Volume Rendering in Architecture

Overlapping and combining 3d voxel volume data with 3d building models

Christian Tonn¹, René Tatarin²
Bauhaus-University Weimar, Germany
¹http://infar.architektur.uni-weimar.de,
²http://www.uni-weimar.de/cms/bauing/organisation/fib.html
¹christian.tonn@uni-weimar.de, ²rene.tatarin@uni-weimar.de

Abstract. Volume rendering is an illustration technique for visualising different 3D measured data or 3D simulation data interactively on screen. This paper introduces a method that overlays several types of volume data on an architectural surface model. This complex calculation takes place on the graphics card using hardware-accelerated shaders. An implemented software prototype entitled “VolumeRendering” is introduced. In addition to interactive visualisation, the objective was to create a user-friendly interface. Synergies and new evaluation possibilities arise through the overlay, e.g. of different measuring techniques, with a surface model. Finally the use of the software prototype is illustrated using examples from our interdisciplinary research project.

Keywords. Multiple Volume Rendering; Overlay; 3D Surface Models.

INTRODUCTION

The work presented in this paper is the product of an interdisciplinary research project entitled “nuBau” (methods and materials for user-oriented building renovations) involving researchers from the fields of building surveying and diagnostics and building physics as well as material scientists and architectural computer scientists. Our aim as computer scientists is to incorporate the different data about a building into a dynamic digital building model so that this is available to everyone in the architectural planning process. One aspect of this building model is volume data. “Volume data” refers to a three-dimensional cubic volume that is sub-dividable into regular “volume elements” or “voxels”. Each voxel can contain data describing its characteristics for simulation purposes (e.g. temperature, air flow velocity and illumination levels) and for non-destructive material testing techniques (georadar, ultrasound tomography). These different characteristics need to be superimposed so that they can be used in combination, visualised and assessed in the context of a three-dimensional building model.

Volume rendering is already well established in the field of computer graphics as a means of visualising large sets of volume data (Levoy, 1988) and is already widely used in the field of medicine (e.g. MRI, CT) to render and examine human tissue. In the field of building diagnostics, volume rendering is likewise already used for individual, separate measuring techniques such as ultrasound tomography. What’s new about the technique discussed in this paper is the ability to visualise combinations of different kinds of volume data and the ability to examine them interactively in a digital 3D building model as well as in conjunction with other visual measuring techniques such as distortion-corrected ther-
mography. A standalone software prototype named “VolumeRendering” has been developed which we present in this paper.

**CLASSIFICATION**

Volume rendering is an ongoing area of research in the field of computer graphics. Though the foundations were already laid during the 1980s, the techniques described then have only recently become possible to realize using normal computers. By using hardware-accelerated algorithms on modern graphics cards (general purpose computation on graphics processing unit GPGPU), volume rendering has become possible in “real time”.

Volume rendering is a widely used technology – for further information see “An Overview of Volume Rendering” by A. Kaufman and K. Müller, 2005. In this paper, we don’t introduce a specific or completely new technique but rather combine different concepts from computer graphics to customise them for use in visualising an architectural surface model.

**DATA FORMATS**

The absence of a standard file format for volume rendering might be explained by the niche character of the market and the relatively new technology of volume rendering. There are, in fact, many volume data file formats. Measuring instrument manufacturers, for example, generally define their own proprietary volume data formats according to their own respective requirements. For the software prototype it was therefore necessary to implement several different importers or convertors that can read in different types of voxel volume data as well as irregular point cloud data. The import function also maps local control points in the imported volume data set to global control points in the building model so that imported volume data can be displayed in the correct position, orientation and scale in the building model.

**VOLUME RAY CASTING**

The basic approach used to visualize the volume data sets is the ray casting approach as described by Watt et al., 1992. For each pixel in a view, a ray is cast from the viewpoint of the viewer and passed through the volume data set (see Figure 1). In the software prototype, this is done using a hardware-accelerated GLSL shader on the graphics card. Depending on the shading method used, the volume data values along the ray may be accumulated or alternatively the maximum is shown (see Figure 2).

![Figure 1](image1.png)
*For every pixel a single ray is sent through the volume data.*

![Figure 2](image2.png)
*Along every ray, measured data is taken from the volume, which computes the pixel colour.*

Seven different shader methods are implemented in the software prototype, each employing different parameters to show volume data in its own way:

1. The “Accumulation shader” adds up all measured data from the ray and applies a transfer function afterwards to display the pixel colour (Figure 3).
2. The “Maximum shader” examines all measured data from the ray and displays the maximum found by means of a transfer function (Figure 4).
3. The “Boundary shader” traces the ray until its value reaches a value higher than a user-defined threshold. The exact position of the threshold is then interpolated by means of the last lower measured value. The pixel is then illuminated with the help of the position, the precomputed volume-normal at this point and the lighting settings (Figure 5).

4. The “Transparent boundary shader” works very much like the boundary shader but does not abort after the first threshold it finds, instead searching for further thresholds on its ray. The lighting results are then added up for display (Figure 6).

5. The “Cross-section shader” can display the volume with an adjustable X, Y and Z cross-section. A transfer function is used to convert the measured data into pixel colours (Figure 7).

6. The “Transparent-opaque boundary shader” combines the two boundary shaders (3 and 4 above) into a single shader. Different thresholds can be set for the opaque and the transparent boundary (Figure 8).

7. The “Cloud shader” adds up all measured data of the ray and afterwards applies a lighting function as well as a transfer function to display the pixel value (Figure 9).
MULTIPLE VOLUME RENDERING WITH SURFACE MODELS

These seven different shader methods can be classified into two groups: shaders in which the rays fully penetrate the volume (1, 2, 4 and 7) and shaders in which the rays end at a point inside the volume (3, 5 and 6). This differentiation is important to consider for the algorithm when several volume data sets are to be overlaid over one another. When computing volumetric visualization on the graphic card, it is important to optimize memory access patterns and compact address blocks to maximize memory caching. For this reason the contributions of each volume data set to the resulting ray per volume is computed individually and then subsequently combined. In a first step, the system checks whether the volume data can contain boundaries using the specified shader method. Out of all the “bounded” volume data sets, the boundary nearest to the viewer is computed for the ray concerned. The “un-bounded” volume data sets to this nearest boundary point are then computed and used to finally calculate the colour value of the pixel with the help of an optional transfer function.

The combination of the volume data sets with the three-dimensional surface model of the architecture employs the same method. One need only inform the graphic-card-based ray-casting algorithm of the position of a boundary, for example of the building, from the given perspective (Kreeger et al. 1999). From an off-screen visualisation of the volume, a depth buffer (z-buffer) is computed for the given camera perspective and passed as a parameter to the GLSL shader. As a result for every pixel one can quickly determine whether the ray intersects with the front-most surface of the three-dimensional building model and therefore whether the ray casting process can be aborted. Using this approach it is possible to overlay volume data sets over an arbitrary three-dimensional surface model.

All this remains hidden to the user because it automatically runs in the background of the software prototype.

METHODS OF DIFFERENTIATING BETWEEN VOLUME DATA SETS

An important aspect of volume renderings is the ease with which the user can differentiate between the different sets of volume data in the model, in particular when several sets of volume data are overlaid in one model. Out of the mass of information, only what is relevant for the user needs to be abstracted. A variety of different methods are implemented in the software prototype, of which the following are perhaps most commonly used:

- The combination of different shader methods,
- The user-definable clipping of volume data sets, or of the surface model,
- The use of colour or transparency transfer functions,
- The direct combination of different volume values with the help of value ranges that serves as a condition for the display of other volume data,
- Different thresholds and colours for the boundary shader.

USER INTERFACE

Besides the menu, an icon toolbar and a status bar, the user interface of the “VolumeRendering” software prototype is made up of a 3D perspective window and a control area at the right edge of the application. This control area is dynamically changeable. It can display some global as well as several grouped parameters for each volume data record. At the top of the control area, a tree structure widget functions like a directory folder. Depending on which “param-

722 | eCAADe 30 - Volume 1 - Design Tool Development
eter group" one selects, the user interface changes accordingly in the lower control area. Figure 10 shows all the available parameter group panels. The parameter groups serve the following purposes:

- **“Volume Rendering”**: The step width of the ray-casting algorithm is set here globally.
- **“Model Clipping”**: The clipping of the 3D surface model is set here.
- **“Shader x Dataname”**: Global settings for this shader type are defined here, for example the shader type or the adjustment of the value range.
- **“Min Max”**: The loaded volume data record can be clipped here.
- **“Surface”**: Material properties and thresholds for the boundary shaders.
- **“Light”**: The position of the light source.
- **“Cuts”**: The position of the cross-sections.
- **“Location”**: The position, scaling and rotation of the volume data record.
- **“Colour”**: Different colour and transparent transfer functions can be defined here. In addition the colour legend is displayed under the histogram.
- **“Condition”**: If multiple volume data records are used, value ranges may be used here as a condition for rendering the other.

**AREAS OF APPLICATION**

In this section we present a brief overview of some application areas of the work resulting from our research project.

**Physical flow simulation**

Figures 11 and 12 show examples of flow simulation. In the example, the building physics planner defined a room including air intake and outlet openings. A person sitting in the centre of the room acts as a thermally active object. The volume data set contains the results of a simulation showing the tem-
A 3D surface model of the interior of the building element (the concrete test specimen) was then created and overlaid with the measured volume data. In Figures 15 and 16 the overlay of the interior model with the georadar volume data is displayed. The view of the rebar reinforcement on the left area shows that there is an error in the volume data. The temperature and air flow velocity in the room. Models like this can be used to evaluate thermal comfort levels in a room under different conditions (Voelker et al., 2011).

**Non-destructive material measuring techniques**

Figures 13 and 14 show a further example from the field of building diagnostics. Non-destructive measuring techniques (ultrasound and georadar measurements) are used to examine particular building elements. To assess the accuracy of the method, a concrete test specimen was created with a known interior structure (Figure 14). The test specimen was then measured using two different non-destructive measuring instruments (an ultrasound tomography device and a georadar scanner). Figure 13 shows the measured volume data: the coloured values show the ultrasound tomography while the white rods stem from the georadar scanner.
measuring wheel, which determines the position of the measured data on the concrete face, probably slipped during the measurement process. This is an error that can often occur in non-destructive georadar investigations.

Figures 17 and 18 show the interior model overlaid with the ultrasonic volume data. Here one sees the upper pipes quite clearly. This method offers a means of providing “as-built” documentation for building elements that are concealed within walls.

**Light simulation**

Using the Bauhaus “Musterhaus am Horn” in Weimar as a test case, a simulation of light levels has been undertaken using the software prototype “Colored Architecture” (Tonn et al., 2006). Figures 19 and 20 show an example, in which the level of illumination for every point in the building has been calculated. The results are shown superimposed over the building model. The two images show an examination of the interior of the building. For this an option was incorporated that allows one to produce a cutaway view at a defined point by clipping the 3D surface model.
CONCLUSION
In this paper we show that volume rendering represents an important aspect of the visualisation of digital building models. The examples outlined here show the breadth of potential application areas and illustrate how the results of measurements and simulations by specialist planners can be presented and assessed using a digital building model. The ability to combine and present different volumetric, image and surface model data in a superimposed view is particularly useful. As suitably powerful graphic cards with volume rendering capabilities become more widespread along with corresponding software tools, the visualisation of volume data will become an increasing relevant area for building planning.

REFERENCES