Volume Rendering on one RLE compressed data set by a new combination of Ray-Casting and Shear-Warp

Gregor Schlosser, Jürgen Hesser, Reinhard Männer*
Chair of Computer Science V, University of Mannheim

1 Introduction

In recent years, a growing demand to visualize ever larger (e.g. gigabyte) data sets poses the problem of handling large volumes and to render them in interactive times on a normal PC. One approach is based on lossy compression schemes.It compromises between high compression ratio, fast run-time random access and visualization quality. Our new approach based on shear-warp volume rendering, doesn't have any visible artefacts due to quantization and after classification of the volume we achieve high compression ratios as well. This hybrid algorithm combines the advantages of image-order and object-order algorithms. Further, we are able to preserve the benefits from Lacroutes original implementation. Like, volume rendering from run-length-encoded (RLE) data, alignment of volume and intermediate image (IIM) scanlines, only one object-order traversal of the volume, efficient and synchronized space leaping and early-ray-termination (ERT) using RLE data structures, preservation of the final warp step for efficient calculations. Nevertheless, we solved two problems found in the original implementation. Firstly, we removed the 3-fold redundancy of classified volumes (RLE data sets). Therefore it was necessary to develop a new rendering algorithm for one main viewing direction. We achieve better image quality (also preserving the Shannon-Nyquist theorem), using a ray-casting approach within 4 RLE scanlines and new data structures for efficient ERT. Secondly, we increased the rendering quality for the other two main viewing directions by extending the discrete volume grid to continuous space by Hermite polynomials.

2 New Volume Rendering Technique

The original implementation of the shear-warp algorithm uses 3 classified volumes for rendering, one for each major viewing direction. One RLE data set can be removed by using an array of pointers to every RLE scanline.However, to eliminate another encoding we had to develop a completely new rendering method for one major viewing direction, since the scanlines of the volume and the scanlines of the IIM are not parallel anymore. Therefore we have constructed two new data structures, which describe a bijective mapping between object space (voxels) and image space (IIM pixels). The first one represents all information within a column of the volume (denote a 2×2 arrangement of RLEs a column in the following). Due to parallel projection it suffices to store only a single such data structure. After space-leaping or ERT this template allows a direct access to sample point positions, its weights for trilinear interpolation, ray segments, the number of sample points on ray segments and the IIM pixels. The second data structure was developed to implement an efficient ERT technique. To simplify discussion we first describe two special cases: If we look along the 3rd viewing axis (x), we have only one ray segment per column and since the volume RLE scanlines will be processed along the projection direction, we have to consider the IIM pixel only once (no RLE of the IIM). If we rotate the volume about only one axis (y or z), the IIM scanlines pass off either horizontally or vertically. Now, instead of skipping opaque pixels (rays), the dynamic RLE of the IIM skips opaque ray segments. In the general case the user rotates the volume aboute two axes (y and z) simultaneously, concerning the projection of the volume scanline the IIM scanline now passes off oblique. Therefore, we have encoded the IIM scanlines in that way. Hence, the second data structure represents a projected scanline of the volume onto the IIM. Let us assume we have just composited some sample points and now we are at the beginning of a transparent voxel run. After space leaping, we use the column template to find the current sample point and the affected IIM pixels. Using the second template, one can skip (ERT), say m, opaque IIM pixels along oblique, discrete lines (Bresenham) and finally we can find the corresponding voxel cuboid where we should start compositing.

To suppress artefacts when viewing along the two other directions, Sweeney and Mueller proposed a method which interpolates one intermediate slice half-way between two adjacent volume slices. We developed a numerical continuation of the discrete data set, a Hermite polynomial between sample points in the two adjacent volume slices. It is determined by constraints on two sample points $\vec{p}_1$, $\vec{p}_2$ and their tangent vectors $\vec{n}_1, \vec{n}_2$ at these sample points:

$$\begin{aligned} Q(t) &= (2t^3 - 3t^2 + 1)\vec{p}_1 + (-2t^3 + 3t^2)\vec{p}_2 \\ &+ (t^3 - 2t^2 + t)\vec{n}_1 + (t^3 - t^2)\vec{n}_2 \end{aligned} \quad (1)$$

where the parameter $t \in [0,1]$ specifies an intermediate point and tangent slope $dQ(t)/dt$ at the curve.

3 Conclusions and Results

Our new algorithm completely removes the 3-fold redundancy of the RLE data sets (memory usage $\approx O(N/3)$, even though the rendering speed is reduced by a factor of 2 up to 4 compared to the orginal algorithm). The extra overhead for rendering originates from the higher sampling rate ($\sqrt{3}$ if viewing along the major diagnonal) and a true tri-linear interpolation (factor of 2), both techniques that lead to a significant higher image quality. Otherwise we would achieve 85 % of the original implementation performance. Further, to suppress artefacts at 45 degree transitions, when viewing along the two other axes, we have incorporated a new method of intermediate slices by using Hermite curves, where the artefacts are decreased at the cost of about 30-40 % reduced frame rate. Finally, the data structures also allow speeding up object-based rendering (e.g. splatting) by a more efficient implementation of ERT.

*e-mail:sgregor@rumms.uni-mannheim.de