8
3D RENDERING METHOD FOR MRI IMAGE: A SURVEY

Harja Santana Purba, Daut Daman, Ghazali Sulong

INTRODUCTION

Magnetic Resonance Imaging (MRI) is the most common systems used in acquiring detailed anatomical information in medical imaging. The key feature of the imaging technologies is their ability to provide detailed information about the anatomical structure and abnormalities. MRI images are obtained by varying the number and sequence of pulsed radio frequency field in order to take advantage of magnetic relaxation properties of hard and soft tissues. Specifically a strong magnetic field is generated to cause atoms inside the body to become aligned. After alignment, a radio wave is issued to activate the atoms. Once the radio signal is turned off, the atoms give off a small characteristic signal. Those signals are then measured with a sensitive antenna called an MRI coil. This process is repeated many times until enough measurements are detected to create a series of detailed images. MRI does not use any ionizing radiation, and can create images of almost any body part oriented in any direction. Figure 8.1 shows a MRI example of a head.

This chapter is a survey of the literature involving methods for rendering volumetric MRI image. Over the years many techniques have been developed to visualize volumetric data. Since methods for displaying geometric primitives were already well-established, most of the early

methods involve approximating a surface contained within the data using geometric primitives. Common methods include raycasting (Levoy 1988), splatting (Westover 1990), shear-warp (Lacroute and Levoy 1994), and 3D texture-mapping (Rezk et al. 2000)(Cabral et al. 1994). These algorithms fit geometric primitives, such as polygons or patches, to constant-value contour surfaces in volumetric datasets. After extracting this intermediate representation, hardware-accelerated rendering can be used to display the surface primitives. In general, these methods require to make a decision for every data sample whether or not the surface passes through it. As information about the interior of objects is generally not retained, a basic drawback of these methods is that one dimension of information is essentially lost.

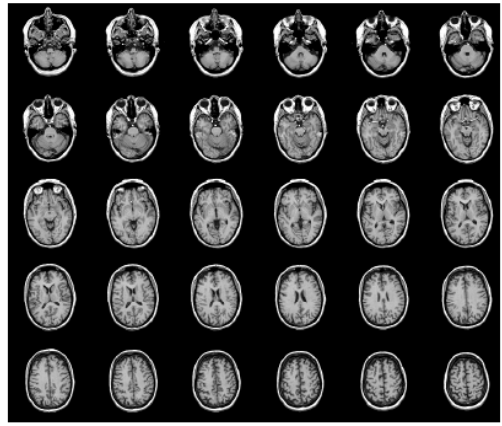

Figure 8.1 Medical Image of Skull

3D RENDERING METHODS

In general, a volumetric dataset consists of samples arranged on a regular grid. These samples are also referred to as voxels. While most volume rendering techniques are based on the theoretical framework presented in Section 2.1, several different techniques that implement this optical model have emerged. In this chapter, we concern with the direct volume rendering approach and with volumetric datasets that are described on cubic and uniform rectilinear grids, which is grid with anisotropic spacing along the three grid axes. Datasets of this nature are commonly obtained by means of volumetric scanning device, such as MRI. Four popular techniques in this field are Raycasting (Levoy 1988), Splatting (Westover 1990), Shear-warp (Rezk et al. 2000), and 3D texture-mapping approaches (Rezk et al. 2000)(Cabral 1994).

Raycasting

Raycasting (Levoy 1988) is an image-order algorithm that casts viewing rays through the volume. The image-order approach to volume rendering determines the data samples which contribute to each pixel on the image plane. At discrete intervals along the ray, the three-dimensional function is reconstructed from the samples and the optical model is evaluated. A ray starting at an image pixel is cast through the volume, evaluating the optical model at each resample location. This process is illustrated in Figure 8.2. The volumetric Ray Casting algorithm sends a ray into the scene for each pixel on the object (Figure. 8.2a). Starting at the point where the ray enters the volume (Figure. 8.2b),

the ray is followed while sampling the volume at constant distances (Figure. 8.2d). It accumulates the colors and opacities of these sample values. As the accumulation is performed in front-to-back order, viewing rays that have accumulated full opacity can be terminated. Processing of occluded regions is effectively avoided and is one of the main advantages of ray-casting.

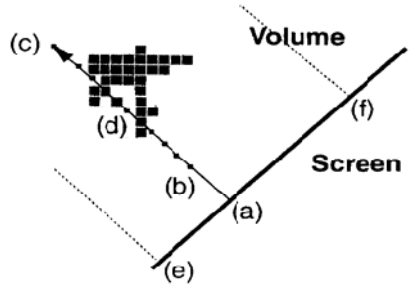

Figure 8.2 Illustration of ray-casting

One challenge in ray-casting is the efficient skipping of non-contributing which is the regions that have been classified as transparent regions. The ray is no longer followed when the value cannot change significantly, that is when it has accumulated an opaque color or when it is no longer inside the volume (Figure. 8.2c). This has a major performance impact as typical medical datasets commonly contain a large number of transparent voxels,. Numerous approaches for improving the performance of ray-casting have been presented. Most of them rely on one of the following principles or more: Image-Space Coherency,

Object-Space Coherency, Inter-Ray Coherency, Inter-Frame Coherency, Empty Space Skipping, and Efficient Memory Access.

The first principle is Image-Space Coherency. The probability of finding another pixel that has the same or similar color between two similar pixels is high. This observation is exploited by adaptive refinement (Levoy 1990). The method works by initially casting rays only from a subset of screen pixels. “Empty” pixels residing between pixels with similar values are assigned an interpolated value.

Another raycasting principle is Object-Space Coherency. Datasets contain regions with uniform or similar values. One way to increase the performance of raycasting is to avoid sampling within these regions. In an approach by van Walsum et al. (1992) a ray starts by sampling the volume at low frequency. If a large value difference is encountered between two adjacent samples, additional samples are taken. This idea can be extended to lower the sample rate in regions where only small contributions of opacity are made.

The third principle is Inter-Ray Coherency. For orthographic viewing the increased coherency between rays can be exploited. Although rays may have different origin, they have the same slope. To avoid computations involved in advancing the ray through voxel space, the idea of template-based raycasting has been presented (Yagel and Kaufman 1992). The sample points encountered by a ray are pre-computed and stored in a template. All rays can then be generated by applying the ray template.

The next principle is Inter-Frame Coherency. In interactive application viewing the differences between subsequent frames are usually small. The C-Buffer

approach (Yagel and Shi 1993) works by storing, at each pixel location, the object-space coordinates of the first non-empty voxel hit by the corresponding ray. This information is used to estimate the initial position of a ray in the consecutive frame. For each change of viewing parameters, the C-Buffer is transformed accordingly. In the case of rotation, a transformed buffer goes through a process of eliminating coordinates that might have become hidden.

Empty Space Skipping is another principle of raycasting. As datasets usually contain large regions which are classified as transparent, several methods have been suggested to rapidly traverse empty space. Levoy presented an approach called hierarchical spatial enumeration (Levoy 1990). The algorithm first creates a binary pyramid of the volume, which encodes empty and non-empty space. Raycasting is started at the top level of the pyramid. Whenever a ray reaches a non-empty cell, the algorithm moves down one level, entering whichever cell encloses the current location. Otherwise, the intersection point with the next cell is calculated and the ray is forwarded to this position. Following this idea, a min-max octree based on the volume's data values can be generated. This octree can be used to efficiently create the pyramid data structure whenever the classification changes. Another approach is space leaping (Cohen and Sheffer 1994)(Stander and Hart 1994)(Freund and Sloan 1997). Here, a distance transform is applied to the volume to calculate a proximity or skip value for each empty cell which encodes the distance to the nearest opaque cell. The value therefore is the distance that can be safely skipped along any ray that samples this cell. A drawback of this method is that it requires extensive processing every time the transfer function is changed.

The last principle is Efficient Memory Access. For large datasets, memory access has a considerable impact on

the overall processing time of a raycasting algorithm. The most simple memory layout for raycasting is a three-dimensional array. However, this may lead to view-dependent render times, due to changing memory access patterns for different viewing directions. This can greatly affect the performance for large datasets. Another common storage scheme is bricking (Parker et al. 1999), where the volume data is stored as sub-cubes of a fixed size. In general, this approach reduces the view dependent performance variations but does not increase the memory consumption. Law and Yagel have developed a thrashless raycasting method based on such a memory layout (Law and Yagel 1996). In their approach, all resampled locations within one block are processed before the algorithm continues to process the next block. Knittel (2000) and Mora et al. (2002) achieved impressive performance by using a spread memory layout. The main drawback of such an approach is the enormous memory usage. In both systems, the memory usage is approximately four times the data size.

Splatting

Splatting (Westover 1990) is a technique that traverses and projects footprints, known as splats, onto the image plane (Figure 8.3). In contrast to image-order techniques, object-order methods determine, for each data sample, how it affects the pixels on the image plane. In its simplest form, an object-order algorithm loops through the data samples, projecting each sample onto the image plane. Voxels that have zero opacity, and thus do not contribute to the image, can be skipped. This is one of the greatest advantages of splatting, as it can tremendously reduce the amount of data

that has to be processed. But there are also disadvantages: Using pre-integrated kernels introduces inaccuracies into the compositing process, since the 3D reconstruction kernel is combined as a whole. This can cause color bleeding artifacts, where the color of hidden background objects may bleed into the final image.

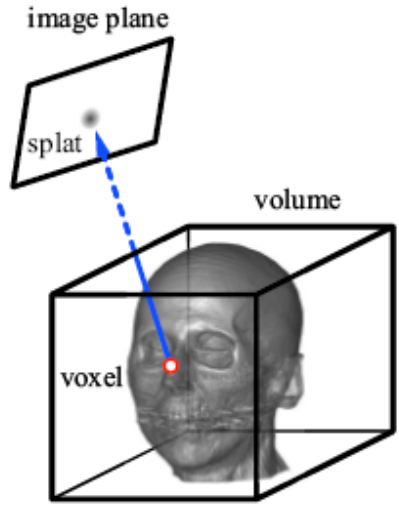

Figure 8.3 Illustration of splatting

To refine these features, an approach has been developed which sums voxel kernels within volume slices most parallel to the image plane. However, this leads to severe brightness variations in interactive viewing. To eliminate these drawbacks Mueller et. al. (Mueller and Crawfis 1998) introduced a special method. Their approach

processes voxel kernels within slabs aligned parallel to the image plane. All voxel kernels that overlap a slab are clipped to the slab and summed into a sheet buffer. Once a sheet buffer has received all contributions, it is composited with the current image, and the slicing slab is advanced forward. Mueller et. al. also presented an acceleration technique called early splat elimination which allows skipping footprint rasterization for occluded voxels (Mueller et al. 1999). However, the projection transformation still has to be performed for these voxels, hence; this optimization is not as effective as early ray termination in raycasting.

Shear-Warp

Shear-warp (Lacroute and Levoy 1994) is such an algorithm, that combine the advantages of both Image-order and Object-order approaches. It is considered to be the fastest software-based volume rendering algorithm. It is based on a factorization of the viewing transformation into a shear and a warp transformation. The shear transformation has the property that all viewing rays are parallel to the principal viewing axis in sheared-object-space. This allows volume and image to be traversed simultaneously. Compositing is performed into an intermediate image. A two-dimensional warp transformation is then applied to the intermediate image, producing the final image. This basic mechanism is illustrated in Figure 8.4. The volume slices are sheared so that all viewing rays are parallel to the major viewing axis. After the projection process has been performed, the distorted intermediate image is warped into the final image.

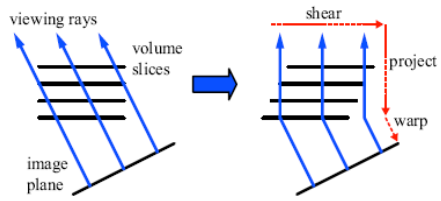

Figure 8.4 Illustration of the shear-warp mechanism

The aligned traversal is the basis for many optimizations: a runlength-encoding of the intermediate image allows an efficient early-ray termination approach. Additionally, runlength-encoding of the volume for each of the three major viewing axis allows skipping of transparent voxels. Additionally, an approach for empty space skipping which is based on a min-max octree has been presented. In contrast to runlength-encoding, this approach allows fast classification and does not require three copies of the volume.

The problem of shear-warp is the low image quality caused by using only bilinear interpolation for reconstruction, a varying sample rate which is dependent on the viewing direction, and the use of pre-classification. Some of these problems have been solved (Sweeney and Mueller 2002), but the image quality is still inferior when compared to other methods such as raycasting.

3D Texture Mapping

As graphics hardware becomes increasingly powerful, researchers have started to utilize the features of

commodity graphics hardware to perform volume rendering. These approaches exploit the increasing processing power and flexibility of the Graphics Processing Unit (GPU). Nowadays, GPU-accelerated solutions are capable of performing volume rendering at interactive frame rates for medium-sized datasets on commodity hardware.

One method to exploit graphics hardware is based on 2D texture mapping (Rezk et al. 2000)(Cabral et al. 1994). This method stores stacks of slices for each major viewing axis in memory as two-dimensional textures. The stack most parallel to the current viewing direction is chosen. These textures are then mapped on object-aligned proxy geometry which is rendered in back-to-front order using alpha blending. This approach corresponds to shear-warp factorization and suffers from the same problems, which is only bilinear interpolation within the slices and varying sampling rates depending on the viewing direction.

The approach usually uses 3D texture mapping and will upload the whole volume to the graphics hardware as a three-dimensional texture (Cabral et al. 1994; Gelder and Kim 1996; Westerman and Ertl 1998; Meibner et al. 1999). The hardware is then used to map this texture onto polygons which are parallel to the viewing plane. These polygons will be rendered in back-to-front order using alpha blending. 3D texture mapping allows the use of trilinear interpolation which is supported by the graphics hardware and provides a consistent sampling rate. A problem of these approaches is the limited amount of video memory. If a dataset does not fit into this memory, it has to be subdivided. These blocks are uploaded and rendered separately, making the bus bandwidth a bottleneck. One

way to overcome this limitation is the use of compression strategy (Guthe et al. 2002).

The increasing programmability of the graphics hardware has enabled several researches to apply acceleration techniques to GPU-based volume rendering (Roettger et al. 2003; Kruger and Westermann 2003). The performance of the approaches is quite heavy and it depends on the hardware implementation of specific features.

CONCLUSION

An extensive comparison of available algorithms for volume rendering has been performed by (Meißner et al. 2000). Although the research is progressing, their basic findings are still valid. They concluded that the raycasting and splatting yield to similar image quality. The render times of these methods are very much dependent on the type of dataset and transfer function. Shear-warp and 3D texture mapping provide high performance, but at the cost of degraded image quality. Recent work has been able to improve the quality of texture mapping approaches (Engel et al. 2001).

REFERENCE

A. LAW AND R. YAGEL. Multi-frame thrashless ray casting with advancing ray-front. In Proceedings of Graphics Interfaces 1996, pages 70–77, 1996.

A. VAN GELDER AND K. KIM. Direct volume rendering with shading via three-dimensional textures. In Proceedings of the Symposium on Volume Visualization 1996, pages 23–30, 1996.

B. T. STANDER AND J. C. HART. A Lipschitz method for accelerated volume rendering. In Proceedings of the Symposium on Volume Visualization 1994, pages 107–114, 1994.

B. MORA, J.-P. JESSEL, AND R. CAUBET. A new object-order ray-casting algorithm. In Proceedings of Visualization 2002, pages 203–210, 2002.

B. CABRAL, N. CAM, AND J. FORAN. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In Proceedings of the Symposium on Volume Visualization 1994, pages 91–98, 1994.

B. CABRAL, N. CAM, AND J. FORAN. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In Proceedings of the 1994 symposium on Volume visualization, p.91-98, October 17-18, 1994.

C. REZK-SALAMA, K. ENGEL, M. BAUER, G. GREINER, AND T. ERTL. Interactive volume rendering on standard PC graphics hardware using multitextures and multi-stage rasterization. In Proceedings of the Workshop on Graphics Hardware 2000, pages 109–118, 2000.

D. COHEN AND Z. SHEFFER. Proximity clouds: An acceleration technique for 3D grid traversal. The Visual Computer, 11(1):27–38, 1994.

G. KNITTEL. The UltraVis system. In Proceedings of the Symposium on Volume Visualization 2000, pages 71–79, 2000.

J.L. FREUND AND K. SLOAN. Accelerated volume rendering using homogenous region encoding. In Proceedings of Visualization 1997, pages 191–196, 1997.

J. SWEENEY AND K. MUELLER. Shear-warp deluxe: the shear-warp algorithm revisited. In Proceedings of the Joint EUROGRAPHICS – IEEE TCVG Symposium on Visualization 2002, pages 95–104, 2002.

J. KR¨UGER AND R. WESTERMANN. Acceleration techniques for GPU-based volume rendering. In Proceedings of Visualization 2003, pages 287–292, 2003.

K. MUELLER AND R. CRAWFIS. Eliminating popping artifacts in sheet bufferbased splatting. In Proceedings of Visualization 1998, pages 239–246, 1998.

K. MUELLER, N. SHAREEF, J. HUANG, AND R. CRAWFIS. High-quality splatting on rectilinear grids with efficient culling of occluded voxels. IEEE Transactions on Visualization and Computer Graphics, 5(2):116–134, 1999.

K. ENGEL, M. KRAUS, AND T. ERTL. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In Proceedings of the Workshop on Graphics Hardware 2001, pages 9–16, 2001.

L. WESTOVER. Footprint evaluation for volume rendering. Computer Graphics, 24(4):367–376, 1990.

M. LEVOY. Display of surfaces from volume data. IEEE Computer Graphics and Applications, 8(3): 29–37, 1988.

M. LEVOY. Volume rendering by adaptive refinement. The Visual Computer, 6(1):2–7, 1990.

M. MEIßNER, U. HOFFMANN, AND W. STRAßER. Enabling classification and shading for 3D texture mapping based volume rendering using OpenGL and extensions. In Proceedings of Visualization 1999, pages 207–214, 1999.

M. MEIßNER, J. HUANG, D. BARTZ, K. MUELLER, AND R. CRAWFIS. A practical evaluation of popular volume rendering algorithms. In Proceedings of the Symposium on Volume Visualization 2000, pages 81–90, 2000.

M. LEVOY. Efficient ray tracing of volume data. ACM Transactions on Graphics, 9(3):245–261, 1990.

P. LACROUTE AND M. LEVOY. Fast volume rendering using a shear-warp factorization of the viewing transformation. Computer Graphics, 28(Annual Conference Series):451–458, 1994.

R. YAGEL AND Z. SHI. Accelerating volume animation by space-leaping. In Proceedings of Visualization 1993, pages 62–69, 1993.

R. WESTERMANN AND T. ERTL. Efficiently using graphics hardware in volume rendering applications. In Proceedings of SIGGRAPH 1998, pages 169–178, 1998.

R. YAGEL AND A. KAUFMAN. Template-based volume viewing. Computer Graphics Forum, 11(3):153–167, 1992.

S. GUTHE, M. WAND, J. GONSER, AND W. STRAßER. Interactive rendering of large volume data sets. In Proceedings of the Visualization 2002, pages 53–60, 2002.

S. ROETTGER, S. GUTHE, D. WEISKOPF, T. ERTL, AND W. STRASSER. Smart hardware-accelerated volume rendering. In Proceedings of the Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualisation 2003, pages 231–238, 2003.

S. PARKER, M. PARKER, Y. LIVNAT, P.-P. SLOAN, C. HANSEN, AND P. SHIRLEY. Interactive ray tracing for volume visualization. IEEE Transactions on Visualization and Computer Graphics, 5(3):238–250, 1999.

T. VAN WALSUM, A. J. S. HIN, J. VERSLOOT, AND F. H. POST. Efficient hybrid rendering of volume data and polygons. In F. H. Post and A. J. S. Hin, editors, Advances in Scientific Visualization, pages 83–96. Springer-Verlag Berlin-Heidelberg, 1992.