Photo-Realistic Image Synthesis in Volume Rendering for Rapid Prototyping and CAM Applications

Alceste Scalas and Piero Pili
Laboratory for Advanced Planning and Simulations
GEMMS Area
Energy and Industrial Processes
CRS4
(Center for Advanced Studies, Research
and Development in Sardinia)

January 2002

Abstract

This document describes some preliminary results of the application of direct volume visualization and realistic image synthesis techniques in Computer Aided Manufactoring (CAM) field.

The final goal is to analyze the use volume data representation and direct volume visualization techniques respectively as basic geometric element and rendering method for numerical control (NC) process planning and simulation. In particular we want to study their application in the process of human organs replication by rapid prototyping techniques. Volume rendering techniques will be used to evaluate the geometric reconstruction of volumetric data (e. g. parts of human body extracted from a CT scan as the carotid arteries), in order to obtain the best polygonal approximation of a dataset. The polygonal model resulting from this evaluation stage will be used to drive its physical reconstruction via rapid prototyping.

A volume rendering program, called *VolCastIA*, is presented. *VolCastIA* is a software enviroment built to evaluate the use of state of the art image synthesis methods in the volume rendering context.

1 Introduction

The Volume Rendering field gained high popularity in the last years. It offers a way to inspect volumetric datasets without the need to preprocess them by geometric sampling techniques (for example, the marching cubes technique) that can cause loss of information and evident reconstruction artifacts.

The use of volumetric datasets representation as central element in rapid prototyping industrial applications is becoming an interesting research problem. And it is also interesting to evaluate the use of volumetric dataset in the industrial milling process field.

1.1 Volume rendering and rapid prototyping

Rapid prototyping tecniques give the possibility to obtain a physical prototype of a polygonal model. They are often used in the industrial field, for example to evaluate the characteristics of a new object, before starting its mass production and its launch on the market.

Furthermore, rapid prototyping can also be used to reconstruct parts of the human body extracted from a CT scan, for example in order to evaluate the possibility to make a prothesis using the patient's dataset as a direct reference model.

Since the rapid prototyping process is actually driven by geometric data, it is important to build a robust system which is able to produce an accurate polygonal mesh from classified or segmented voxels. For this reason the hybrid photo-realistic visualization of polygonal and volumtric data is useful to evaluate and measure the differences between the original dataset and the derived polygonal model to be produced by rapid prototyping device.

1.2 Volume rendering and industrial milling process

1.2.1 Current techniques for milling process simulation and visualization

The current techniques for the visualization and simulation of the industrial milling process are based on the "classical" approach for every three-dimensional visualization problem: using a polygonal model — that, in this field, offers a representation of the object to be machined.

The main problem of this approach is that the milling process is conceptually based on *volume* manipulations (i. e. the removal of parts from a solid block of material), while the polygonal representation is based on *surfaces* (i. e. every object is approximated with a mesh of polygons that represents its outer surface).

This discrepancy causes a series of problem — first of all, during the milling process planning and simulation: the transformations that the milling tool performs on a solid object must be simulated using geometric computations based on planes, rather than on volumes. This is a complicated task, expecially regarding the check of possible collisions between the milling tool (that could be composed by up to 5 articulated axis) and the polygonal structures of the object to be milled. Furthermore, every alteration on the initial polygonal model (caused by the milling process itself) implies the calculation of a new polygonal mesh, with various degrees of approximation and loss of precision.

The same problems can be found during the simple visualization of the milling process. In fact, this visualization must offer a way to understand some fundamental information about the milling phases: for example, the amount and consistency of the residual material, and the regions of the objects in which it is accumulated. This kind of information can be hardly represented using classical polygonal visualization techiques, and the results are not satisfying, both from the aestetic and informative point of view (see figure 1);

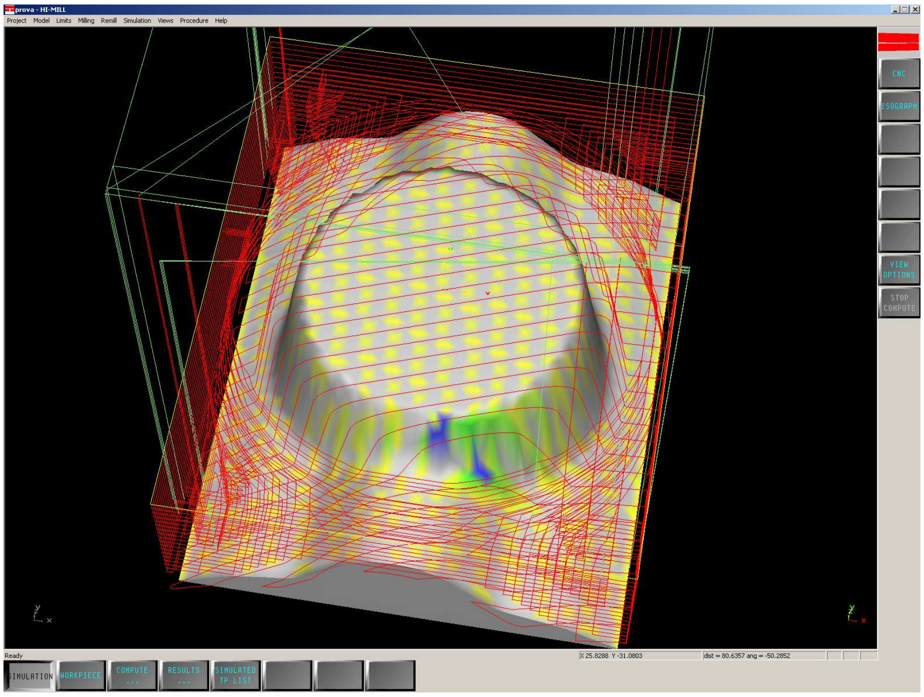

Figure 1: Screenshot from the *Himill* program for milling process planning and simulation, by Fidia S.p.A.. The coloured spots on the machined object represent the volume of residual material. This displacement of the spots does *not* represent the real consistency of the residual material, but it is due to approximations related to the polygonal visualization techniques. More in detail, the volume of the residual material is sampled in some zones of the object, and every sample is rendered with an appropriate colour (based on the volume consistency). There is no way to know (at least intuitively) the amount of residual material in the zones of the object that have not been sampled.

1.2.2 Advantages and disadvantages in volume visualization techniques

The volumetric representation of the object to be machined can solve a number of problems, since the volume manipulations implied in the milling process are directly represented by volumetric information (on which volume rendering operates).

During the milling process planning and simulation, the alteration of the initial volumetric model of the object to be milled does not need approximations or loss of precision: the same resolution is preserved during all the milling phases. There are no complicated polygonal structures that can interact with the milling tool — the machined object is represented with a discrete grid of values, and it makes easier to detect possible paths for the tool and avoid collisions.

Furthermore, during the milling process visualization, volume rendering techniques make it easy to classify different materials (e. g. final object and residual material in the milling process) with different visual properties. There is no need to arbitrarily sample the machined object, in order to detect the residual material information: this information is carried by the volumetric data itself.

On the othe hand, volume manipulation and rendering is a cumputationally intensive task. In the past years, its application fields have been restricted by its hardware requirements: fast and expensive processors (or dedicated hardware), and high amounts of expensive memory were required to perform all the needed computations in a reasonable time. [17] [16] [1] [6] [15]

Today, the low prices of computer hardware (both for processors and memory), and the research progress in volume manipulation alghorithms, make it possible to perform volume manipulations on

And one of the new application fields can be the industrial milling process simulation and visualization.

2 Research goals

The primary goal of this study is the realistic visualization of the volumetric dataset, both for geometrical reconstruction and milling process evaluation. In this context we are interested on high quality rendering of volumetric data and its comparison with the geometrical representation in the format STL obtained by the the geometric reconstruction algorithm.

In this kind of application, volumetric representation and photo realistic image synthesis techniques play a fundamental role: an accurate volume visualization can ease the evaluation of a geometrically-reconstructed dataset to be used for rapid prototyping. Furthermore, during the rapid protototing, and also the milling process, visualization, the operator needs to have a realistic preview of the object under manufactoring in order to evaluate the final results.

However, studying the use of volume visualization in the rapid prototyping and in industrial milling production processes is a more complicated task: photo-realistic image rendering is not usually considered to be a primary goal for volume visualization. In fact its applications, mainly deriveved from the medical field, tend to enhance the dataset understanding, rather than the rendering appearance, or the fidelity to the reconstructe object [14].

Of course the photorealistic volume rendering techniques that will be studied and developed during the research will need, in a second phase, to be compared with the state-of-the-art polygonal geometric representations and rendering techniques currently used in layer-by-layer manufactoring algorithm currently used in rapid prototyping setup and also in milling control simulation programs.

As an advanced objective of this reaseach will be the development of new rapid prototyping and milling process planning strategies, that can be made possible the use of volumetric representation for rapid prototyping process itself.

3 The volume rendering pipeline

Volume visualization encompasses both multidimensional image synthesys and multidimensional image processing. It provides for the analysis and understanding of 3D volumetric scanned or experimental data. The Volume Visualization pipeline takes in input the geometric model and produces as output a 2D shaded projection picture.

In order to identify the volume visualization techniques usable for photo realistic volume rendering application, it is necessary to characterize each stage of the volume visualization pipeline.

An introduction of the pipeline can be found in [7] and [14]. In this section we introduce the Kaufman nomenclature and we give on overview of the main stages of the pipeline. We are interested a photo realistic hybrid volume rendering system i.e. a framework which is able to produce 2D pictures which combine together 3D volumetric data and surfaces of geometric objects.

3.1 Voxelization stage

The first stage of the pipeline is the voxelization of the geometric model (for example the 3D solid to be milled) where we compute the 3D voxel volume (a digital and discrete representation of the object). The primary goal of the voxelization of geometric models (3D scan-converting) is to devise a framework for discrete space that is a close analog of continuous space. The digital representation has to confirm the fidelity and connectivity requirements. For example, a digital surface formed by voxelization has a “thickness” characterized by the absence of tunnels or holes of a certain connectivity.

3.2 Manipulation stage

Once the 3D voxel image has been recontructed (voxelized) or acquired by a scanner apparata as a CT and MRI disposal (in this case the voxelization stage is substitutes by the acquisition, enhancement and reconstruction stages), it is then **3D enhanced** by 3D image-processing techniques (such as median filtering or non linear rank order to reduce the staircase artifact) in the manipulation of volumetric data stage. The manipulation stage also include geometric transformation.

3.3 Classification stage

Then we meet the classification stage. The stage is either a thresholding step or a material classification step. Tresholding is used primarly in the surface-rendering approach, in which a density value of the interface between two materials is selected. All value below it are associated a “0” (transparent, white) value, while all those above are set to “1”. For volume rendering, non binary material classification is performed at the classification stage of the pipeline. In general the problem of classifyng voxel in a dataset that contains multiple material is difficult. Classification for volume rendering may assign such optical properties as light attenuation (e.g. opacity level) and brightness (e.g. color, emission) to each voxel in the dataset. The attenuation and brightness value are assigned on the basis of the voxel value (e.g. density), the percentuage of material prensent in the voxel, the position of the voxel, the knowledge about the material (e.g. bone, tissue, or fat in CT data). Different classifiers require different kind of remdering algorithms and result in different kind of images.

3.4 Mapping stage

At the and of the classification stage, the data reaches the **Mapping** stage. It maps the 3D voxel data into geometric or display primitives. Regardless the origin of the dataset, it can be stored, manipulated, and displayed using a variaty of display primitives. Although volume visualization focuses on 3D primitives, the volume visualizer may capitalize on the advanteges of the more traditional, lower dimensional primitives, to represent and visualize the variour aspects of the same dataset. The choice of the display primitive depends on whether volume rendering, surface rendering or hybrid rendering is used. There are several techniques for extracting the display primitive from the volume dataset. These include: the cuberille model, a surface tracking algorithm that collect the exterior voxel faces, a very fine polygonal mesh generated by the marching cubes algorithm, a cloud of points. There are also many data structures, closely related to the display primitives, that can be used to represent dataset. These include spatial enumeration using voxel matrix, a sparse voxel matrix, or a flat voxel matrix, cell decomposition as in octrees, run-lenht encoding, surfaces of the objects, primitive instancing, constructive solid geometry and boundary models.

3.5 Viewing stage

Once the display primitive are generated they are projected to form a 2D screen image by the volume viewing stage. The projection algorithm depends heavily on the display primitive generated at the mapping stage and on whether volume rendering or surface rendering is used. Conventional viewing algorithm and geometry engines can be exploided to display conventional geometric primitive. However, when volume primitives are displayed directly, a special volume rendering algorithm is used. This algorithm captures the contents of the voxels outside as well as inside the volume being visualized. There are various algorithms used to create a 2D projection from the 3D volumetric dataset using volume rendering. Between them we are interested to forward (iamge-order) projection algorithms, where for each pixel the vopxel contribution for that pixel is gathered from the voxel space. The voxel spatial occupancy can be represented by binary (object/background) function or by a gradued (multivalued, percentuage) function. In the latter category, the multivalued function may be classified as representing a partial coverage, variable densities, and/or graded opacities. Binary classification manifests itself as visual artifact in the displayed image. In contrast, teh graded function approach uses a classification in partial opacity, color, instead of binary classificatio. The resulting coored translucemt gel is then projected by carefully resampling and compositing all the voxels along a projection ray to form the corresponding pixel. This approach generates high quality rendering and it is the focus of this reaserach.

3.6 Shading stage

The final stage of the volume visualization pipeline is the shading stage of the 2D projection. Conventional shading techniques can be used when geometric primitives are the display primitives. Otherwise, a special volume (discrete) shading technique, in which the displayed surface gradient is recovered from the volume data itself, is used. The ultimate goal of the volume shading is to provide a 2D image that is practivcally indistiagshible from a photograph of the real object but retains all its minute details which may be of great scientific or medical value. Unlike shading of geometric objects, where the normals are calculated directly from the geometric information, in volume shading these normals are not available and have to be recovered from the volume data are concerned in this issue.

4 VolCastIA framework

The highest understanding of the visualized volume is here achieved by offering a realistic representation of the volume itself — and here comes the necessity to explore various volume rendering techniques, in order to reach the best photo realistic results. In this section we concentrate our efforts in the follwing stages:

1. classification and mapping stages:
   (a) gradient calculation;
   (b) opacities classification;
   (c) materials classification;
2. viewing and shading stages:
   (a) gradient interpolation;
   (b) opacities interpolation;
   (c) colors/materials interpolation;
   (d) ray projection (parallel, perspective);
   (e) ray processing (interpolation methods, depth cueing, maximum intensity projection, etc.).

Main algorithms which characterize each stage have been implemented in *VolCastIA* in order to evaluate their influence in the generation of final image. The most important algorithms are presented in the next sections.

4.1 Gradient estimation

The gradient calculation method determines the shading appearence (see 3.3 3.6) of the final model (which has got a great influence in the rendering appearence), and, in some cases, it influences the opacities classification phase.

To describing gradient estimation algorithms that we have implemented in *VolCastIA* we use the following notation :

- $f(x, y, z)$ is the voxel value at position (x, y, z);
- $\nabla f(x, y, z)$ is the gradient vector for the voxel (x, y, z);
- $\nabla f(x, y, z)_x$, $\nabla f(x, y, z)_y$ and $\nabla f(x, y, z)_z$ are the components of the gradient for the voxel (x, y, z);
- $\|\nabla f(x, y, z)\|$ is the gradient euclidean norma for the voxel (x, y, z), normalized to be $0 \leq \|\nabla f(x, y, z)\| \leq 1$. Thus, given $\overline{\nabla f}$ the maximum gradient lenght found in a volumetric dataset, we have:

$$\|\nabla f(x, y, z)\| = \frac{\sqrt{\nabla f(x, y, z)_x^2 + \nabla f(x, y, z)_y^2 + \nabla f(x, y, z)_z^2}}{\overline{\nabla f}}$$

Given previous notation, the gradient components may be calculated by:

- Central difference;
- Intermediate difference;
- Sobel estimator.

Next paragrafs discuss each technique.

4.1.1 Central difference

The central difference is one of the most common gradient operators and it is used by Marc Levoy in [11], and it is also implemented in *VolCastIA*.

$$\nabla f(x, y, z)_x = \frac{1}{2}(f(x-1, y, z) - f(x+1, y, z))$$
$$\nabla f(x, y, z)_y = \frac{1}{2}(f(x, y-1, z) - f(x, y+1, z))$$
$$\nabla f(x, y, z)_z = \frac{1}{2}(f(x, y, z-1) - f(x, y, z+1))$$

Hence, its convolution kernel is:

$$\begin{bmatrix} -\frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}$$

Figure 2 reports a rendering that uses this method.

Figure 2: Zoom on a detail of an image rendered with the central difference gradient calculation method.

Figure 3: Zoom on a detail of an image rendered with the intermediate difference gradient calculation method. It is possible to observe that the image is more detailed than figure 2, but it is also less smooth.

4.1.2 Intermediate difference

This is a simple gradient operator, similar to the central difference method. The voxel referenced, where the gradient is calculated, is used in the computation.

$$\begin{aligned}
\nabla f(x,y,z)_x &= f(x,y,z) - f(x+1,y,z) \\
\nabla f(x,y,z)_y &= f(x,y,z) - f(x,y+1,z) \\
\nabla f(x,y,z)_z &= f(x,y,z) - f(x,y,z+1)
\end{aligned}$$

Its convolution kernel is:

$$\begin{bmatrix} -1 & 1 \end{bmatrix}$$

The main advantages of the intermediate difference operator are its simplicity (it requires only a few calculations), and its sensibility (this gradient estimator can perceive very rapid voxel values changes, and so it can show a number of thiny details in the final image). On the other hand, the rendering result is aften very rough, since the gradient vectors can show very frequent (and sometimes unpleasant) changes in their direction and intensity.

Figure 3 reports an image rendered using the intermediate difference konvolution kernel.

Figure 4: Zoom on a detail of an image rendered with the sobel gradient calculation method. This image is more visually pleasant than the ones in figures 2 and 3, at the price of a loss of detail due to the high interpolation.

4.1.3 Sobel estimator

The Sobel operator convolution kernel (for the z direction) is:

$$\begin{bmatrix} -2 & 0 & 2 \\ -3 & 0 & 3 \\ -2 & 0 & 2 \end{bmatrix} \text{(for } x = -1\text{)}$$

$$\begin{bmatrix} -3 & 0 & 3 \\ -6 & 0 & 6 \\ -3 & 0 & 3 \end{bmatrix} \text{(for } x = 0\text{)}$$

$$\begin{bmatrix} -2 & 0 & 2 \\ -3 & 0 & 3 \\ -3 & 0 & 3 \end{bmatrix} \text{(for } x = 1\text{)}$$

In order to calculate the gradient for a given voxel, this operator uses all the 26 voxels in its neighborhood. This determines a very good gradient estimation, but also a heavy load due to the large number of mathematical operations needed to complete the operation, and also an effect of "blurring" on the final image, due to the high level of interpolation on the gradient values.

Figure 4 reports a rendering that uses this method.

4.2 Transfer functions

In order to assign a colour or an opacity to one or more voxel values, it is necessary to define a *transfer function* that, using that value, determines other characteristics (e. g. colour, opacity, reflectance, roughness...).

For this task, actually *VolCastIA* relies on the program user, that creates a configuration file containing the proper associations between isovalues and material/opacities.

An interesting option would be the possibility to help the user in this phase — i. e. giving some sort of automation that, after the analysis of the volumetric dataset, defines one or more transfer functions suitable for viewing the most interesting information of that dataset.

Various examples of this technique, with different approaches, can be found in [4], [5] and [8].

4.3 Opacities classification

The variation of the value opacity in the rendered model is a method to hide unessential information on the final image, or to enhance its important details (see 3.3).

In *VolCastIA* opacities classification stage is computed by using an external data file which store, in tables, the opacity value α_v associated with the isovalue f_v, both defined by the user.

Different opacities classification alghoritms are available in *VolCastIA* to compute the opacity value of each voxel. Next paragraphs describe methods and parameters that determine the quality of the final image.

4.3.1 Linear opacities classification

This method is quite simple, and it has been inspired by Drebin, Carpenter and Hanrahan works [3]. Essentially, every voxel is assigned with the opacity of the isovalue it belongs to. The voxels between two isovalues are assigned with an opacity derived from the linear interpolation of the opacities of the isovalues at bounds — and this interpolation is influenced by a parameter $0 \le p_v \le 1$.

Given two isovalues f_{v_n} and $f_{v_{n+1}}$, with their respective opacities α_{v_n} and $\alpha_{v_{n+1}}$, and given a "transition value" $t_{v_n} = f_{v_n} + p_{v_n}(f_{v_{n+1}} - f_{v_n})$ (which determines the beginning of the linear interpolation between the isovalues), the opacity $\alpha(x, y, z)$ for the voxel at position (x, y, z) is given by:

$$\alpha(x,y,z) = \begin{cases} \alpha_{v_n} & \text{if } f_{v_n} \le f(x,y,z) < t_{v_n} \\ \alpha_{v_{n+1}} \left(\frac{f(x,y,z) - t_{v_n}}{f_{v_{n+1}} - f_{v_n}} \right) + \\ \quad \alpha_{v_n} \left(\frac{f_{v_{n+1}} - f(x,y,z)}{f_{v_{n+1}} - f_{v_n}} \right) & \text{if } t_{v_n} \le f(x,y,z) < f_{v_{n+1}} \\ 0 & \text{otherwise} \end{cases}$$

This classification method is quite direct, and clearly it shows the volume consistencies of the various regions inside a volume dataset. It works well when every voxel in the dataset is adjacent to other voxels belonging to the same isovalue, or to the adjacent ones (i. e. if $f_{v_n} \le f(x,y,z) < t_{v_n}$, then $t_{v_{n-1}} \le f(x \pm 1, y \pm 1, z \pm 1) < f_{v_{n+1}}$). Anyway, its behaviour must be completely driven "by hand" (via the f_v and p_v parameters), and there are no optimizations for the enhancement of certain types of information (apart the volume consistency itself).

An image rendered using this method is shown in figure 5.

The next opacities classification methods offer some ways to extract more detailed structural informations from the volume data.

4.3.2 Isovalues contour surfaces classification

This method has been described by Marc Levoy in [11]. It has been implemented (slightly modified) in *VolCastIA*. Given an isovalue f_v and its related opacity α_v, the opacity for the voxel at position (x, y, z) will be α_v when $f(x, y, z) = f_v$ and $\nabla f(x, y, z) = 0$. For the voxels whose values are different from f_v, the

Figure 5: Image rendered using the linear opacities classification method. It is possible to observe the volume of the tissue (that has been made more transparent than the bone).

Figure 6: Image rendered using the region boundary opacities classification method. The volume of the tissue is almost disappeared, while the sourfaces of the boundary regions between two different materials are enhanced.

opacity will fall off at a rate which is proportional to the difference between f_v and $f(x, y, z)$, and inversely proportional to $\|\nabla f(x, y, z)\|$.

The related expression is:

$$\alpha(x,y,z) = \alpha_v \begin{cases} 1 & \text{if } \|\nabla f(x,y,z)\| = 0 \wedge f(x,y,z) = f_v \\ 1 - \frac{1}{r_v} \left| \frac{f_v - f(x,y,z)}{\|\nabla f(x,y,z)\|} \right| & \begin{array}{c} \text{if } \|\nabla f(x,y,z)\| > 0 \wedge \\ f(x,y,z) - r_v \|\nabla f(x,y,z)\| \leq f_v \leq \\ f(x,y,z) + r_v \|\nabla f(x,y,z)\| \end{array} \\ 0 & \text{otherwise} \end{cases}$$

The r_v parameter represents the thickness of the transition from the f_v isovalue, and influences the rate of the opacity falling off.

When more than one isovalue has to be rendered, the above expression is applied separately to each element, and the results are then combined. Given various isovalues $f_{v_n}, 1 \leq n \leq N, n \in \aleph$, with their respective opacities α_{v_n} and transition thicknesses r_{v_n}, the total opacity for a voxel at position (x, y, z) is given by:

$$\alpha_{\text{tot}}(x,y,z) = 1 - \prod_{n=1}^{N} (1 - \alpha_n(x,y,z))$$

Where $\alpha_n(x, y, z)$ has been calculated with the previous expression, for every isovalue.

This classification method is particularly useful when the rendering is focalized to the structure of one or more omogeneous regions of the volume dataset. The zones in which the voxel values are constant (i. e. $\|\nabla f(x, y, z)\| = 0$) and equal to an isovalue are rendered with their full opacity, while transition regions (i. e. $\|\nabla f(x, y, z)\| > 0$) are rendered with an opacity inversely proportional to the speed of the transition itself. This classification enhances the understanding of some types of dataset, such as the ones derived by electron density maps [11], or, as we will see later, derived from the discretization of a geometric function.

4.3.3 Region boundary sourfaces classification

Marc Levoy described this classification algorithm in [11]. It has been implemented (slightly modified) in *VolCastIA*. This method has been developed for the rendering of human body CT datasets.

Human tissues are quite homogeneous, and, for simplicity, it is possible to assume that, in a given volume, every one of them touches at most two other types, which are expressed in the CT dataset with different values. Given this assumption, it is possible to implement an opacities classification method that, instead of the regions of a dataset belonging to one or more isovalues, enhances the surfaces at the boundary of the isovalues themselves. It offers to physicians a better understanding of a CT dataset, since they are often more interested in the tissues boundary regions, rather than in the internal tissues (that could be an obstacle for a clear visualization).

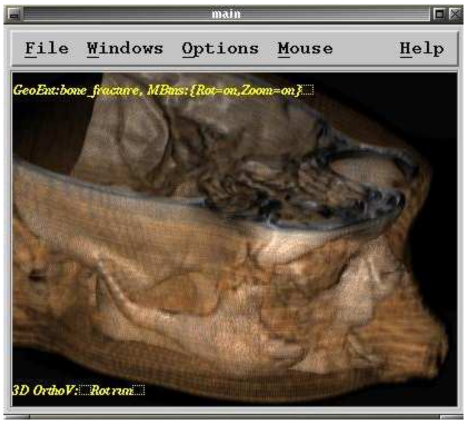

Figure 7: Rendering of a CT scan of a patient with a cheekbone fracture.

Figure 8: Rendering of a CT scan of the head of a cadaver. The beams projecting from the mouth are an artifact in the dataset, and are due to dental fillings scattering X-rays.

In order to remove internal tissue classification, it is possible to make the voxels opacities $\alpha(x,y,z)$ linearly proportional to the gradient magnitude $\|\nabla f(x,y,z)\|$. This way, the internal tissue (where $\|\nabla f(x,y,z)\| \rightarrow 0$) are rendered as (almost) completely transparent.

Given two isovalues f_{v_n} and $f_{v_{n+1}}$, with their respective opacities α_{v_n} and $\alpha_{v_{n+1}}$, the opacity for a voxel at position (x,y,z) is given by:

$$\alpha(x,y,z) = \frac{\|\nabla f(x,y,z)\|}{h} \begin{cases} \alpha_{v_{n+1}} \left(\frac{f(x,y,z)-f_{v_n}}{f_{v_{n+1}}-f_{v_n}}\right) + \\ \quad \alpha_{v_n} \left(\frac{f_{v_{n+1}}-f(x,y,z)}{f_{v_{n+1}}-f_{v_n}}\right) & \text{if } f_{v_n} \leq f(x,y,z) \leq f_{v_{n+1}} \\ \\ 0 & \text{otherwise} \end{cases}$$

In this expression, $h \in \Re^+$ is a parameter used to make the $\frac{\|\nabla f(x,y,z)\|}{h}$ fraction tend more rapidly to ∞ (or 0), thus modifying the gradient magnitude influence on the algorithm "expanding" and highlighting (or "shrinking" and attenuating) the surfaces detected at the isovalues boundary regions. After applying the above expression, the values of $\alpha(x,y,z)$ greater than 1 will be set to 1.

In figures 7 and 8 it is possible to observe the renderings of two medical datasets, using the region boundary sourfaces classification. Furthermore, in figure 6 is shown the same dataset of figure 5, with different opacities classification method.

4.4 Materials classification

In this phase of the volume rendering pipeline every voxel value is assigned with a percentuage of one or more materials, that will be used to determine the final colour of the voxel when it will be used for the sampling of view rays.

Currently, only one materials classification method is implemented in *VolCastIA*

Linear materials classification

Like the linear opacities classification method seen above, this algorithm has been inspired by Drebin, Carpenter and Hanrahan'sworks [3] works, and basically works the same way (it is incluenced by the same p_v parameter).

Given two isovalues f_{v_n} and $f_{v_{n+1}}$, with their assigned materials m_{v_n} and $m_{v_{n+1}}$ (with their colours c_{v_n} and $c_{v_{n+1}}$), and given a "transition value" $t_{v_n} = f_{v_n} + p_{v_n}(f_{v_{n+1}} - f_{v_n})$ (the same seen before), the colour $c(x, y, z)$ for the voxel at position (x, y, z) is given by:

$$c(x,y,z) = \begin{cases} c_{v_n} & \text{if } f_{v_n} \le f(x,y,z) < t_{v_n} \\ c_{v_{n+1}} \left(\frac{f(x,y,z) - t_{v_n}}{f_{v_{n+1}} - f_{v_n}} \right) + \\ \quad c_{v_n} \left(\frac{f_{v_{n+1}} - f(x,y,z)}{f_{v_{n+1}} - f_{v_n}} \right) & \text{if } t_{v_n} \le f(x,y,z) < f_{v_{n+1}} \\ 0 & \text{otherwise} \end{cases}$$

4.5 Ray processing

Various kind of processing of the data acquired during the ray volume traversal offer the possibility to obtain different results on the final image.

4.6 Depth cueing

The human perception of depth in three-dimensional scenes is mostly based on the visual focus — that enhances the foreground objects contours, while blurring the background ones.

A similar help to the understanding of the depth of a three-dimensional dataset (and, more in general, of every computer-synthesized three-dimensional object) can be achieved by applying the depth-cueing effect: that means, making the background objects darker than the foreground ones, simulating the presence of a uniform light-absorbing fog.

4.7 Interpolation methods

Various interpolation methods have been implemented in order to assign a value to every ray sample, based on the values of the adjacent voxels. These methods are explained in the "Interpolation methods" section.

4.8 Maximum Intensity Projection

This volume rendering technique is based on visualizing only the sample with the greater value found by the ray during volume traversal [10]. Even if this technique is not strictly related to the realistic representation of a volume, it could be useful to give more information about the rendered dataset.

4.9 Sum projection

This method is based on the visualization of the sum of all the values sampled during ray traversal, with a visual effect similar to the one given by a X-ray image.

4.10 Interpolation methods

Every sample of every ray shot in the volumetric dataset has to be computed by interpolating various information (i. e. opacities and materials/colours) carried by the adjacent voxels. This interpolation may be performed using various techniques, that usually give a more realistic result at the price of an increased computational complexity.

4.11 Parallel and perspective rendering

The difference between parallel and perspective volume rendering is usually a matter of performance vs. realism: perspective volume rendering offers the maximum image understanding, while the parallel rendering, at the price of a slightly perceptible loss of proportion in the represented object, offerse various methods to enhance the rendering speed, by using data access patterns that exploit cache coherency [13].

In *VolCastIA*, no parallel rendering optimization alghoritms have been implemented — and the possibility to obtain parallel rendering has been implemented in order to evaluate the perceptible differences between these two methods.

4.12 Colour spaces

In order to reach the maximum realism in volume visualization, the *VolCastIA* rendering pipeline has been extended with the possibility to compute the final image colours using various color spaces: common RGB, CIE-xyz, CIE-luv, and direct color spectrum analysis and interpolation. The differences in terms of rendering realism and speed have been evaluated, in order to obtain the maximum image fidelity (obtained by using a colourspace that follows the human visual sistem response to colours variations [2]), but also to choose an optimal trade-off between these two aspects.

4.13 Hybrid rendering of volumetric and geometric data

The study of volume rendering CAD/CAM and rapid prototyping applications requires to evaluate the differences with geometric rendering, both in terms of visual appearence, and objects representation precision.

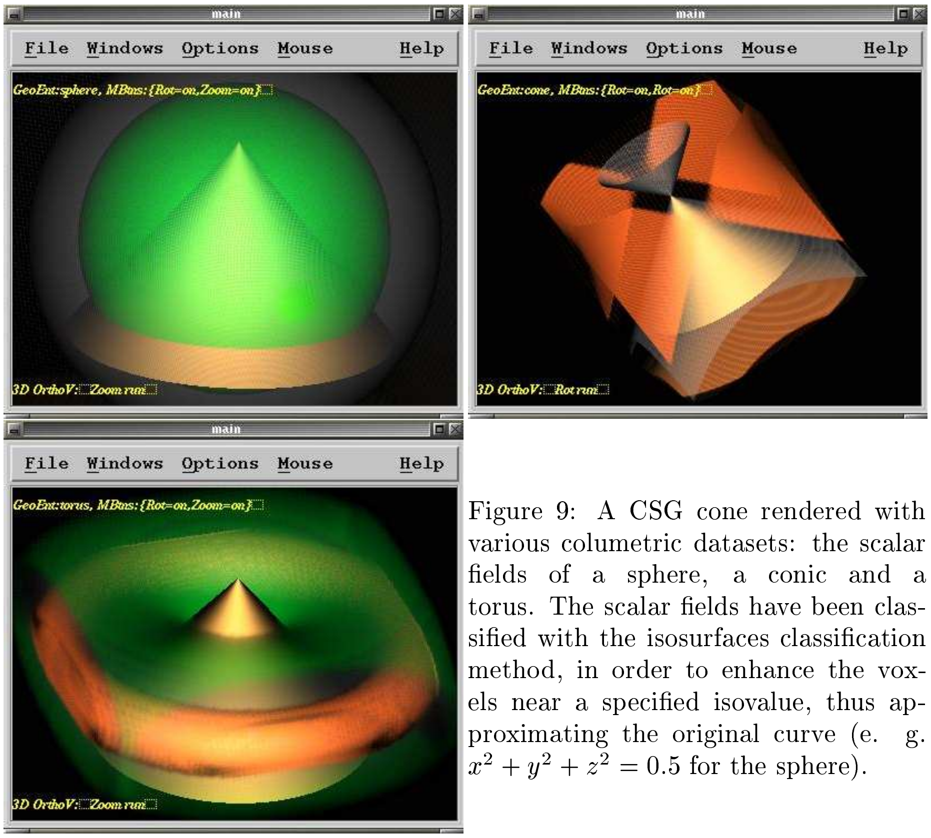

Figure 9: A CSG cone rendered with various columetric datasets: the scalar fields of a sphere, a conic and a torus. The scalar fields have been classified with the isosurfaces classification method, in order to enhance the voxels near a specified isovalue, thus approximating the original curve (e. g. $x^2 + y^2 + z^2 = 0.5$ for the sphere).

In this phase, it is necessary to visualize geometric and volumetric data, in order to examine the possibility to represent both in a CAD/CAM or rapid prototyping planning program (since geometric objects could be useful to enhance the graphical feedback), and to evaluate the rendering precision (e. g. measuring the differences between a geometrically generated sphere and its voxel representation) (figure 10).

For this pourpose, it has been implemented a method inspired by [12], adding to *VolCastIA* some of the characteristics of a CSG program developed internally by CRS4 (*RaycastCSG*). In figure 9 it is possible to observe the rendering of a CSG cone rendered with a volumetric dataset, and in figure 10 is shown a detail of the cone vertex.

References

[1] Arie E. Kaufman — Reuven Bakalash. Memory and processing architecture for 3d voxel-based imagery. *IEEE Computer Graphics and Applications*, 8(11), November 1988.

[2] Andrew S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufman, 1995. ISBN 1-55860-276-3.

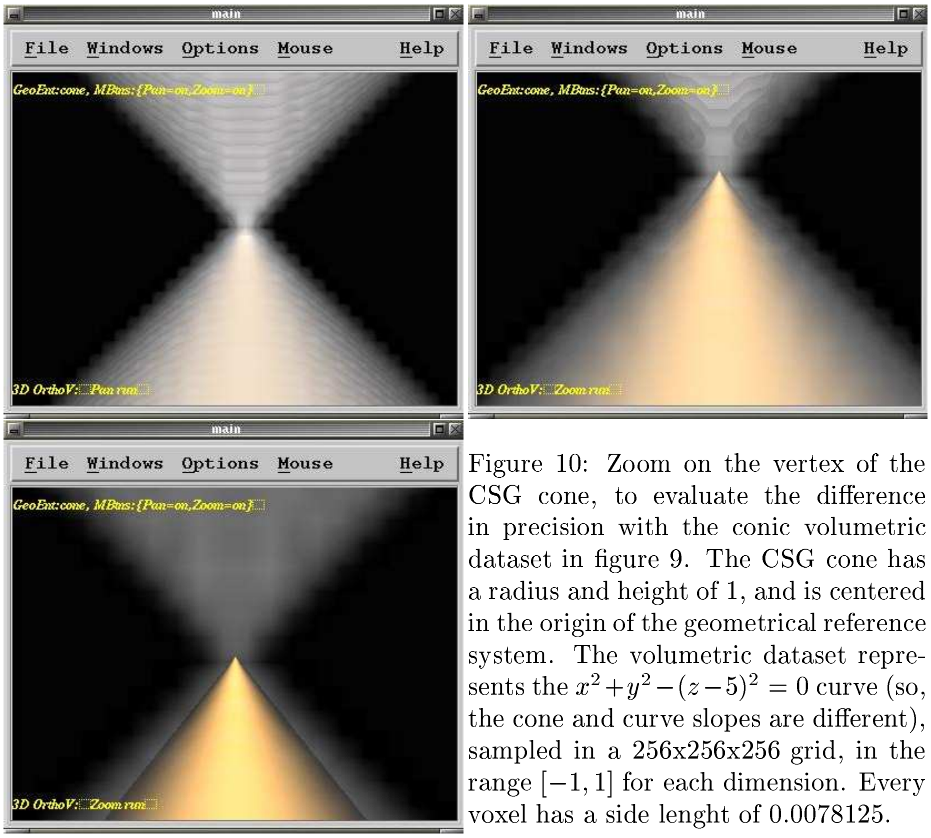

Figure 10: Zoom on the vertex of the CSG cone, to evaluate the difference in precision with the conic volumetric dataset in figure 9. The CSG cone has a radius and height of 1, and is centered in the origin of the geometrical reference system. The volumetric dataset represents the $x^2+y^2-(z-5)^2 = 0$ curve (so, the cone and curve slopes are different), sampled in a 256x256x256 grid, in the range $[-1, 1]$ for each dimension. Every voxel has a side lenght of 0.0078125.

[3] R. Drebin — L. Carpenter — P. Hanrahan. Volume rendering. *Computer Graphics (ACM Siggraph Proceedings)*, 22(4):65–74, 1988.

[4] Taosong He, Lichan Hong, Arie Kaufman, and Hanspeter Pfister. Generation of transfer functions with stochastic search techniques. In Roni Yagel and Gregory M. Nielson, editors, *IEEE Visualization '96*, pages 227–234, 1996.

[5] Jiří Hladůvka, Andreas König, and Eduard Gröller. Curvature-based transfer functions for direct volume rendering. In Bianca Falcidieno, editor, *Spring Conference on Computer Graphics 2000 (SCCG 2000)*, volume 16, pages 58–65, may 2000.

[6] D. Jackel. Reconstructing solids from tomographic scans: the parcum ii system. *Advances in Computer Graphics Hardware II*, 1988.

[7] A. Kaufman. Volume visualization. In *Volume Visualization*. IEEE Computer Society Press Tutorial, 1990.

[8] Gordon Kindlmann and James W. Durkin. Semi-automatic generation of transfer functions for direct volume rendering. In *IEEE Symposium on Volume Visualization*, pages 79–86, 1998.

[9] Gunter Knittel. The ultravis system. *Hewlett-Packard Laboratories, Visual Computing Department*, 2000.

[10] L. Mroz — E. Gröller — A. König. Real-time maximum intensity projection. In E. Gröller, H. Löffelmann, and W. Ribarsky, editors, *Data Visualization '99*, pages 135–144. Springer-Verlag Wien, 1999.

[11] M. Levoy. Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, pages 29–37, May 1988.

[12] M. Levoy. A hybrid ray-tracer for rendering polygon and volume data. *IEEE Computer Graphics and Applications*, 10(2):33–40, March 1990.

[13] P. Lacroute — M. Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. *Proceedings of SIGGRAPH '94*, pages 451–458, July 1994.

[14] Berthold Lichtenbelt — Randy Crane — Shaz Naqvi. *Introduction to Volume Rendering*. Hewlett Packard - Prentice Hall PTR, 1998. ISBN 0-13-861683-3.

[15] Claudio T. Silva — Arie E. Kaufman — Constantine Pavlakos. Pvr: High-performance volume rendering. *IEEE Computatonal Science & Engineering*, 3(4), Winter 1996.

[16] Toshiaki Ohashi — Tetsuya Uchiki — Mario Tokoro. A three-dimensional shaded display method for voxel-based representation. *Proceedings of EUROGRAPHICS 1985*, September 1985.

[17] Samuel M. Goldwasser — R. Anthony Reynolds — Ted Bapty — David Baraff — John Summers — David A. Talton — Ed Walsh. Physician's workstation with real-time performance. *IEEE Computer Graphics and Applications*, 5(12), December 1985.