardware has also progressed, is generally impossible to achieve interactivity in volume render with such dimensions. To solve this problem various solutions have emerged and one of the most studied was the technique of multiresolution data hierarchy.

B.2.1 Multiresolution data hierarchy

In this technique the data volume is divided into smaller subvolumes, allowing the visualization of data with different levels of detail using methods to achieve a balanced level-of-detail [37].

Recently have been presented new proposals using this technique, as [38] e [39], however this technique will always have need for additional space and added difficulties in switching the resolution in real time [36].

B.2.2 Volume data reduction

In [36] is present an alternative technique to multiresolution, called "volume data reduction" that consists in filling the areas of less interest sparsely. Among other advantages, this technique dramatically reduce the amount of memory used and implements a "focus + context" visualization, rendering areas of interest as focus (on higher quality) and the rest as context [36]. The methodology adopted by the authors consists in partitioning the volume into cubic regions and each region is classified with a level of importance, then, according to this information the volume is rendered with greater emphasis on the most important regions [36].

B.3. Slicing

Slicing is one of the most simple and powerful exploration tool, allowing the user to create a mental model of the data structure in three dimensions [40]. This technique makes a cut on volumetric data parallel to one of three axis or through an arbitrary orientation, allowing representation of the intersected data by the cutting plane into a 2D image [41].

B.4. Isosurfacing

The isosurfaces are very useful for use in conjunction with volumetric data, either to represent the volumetric data in an alternative way, for further processing, such as segmentation, volume slices (2D representation) and other operations. Until a few years ago, this technique was less computationally demanding than volume rendering, however currently this is not always the case and some volume rendering algorithms are more efficient than the algorithms to extract isosurfaces [5]. However depending on the types of data, the visualization goals, the detail needed and other parameters, the isosurface extracting techniques from volumetric data are still widely used and very useful.

B.4.1 Marching cubes

This technique was proposed by Lorensen and Cline in [42] and is probably the technique most used in isosurface extraction.

Considering a volumetric dataset, with a scalar value in each voxel, the marching cubes technique considers that these scalars are continuous values between voxels in all three dimensions [43]. And the isosurface with the scalar value α (α is called isovalue) is obtained by calculating a set of triangles that form one (if not continuous) or several isosurfaces separating higher α values and less α values [43]. In it's original version the isosurface calculation is performed in each cube (voxel) in an isolated forma [42] and for each cube vertex that has equal or greater value to α is classified as "marked", otherwise it is classified as "unmarked" [43]. So, with two chances per vertex and eight vertex there are 256 (2^8) possible ways of marking the cube, however applying reflective and rotational symmetry there are only 14 different cases [42]. The topology of each voxel is then stored in a look-up table and based on this topology the triangular surfaces that intersect individual cubes can then be calculated for example using linear interpolation [42].

This technique was also extended to non-rectangular data by He et al. in [44] and adapted to any number of dimensions in [45, 46].

B.5. Visualizing data with more than three dimensions

Over the last few years the use of volumetric data not only grew, but led to new requirements, including the representation of a fourth dimension, for example to represent the time when there are several samples of different times or to represent uncertainty or other parameters associated with the volumetric data. In the following section are shown solutions to represent a fourth dimension that represents the uncertainty associated with volumetric data, but can be easily adapted to other parameters that we wish to represent.

B.5.1. Visualization of uncertainty in volumetric data

Visualization is a powerful tool to represent data and information. Nevertheless, when the acquired data have some associated uncertainty, the final visualization may lead to erroneous conclusions, since users may consider interpolated data as acquired data [47]. To avoid this, and according to [48], uncertainty visualization techniques should be informative, intuitive, non distracting, and interactive.

Uncertainty representation in volumetric data is a four dimensional representation problem, with great interest in various scientific areas, for which several solutions have been proposed.

A general classification of uncertainty encoding methods into two groups is proposed in [49]: mapping uncertainty by an additional piece of data or integrating it in volumetric data (through color, transparency, etc.).

One of the first proposals for uncertainty representation involves the application of high levels of transparency in places where the data have more uncertainty, and greater opacity where data have greater certainty [50]. This technique is intuitive for the user, since more emphasis is given to data of greater certainty. Nevertheless, it has two major disadvantages [50, 51]: if the user cannot interactively activate and deactivate the visualization, it can lead to non-visualization of data with little certainty that may be relevant, and to ambiguities when data are represented by color, since when applying opacity to a color it will be represented in a lighter shade (on a light background) [50, 51].

To overcome the problems described in the previous method, Djurcilov, et al. [51] proposed three alternative methods to represent uncertainty:

- inserting speckles/holes — It consists in placing small speckles (glyphs) on the data representation, adding more speckles or larger speckles in locations where there is greater uncertainty [51, 52];
- adding noise — It consists in introducing noise in data according to the uncertainty associated with each voxel, by adding or subtracting a random value proportional to the uncertainty [51, 52];
- adding texture — It consists in applying a texture to the volume according to the uncertainty [51, 52];

III. INTERACTION

In visualization the interaction involves multiple aspects as number of frames per second achieved or the possibility to manipulate objects in real time. However this work seeks to address the interaction in a more restricted domain: the use of methods that modify the visualization or implement artifacts that allow information to be transmitted more efficiently to the end user.

A. Focus + context techniques

Over the past few years several interactive "Focus and Context" techniques have been presented, summarized in [53], as [54] where regions of most interest take higher resolution, or as [55] where no interest areas are discarded or the use of deformations for browsing volumetric data [56].

A.1. Visual filters – volumetric lenses

The concept of filtering using 3D lens derived from the work presented in [57] (2D Magic Lenses) where widgets 2D change rendering style. This is a "focus + context technique" [58], introduced in volume rendering by Viega et al. and led to several works that culminated in

interactive techniques to explore hidden data inside the volumetric data [40, 53, 59].

For example in [40] are shown a variety of applications to magic lenses. The user can demark an area of interest through a geometric shape (a cube, a sphere, etc.) and then is possible to apply three different classes of magic lenses [40]:

- clipping volumes – consists in extracting a volume region to view the inside of the volume at this location (Figure 2-a);
- showing different transfer function – involves the application of different TFs in voxels outside the area of interest (context) and voxels contained within the area of interest (Figure 2-b);
- drawer metaphor – performs the duplication of interest area and its representation outside the original volume, this area can also assume a different TF (Figure 2-c).

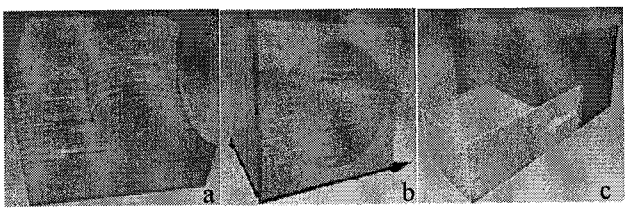

Fig. 2 - Volumetric lenses [40]

This technique has the advantage of allowing exploration of volumetric data in a flexible and dynamic way enhancing date of interest or suppress distracting information by changing the visual representation of the interest area in the volume without losing the view context [40].

A.2. Time navigation with 3D lenses

The Magic Lens technique was also applied to temporal exploration of volumetric data in [59] with the ability to navigate through time lines as shown in Figure 3.

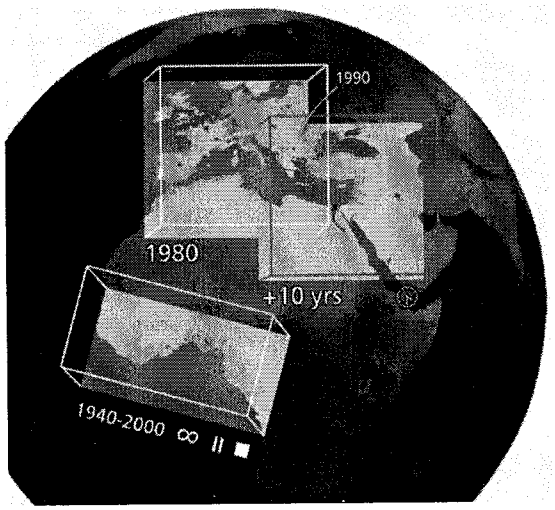

Fig. 3 - Time navigation with 3D lenses [59]

A.3. 3D orientation widget

During the exploration of a small portion of the volumetric data sometimes is difficult to understand exactly the position and orientation of the camera in the data, to solve this problem in [40] is proposed the integration of a widget in the visualization to represent these informations.

A.4. Deformations

M. J. McGuffin, et al. presented in [60] a technique for "looking inside volumetric data" using deformations as shown in Figure 4. In relation to cutting based techniques, that removes parts of the volume, this technique has the advantage of keeping these parts as context, deforming them because they may be potentially important [60].

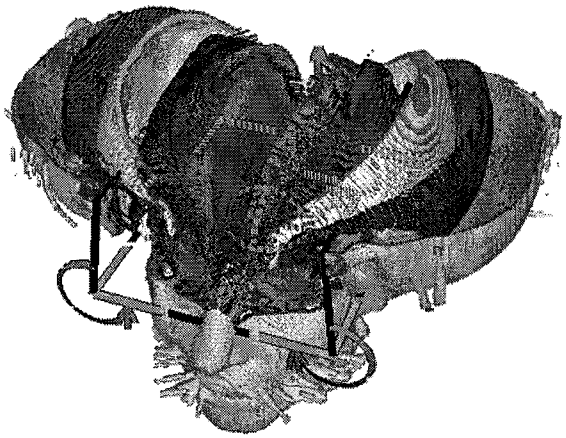

Fig. 4 - Visualize volumetric data through deformations [60].

B. Direct volume haptics (DVH)

DVH is the implementation of force feedback in direct volume rendering increasing the information given to the user through the exploration of our sense of touch [61].

According to [61] there are two ways to implement haptic in volumetric data: extracting an isosurface by intermediate geometry such as marching cubes [62] (this involves pre-processing) or through the volumetric data, which allows the haptic representation of all data instead of just a subset. However there are hybrid implementations that combine the advantages of both models as [63].

Haptics can be implemented with any input device and feedback is given by visualization. For example moving the mouse cursor from one point to another point in the volume is calculated a resistance that will reduce the speed of this movement, according to the characteristics of data [61], or in more natural way using touch screens [63]. This is also a very interesting technique for simulation of some practical cases such as medical training, allowing a significant increase of realism [63].

To reduce the effort in implementing this technique there are some APIs that implement volume haptics such as

Volume Haptics ToolKit [64] and the OpenHaptics™ Toolkit [65]. The OpenHaptics™ Toolkit stands out for having an easy integration with OpenGL.

IV. INTEGRATION OF VISUALIZATION AND INTERACTION

To implement interactive techniques mentioned above is necessary to use a visualization technique that will achieve a number of frame rates sufficient to achieve interactivity in the end-user device. So it is necessary know very well the techniques of visualization, among others aspects described below.

Table 1 summarizes the visualization techniques presented, however today it is very difficult to select the best technique for a particular case, in addition to described techniques it is very common to see hybrid implementations that take advantage of various techniques, and new refinements that reduce the resources used and increase the quality of final image such as [66, 67], or even proposals of new techniques or combinations of existing techniques like [19].

Whenever selecting a visualization technique several issues should be considered:

- Deciding the priority: high-quality images, high interactivity (with less quality), or an intermediate solution?
- Computational resources allocated to the end user (with more resources we can offer more quality or better interactivity);
- Each graphics library has its own particularities and implements only some of the techniques described or a combination of these techniques;
- Different graphics libraries have different implementations which also affects the final performance;
- Implementation through hardware can be extraordinarily more efficient;

Technique	1st Pub.	CPU usage	Image Quality	Interactivity
Ray tracing	1969	Very high	Excellent	Only in dedicated systems
Ray casting	> 1969	High	Very Good	Only on small/medium data size
Splatting	1989	Moderate / high	Medium	Frequently
Shear warp	1992	Moderate	Medium	Frequently
Texture mapping	1994	Moderate / Low	Medium / Low	Almost always

Table 1 - Synthesis of DVR techniques.

V. CONCLUSIONS

Currently the volume rendering technique is applied in medicine, industry, construction, and in many other areas. This technique already has a few decades, however only with the popularization and increase in computational power of computers its application to multiple areas became feasible.

Nevertheless, even at present, this technique is still computationally very demanding, since although the computing power has increased dramatically, methods of acquiring volumetric data also progressed, resulting in huge datasets every day larger and more difficult to analyze.

Over this work two major focus of research which are currently very active were identified: volume rendering through hardware and interactive methods that implement various methods to transmit information contained in a volumetric data in a more efficient way.

The former allows high interactivity levels, even for large datasets.

Interactive methods are more and more correlated with volume rendering through hardware, especially in the exploitation of today powerful GPUs, as data volumes are growing more and more, making it difficult to achieve interactivity without using GPUs.

REFERENCES

[1] F. Sroubek, M. Sorel, J. Boldys, and J. Sroubek, "PET image reconstruction using prior information from CT or MRI," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, 2009, pp. 2493-2496.

[2] R. W. Keach, J. H. McBride, and B. C. Pykles, "Petroleum industry techniques yield new insights into 3D GPR data," in *Ground Penetrating Radar (GPR), 2010 13th International Conference on*, 2010, pp. 1-5.

[3] L. Castanie, B. Levy, and F. Bosquet, "VolumeExplorer: roaming large volumes to couple visualization and data processing for oil and gas exploration," in *Visualization, 2005. VIS 05. IEEE*, 2005, pp. 247-254.

[4] J. Kniss, S. Premoze, C. Hansen, and D. Ebert, "Interactive translucent volume rendering and procedural modeling," presented at the Proceedings of the conference on Visualization '02, Boston, Massachusetts, 2002.

[5] K. Brodlie and J. Wood, "Recent Advances in Volume Visualization," *Computer Graphics Forum*, vol. 20, p. 125, 2001.

[6] D. Hearn and M. P. Baker, *Computer graphics with OpenGL*, Third Edition ed.: Pearson Education, Inc., 2004.

[7] F. d. M. Pinto and C. M. D. S. Freitas, "Volume visualization and exploration through flexible transfer function design," *Computers & Graphics*, vol. 32, pp. 540-549, 2008.

[8] J. Zhang, J. Sun, Y. Zhang, Q. Han, and Z. Jin, "A Parallel Volume Splatting Algorithm Based on PC-Clusters," in *Computational Science and Its Applications – ICCSA 2004*. vol. 3044, A. Laganà, M. L. Gavrilova, V. Kumar, Y. Mun, C. J. K. Tan, and O. Gervasi, Eds., ed: Springer Berlin / Heidelberg, 2004, pp. 272-279.

[9] C. Zhanfang, Z. Guoyu, and Z. Xiaopeng, "Research and Realization of a Volume Rendering Algorithm Based on Shear-Warp Volume Rendering Algorithm," in

Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on, 2009, pp. 1-4.

[10] A. Appel, "Some techniques for shading machine renderings of solids," presented at the Proceedings of the April 30--May 2, 1968, spring joint computer conference, Atlantic City, New Jersey, 1968.

[11] T. Whitted, "An improved illumination model for shaded display," *Commun. ACM*, vol. 23, pp. 343-349, 1980.

[12] R. L. Cook, T. Porter, and L. Carpenter, "Distributed ray tracing," *SIGGRAPH Comput. Graph.*, vol. 18, pp. 137-145, 1984.

[13] T. L. Kay and J. T. Kajiya, "Ray tracing complex scenes," *SIGGRAPH Comput. Graph.*, vol. 20, pp. 269-278, 1986.

[14] I. Georgiev and P. Slusallek, "RTfact: Generic concepts for flexible and high performance ray tracing," in *Interactive Ray Tracing, 2008. RT 2008. IEEE Symposium on*, 2008, pp. 115-122.

[15] S. Boulos, "Notes on efficient ray tracing," presented at the ACM SIGGRAPH 2005 Courses, Los Angeles, California, 2005.

[16] R. Yagel, D. S. Ebert, J. N. Scott, and Y. Kurzion, "Grouping volume renderers for enhanced visualization in computational fluid dynamics," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 1, pp. 117-132, 1995.

[17] R. Westermann and B. Sevenich, "Accelerated volume ray-casting using texture mapping," in *Visualization, 2001. VIS '01. Proceedings*, 2001, pp. 271-278.

[18] S. Lim and B.-S. Shin, "A distance template for octree traversal in CPU-based volume ray casting," *The Visual Computer*, vol. 24, pp. 229-237, 2008.

[19] X. Jian, L. Ke, and T. Jie, "An efficient out-of-core volume rendering method based on ray casting and GPU acceleration," in *Information, Computing and Telecommunication, 2009. YC-ICT '09. IEEE Youth Conference on*, 2009, pp. 130-133.

[20] G. Jing and W. Song, "An Octree Ray Casting Algorithm Based on Multi-core CPUs," in *Computer Science and Computational Technology, 2008. ISCSCT '08. International Symposium on*, 2008, pp. 783-787.

[21] G.-j. Ma and Y.-s. Zhang, "Hierarchical Octree and Sub-Volume Texture Block Projection for GPU Accelerated Ray Casting Volume Rendering," in *Biomedical Engineering and Computer Science (ICBECS), 2010 International Conference on*, 2010, pp. 1-4.

[22] F. Rossler, R. P. Botchen, and T. Ertl, "Dynamic Shader Generation for GPU-Based Multi-Volume Ray Casting," *Computer Graphics and Applications, IEEE*, vol. 28, pp. 66-77, 2008.

[23] L. Westover, "Interactive volume rendering," presented at the Proceedings of the 1989 Chapel Hill workshop on Volume visualization, Chapel Hill, Noth Carolina, United States, 1989.

[24] K. Mueller, N. Shareef, H. Jian, and R. Crawfis, "High-quality splatting on rectilinear grids with efficient culling of occluded voxels," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 5, pp. 116-134, 1999.

[25] S. Zhigang, Z. Jiawan, S. Jizhou, and W. Zunce, "A new view-buffer splatting algorithm," in *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on*, 2003, pp. 1399-1402 vol.2.

[26] L. Westover, "Footprint evaluation for volume rendering," presented at the Proceedings of the 17th annual conference on Computer graphics and interactive techniques, Dallas, TX, USA, 1990.

[27] P. Lacroute and M. Levoy, "Fast volume rendering using a shear-warp factorization of the viewing transformation," presented at the Proceedings of the 21st annual conference on Computer graphics and interactive techniques, 1994.

[28] M. Lin Xiao, J. Li, and X. Yu Xiu, "3D Visualization technology Based on BCC-Grid Shear-Warp Algorithm," in *Computational Engineering in Systems Applications, IMACS Multiconference on*, 2006, pp. 1454-1459.

[29] B. Cabral, N. Cam, and J. Foran, "Accelerated volume rendering and tomographic reconstruction using texture mapping hardware," presented at the Proceedings of the 1994 symposium on Volume visualization, Tysons Corner, Virginia, United States, 1994.

[30] T. J. a. N. Cullip, Ulrich, "Accelerating Volume Reconstruction With 3D Texture Hardware," University of North Carolina at Chapel Hill, 1994.

[31] F. Zhang and M. Xie, "Real-time Medical Image Volume Rendering Based on GPU Accelerated Method," in *Computational Intelligence and Design, 2008. ISCID '08. International Symposium on*, 2008, pp. 30-33.

[32] U. Behrens and R. Ratering, "Adding shadows to a texture-based volume renderer," in *Volume Visualization, 1998. IEEE Symposium on*, 1998, pp. 39-46.

[33] A. V. Gelder and K. Kim, "Direct volume rendering with shading via three-dimensional textures," presented at the Proceedings of the 1996 symposium on Volume visualization, San Francisco, California, United States, 1996.

[34] R\, \#252, d. Westermann, and B. Sevenich, "Accelerated volume ray-casting using texture mapping," presented at the Proceedings of the conference on Visualization '01, San Diego, California, 2001.

[35] S. Grimm, S. Bruckner, A. Kanitsar, and E. Groller, "Memory Efficient Acceleration Structures and Techniques for CPU-Based Volume Raycasting of Large Data," presented at the Proceedings of the 2004 IEEE Symposium on Volume Visualization and Graphics, 2004.

[36] Y. Wang, C. Wang, T. Lee, and K. Ma, "Feature-Preserving Volume Data Reduction and Focus+Context Visualization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. PP, pp. 1-1, 2010.

[37] C. Wang and H. W. Shen, "LOD Map - A Visual Interface for Navigating Multiresolution Volume Visualization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 12, pp. 1029-1036, 2006.

[38] W. Chaoli, G. Jinzhu, L. Liya, and S. Han-Wei, "A multiresolution volume rendering framework for large-scale time-varying data visualization," in *Volume Graphics, 2005. Fourth International Workshop on*, 2005, pp. 11-223.

[39] H. Xavier and T. Sebastien, "Chapter 6: Multiresolution Representation and Deformation of Very Large Volume Datasets Based on Haar Wavelets," in *Geometric Modeling*

and Imaging, 2008. GMAI 2008. 3rd International Conference on, 2008, pp. 34-40.

[40] T. Ropinski and K. Hinrichs, "Interactive Volume Visualization Techniques for Subsurface Data," in *Visual Information and Information Systems*. vol. 3736, S. Bres and R. Laurini, Eds., ed: Springer Berlin / Heidelberg, 2006, pp. 121-131.

[41] R. Machiraju and R. Yagel, "Accuracy control of reconstruction errors in volume slicing," in *Biomedical Visualization, 1995. Proceedings.*, 1995, pp. 50-57.

[42] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," presented at the Proceedings of the 14th annual conference on Computer graphics and interactive techniques, 1987.

[43] T. S. Newman and H. Yi, "A survey of the marching cubes algorithm," *Computers & Graphics*, vol. 30, pp. 854-879, 2006.

[44] M. He, B. Xiong, and H. Yu, "Multi-resolution surface reconstruction," in *Image Processing, 2004. ICIP '04. 2004 International Conference on*, 2004, pp. 1971-1974 Vol. 3.

[45] P. Bhaniramka, R. Wenger, and R. Crawfis, "Isosurfacing in higher dimensions," presented at the Proceedings of the conference on Visualization '00, Salt Lake City, Utah, United States, 2000.

[46] P. Bhaniramka, R. Wenger, and R. Crawfis, "Isosurface Construction in Any Dimension Using Convex Hulls," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, pp. 130-141, 2004.

[47] A. Streit, P. Binh, and R. Brown, "A Spreadsheet Approach to Facilitate Visualization of Uncertainty in Information," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, pp. 61-72, 2008.

[48] G. Grigoryan and P. Rheingans, "Probabilistic surfaces: point based primitives to show surface uncertainty," presented at the Proceedings of the conference on Visualization '02, Boston, Massachusetts, 2002.

[49] A. Pang, "Visualizing Uncertainty in Geo-spatial Data," *In Proceedings of the Workshop on the Intersections between Geospatial Information and Information Technology*, 2001.

[50] T. Zuk and S. Carpendale, "Theoretical analysis of uncertainty visualizations," 2006, p. 606007.

[51] S. Djurcilov, K. Kim, P. F. J. Lermusiaux, and A. Pang, "Volume rendering data with uncertainty information," in *Data Visualization 2001*, D. Ebert, J. M. Favre, and R. Peikert, Eds., ed Vienna: Springer-Verlag Wien, 2001.

[52] S. Djurcilov, K. Kim, P. Lermusiaux, and A. Pang, "Visualizing scalar volumetric data with uncertainty," *Computers & Graphics*, vol. 26, pp. 239-248, 2002.

[53] L. Wang, Y. Zhao, K. Mueller, and A. Kaufman, "The magic volume lens: an interactive focus+context technique for volume rendering," in *Visualization, 2005. VIS 05. IEEE*, 2005, pp. 367-374.

[54] M. Levoy and R. Whitaker, "Gaze-directed volume rendering," *SIGGRAPH Comput. Graph.*, vol. 24, pp. 217-223, 1990.

[55] B. Pflesser, U. Tiede, K. H. H\, \#246, and hne, "Towards Realistic Visualization for Surgery Rehearsal," presented at the Proceedings of the First International Conference on Computer Vision, Virtual Reality and Robotics in Medicine, 1995.

[56] M. J. McGuffin, L. Tancau, and R. Balakrishnan, "Using Deformations for Browsing Volumetric Data," presented at the Proceedings of the 14th IEEE Visualization 2003 (VIS'03), 2003.

[57] E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose, "Toolglass and magic lenses: the see-through interface," presented at the Proceedings of the 20th annual conference on Computer graphics and interactive techniques, Anaheim, CA, 1993.

[58] J. Looser, R. Grasset, and M. Billinghurst, "A 3D Flexible and Tangible Magic Lens in Augmented Reality," presented at the Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007.

[59] W. B. Christoph, "Real-Time Rendering Method and Performance Evaluation of Composable 3D Lenses for Interactive VR," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, pp. 394-410, 2010.

[60] M. J. McGuffin, L. Tancau, and R. Balakrishnan, "Using deformations for browsing volumetric data," in *Visualization, 2003. VIS 2003. IEEE*, 2003, pp. 401-408.

[61] K. L. Palmerius, M. Cooper, and A. Ynnerman, "Haptic Rendering of Dynamic Volumetric Data," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, pp. 263-276, 2008.

[62] S. H. Rizzi, C. J. Luciano, and P. P. Banerjee, "Haptic interaction with volumetric datasets using surface-based haptic libraries," in *Haptics Symposium, 2010 IEEE*, 2010, pp. 243-250.

[63] K. Laehyun, G. S. Sukhatme, and M. Desbrun, "A haptic-rendering technique based on hybrid surface representation," *Computer Graphics and Applications, IEEE*, vol. 24, pp. 66-75, 2004.

[64] W. R. Mark, S. C. Randolph, M. Finch, J. M. V. Verth, and I. Russell M. Taylor, "Adding force feedback to graphics systems: issues and solutions," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[65] I. Brandon, "The OpenHaptics™ Toolkit: A Library for Adding 3D Touch™ Navigation and Haptics to Graphics Applications," 2005, pp. 590-591.

[66] L. Bin, O. Shanxing, and T. Lianfang, "A Rapid Pre-Integrated Perspective Volume Rendering Algorithm," in *Bioinformatics and Biomedical Engineering (iCBBE), 2010 4th International Conference on*, 2010, pp. 1-4.

[67] B. Li, L.-f. Tian, S. Ou, and W. Lifei, "Rapid Pre-Integrated Perspective Volume Rendering algorithm of medical data sets," in *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, 2010, pp. 5999-6003.