

Volume Animation

Edward Dale*
Computer Animation 20052
Professor Joe Geigel†

14 February 2006

Abstract

This paper surveys a number of techniques used in volume animation. Data often come in forms that are not particularly suited to a polygon representation. Medical scans, protein structure, and simulation data are all datasets that are best rendered as volumes. The problem then becomes how to represent and animate the data. Techniques discussed include skeleton determination, spatial transfer functions, and volume keyframing. A special focus is made on applications to galactic and stellar visualization.

Contents

1	Introduction	2
2	Volume Rendering	2
3	Galactic Animation Considerations	3
4	Skeletons	3
4.1	Skeleton Extraction	4
4.2	Skeleton Uses	4
5	Spatial Transfer Functions	5
5.1	Volume Scene Graphs	6
5.2	Galactic Animations	6
6	Volume Keyframing	6
6.1	Volume Features	7
7	Conclusion	7

*erd4819@cs.rit.edu

†jmg@cs.rit.edu

1 Introduction

Volume animation is an important and useful technique because data resulting from scientific simulations and observations is often volumetric. This necessitates a different approach toward animation. While a traditional *render, advance t, repeat* method might work for temporal volumetric data, animations of purely spatial data require some animator controls to be provided. Much of this field involves creating volume analogues to traditional, polygon animation methods.

The rest of this paper is as follows. First, in section 2, an introduction to volume rendering will be discussed. In section 3, some considerations of animating stars and galaxies will be discussed. Next, skeleton-based methods of animation will be presented in section 4. In section 5, the concept of a spatial transfer function is introduced. In section 6, the volume analogue of keyframing is presented followed by section 7 with some concluding remarks.

2 Volume Rendering

Volume rendering is a necessary and useful tool because the data that results from scientific simulations and observations does not often lend itself to the tradition polygon rendering methods of computer graphics. For our purposes, the volumes being rendered will be considered three-dimensional datasets containing scalar or vector fields. The scalar values can represent density, temperature, etc.. Representing these values in polygonal form produces hard to understand images resulting from the fuzziness of boundaries and the possibility of nested areas of interest. Volume rendering solves these problems using a number of well-developed methods [Dal06].

The two main categories of volume rendering are indirect and direct. Indirect volume rendering (IVR) extracts isosurfaces from the volume, polygonizes those surfaces, and renders them using traditional means [Elv92]. Direct volume rendering (DVR) can be done using ray

casting or splatting, among others. Ray casting is the volume analogue of ray tracing; a ray is followed through the volume sampling the voxels it passes. Splatting involves projecting each voxel onto the image plane using a gaussian kernel that is scaled by the distance of the voxel from the image plane. Both of these methods utilize a transfer function to map the value stored in the voxel to the color and opacity that needs to be displayed [KW03].

3 Galactic Animation Considerations

Galactic animation is a focus of this paper in anticipation of the author completing a Masters thesis in the subject. As such, wherever possible, applications will be made to galactic animation. Such parallels are rare because of the lack of possible uses for animator-controlled animation of galactic simulations. Volume representation is very useful for galaxies because of their nature as “structures of varying depth that attenuate traversing light” [Elv92].

4 Skeletons

Skeleton extraction and manipulation is a very intuitive technique for volume animation. The main benefit being the large body of algorithms and techniques for dealing with articulated skeletons stemming from the research done regarding character animation. Skeletons also capture the essential topology of an object. For our purposes, a skeleton consists of the set of voxels that form the medial surface (centered with relation to the boundaries) of an object. A centerline is curve-like representation of the medial surface. The skeleton and centerline need not be completely connected, however, a later step in the algorithm discussed below ensures this. Both of these representations have a single parameter that determines how thin the resulting object should be [GKHS98].

Extracting the skeleton or centerline of a volume dataset is the primary challenge, which

will be discussed in section 4.1. Given a thinned representation, there are a number of novel applications to volume animation that will be discussed in section 4.2.

4.1 Skeleton Extraction

Extracting a skeleton from a volume is done by thinning that volume until a desired thinness is reached. This is the thinning parameter that was mentioned earlier. The method described in [GKHS98] uses a weighted distance transform that eliminates the need for costly floating point operations. The algorithm proceeds by propagating boundary distances inward until there are no new voxels. For each voxel, the average of all its neighbors is calculated and compared with the thinning parameter to determine if it is a skeletal voxel.

The previous paragraph yields a collection of voxels that likely do not form a connected skeleton. This is resolved by creating a completely connected graph with the voxels as vertices and a linear combination of spatial distance and distance transform difference as the edge weight. The connected skeleton is then the minimum spanning tree of this graph.

Determining the centerline from disconnected skeletal voxels is done using a recursive midpoint algorithm. It is initiated using the two desired end-points of the centerline and finishes when the distances fall below a *fineness* threshold.

4.2 Skeleton Uses

The generated skeleton can be used as one would use a skeleton representing any other articulated figure. In particular, importation into an advanced character animation program such as Maya has proven useful. The problem then becomes how to apply skeleton deformations to the original volume. This is resolved by growing each skeletal voxel into a sphere of radius equal to the distance transform value [GS97]. This will, however, lose the texture of the original volume. To maintain the texture, a bounding box is formed around the deformed

skeleton segment and is used to determine the original texture. The mid-plane algorithm is used to prevent breakages from occurring at points of large rotation [SSC03]. The volume can also be further deformed by manually displacing the skeletal voxels before reconstruction. These displacements don't provide much use for galactic animation because of their inherent distorting effect on the data.

The best use for the generated centerline is as a fly-through path for a camera. In this way, endoscopy can be simulated. This is also the primary application of skeleton extraction to galactic animation. It would be desirable to be able to follow the arm of a spiral galaxy such as our Milky Way towards the center and then out the other side.

5 Spatial Transfer Functions

The process of rendering a volume involves mapping the value at the point in the volume to a color and opacity in a step called classification. This mapping is done using a transfer function [MHB⁺00]. In [CSW⁺03], spatial transfer functions, transfer functions that map points in space to other points in space is presented. A spatial transfer function would be used before the classification step to change the voxel being sampled. These functions can be piece-wise functions that can segment off one part of the volume and rotate it away from the other to reveal inner structure.

To apply spatial transfer functions to animation, a temporal aspect needs to be added yielding a temporal spatial transfer function. This can be implemented with a spatial transfer function that varies with time or a number of spatial transfer functions that are interpolated between.

5.1 Volume Scene Graphs

Spatial transfer functions are discussed in the framework of volume scene graphs in [CSW⁺03]. These are the volume analogue of traditional scene graphs, “a hierarchical organization of shapes, groups of shapes, and groups of groups that collectively define the content of a scene” [Nad00]. Volume scene graphs contain procedurally- and data-defined volumetric shapes and functions that can operate on them. A spatial transfer function would be a node in a volume scene graph with leaf nodes being volumetric shapes. The scene graph allows its contents to be voxelized, the result of which can be sent to a volume renderer. Voxelization is done lazily and only when needed to improve animation interactivity.

5.2 Galactic Animations

The versatility that spatial transfer functions provide open up a lot of possibilities for galactic animations. Simple slicing could be used to look at different halves of a galaxy. More complex splits could be created that split on a curve following the areas of high density allowing a spiral galaxy to be opened up along one of arms of the spiral. For a cinematic effect, the arms of the galaxy could “spin off” into space.

6 Volume Keyframing

Another traditional technique that one would like to work with on volumes is keyframing. Keyframing is a classic and intuitive method for specifying animation where the location of important points is defined for each keyframe and interpolated to produce the in-between frames. In volumes, it’s difficult to define an important voxel and even harder to define how a change in that voxel’s position will affect the rest of the volume. One solution to this problem is to build a skeleton by thinning as discussed in section 4 and then keyframe the joints of the skeleton. In cases where the volume is being built from scratch, it would be

desirable to have the skeleton built automatically; this is where volume features come in.

6.1 Volume Features

Features are “collection[s] of voxels that [have] some specific meaning to the modeler” [CMMM00]. The model (volume) is created by means of a sequence of sculpting operations, each of which needs to be recorded in order to keep full information about the model. Each one of these sculpting operations corresponds to none-to-many features (eyes, face, table leg, etc.) defined by the animator. Features are manually combined into a hierarchical skeleton structure, allowing manipulation using the usual methods. Minkowski operators are introduced as shape operators used to combine and decompose sets of voxels. Regularized versions are also introduced that provide isolation of feature changes. Using regularized Minkowski operators allow features to be modified without neighboring features being affected.

7 Conclusion

A number of techniques for volume animation were discussed. This is an interesting area because increased computing power yields larger multi-dimensional datasets that need to be analyzed. Skeleton extraction and manipulation is the primary means of deforming volumes. Volume construction via sculpting operations can also yield a skeleton that is manipulated using the same means. Spatial transfer functions provide a means to procedurally define transformations.

References

- [CMMM00] V. Chandru, N. Mahesh, M. Manivannan, and Swami Manohar. Volume sculpting and keyframe animation system. In *CA*, pages 134–139, 2000.
- [CSW⁺03] M. Chen, D. Silver, A. S. Winter, V. Singh, and N. Cornea. Spatial transfer functions: a unified approach to specifying deformation in volume modeling and animation. In *VG '03: Proceedings of the 2003 Eurographics/IEEE TVCG Workshop on Volume graphics*, pages 35–44, New York, NY, USA, 2003. ACM Press.
- [Dal06] Edward Dale. Volume rendering. For Computer Graphics 2 course, February 2006.
- [Elv92] T. Todd Elvins. A survey of algorithms for volume visualization. *SIGGRAPH Comput. Graph.*, 26(3):194–201, 1992.
- [GKHS98] Nikhil Gagvani, D. Kenchammana-Hosekote, and D. Silver. Volume animation using the skeleton tree. In *VVS '98: Proceedings of the 1998 IEEE symposium on Volume visualization*, pages 47–53, New York, NY, USA, 1998. ACM Press.
- [GS97] N. Gagvani and D. Silver. Parameter controlled skeletonization of three dimensional objects, 1997.
- [KW03] J. Kruger and R. Westermann. Acceleration techniques for gpu-based volume rendering. In *VIS '03: Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*, page 38, Washington, DC, USA, 2003. IEEE Computer Society.
- [MHB⁺00] Michael Meißner, Jian Huang, Dirk Bartz, Klaus Mueller, and Roger Crawfis. A practical evaluation of popular volume rendering algorithms. In *VVS '00: Proceedings of the 2000 IEEE symposium on Volume visualization*, pages 81–90, New York, NY, USA, 2000. ACM Press.
- [Nad00] David R. Nadeau. Volume scene graphs. In *VVS '00: Proceedings of the 2000 IEEE symposium on Volume visualization*, pages 49–56, New York, NY, USA, 2000. ACM Press.
- [SSC03] V. Singh, D. Silver, and N. Cornea. Real-time volume manipulation. In *VG '03: Proceedings of the 2003 Eurographics/IEEE TVCG Workshop on Volume graphics*, pages 45–51, New York, NY, USA, 2003. ACM Press.