ClearView: An Interactive Context Preserving Hotspot Visualization Technique

Jens Krüger, Jens Schneider, and Rüdiger Westermann

Abstract—Volume rendered imagery often includes a barrage of 3D information like shape, appearance and topology of complex structures, and it thus quickly overwhelms the user. In particular, when focusing on a specific region a user cannot observe the relationship between various structures unless he has a mental picture of the entire data. In this paper we present *ClearView*, a GPU-based, interactive framework for texture-based volume ray-casting that allows users which do not have the visualization skills for this mental exercise to quickly obtain a picture of the data in a very intuitive and user-friendly way. *ClearView* is designed to enable the user to focus on particular areas in the data while preserving context information without visual clutter. *ClearView* does not require additional feature volumes as it derives any features in the data from image information only. A simple point-and-click interface enables the user to interactively highlight structures in the data. *ClearView* provides an easy to use interface to complex volumetric data as it only uses transparency in combination with a few specific shaders to convey focus and context information.

Index Terms—Focus & Context, GPU rendering, volume raycasting.

1 Introduction

The necessity to provide additional context information when communicating the exact shape and position of inner organs was already discovered centuries ago by artists aiming for intuitive anatomic sketches. At that time anatomic drawings were entirely new. Without additional visual cues people would not have recognized what they were looking at. Over the centuries the paradigm of presenting both a detailed region and a surrounding context has not changed much. We still gain most information about unknown data not by seeing all information at once, but by looking at a carefully filtered fraction of the data set, usually referred to as the focus, embedded into some other aspect of the data, mostly conveying positional cues, called the context. Among the plethora of examples demonstrating the usefulness of man-made focus+context approaches are the anatomic sketches of DaVinci (see Figure 1) as well as the technical illustrations of our time (Figure 3).

The major insight behind all these examples has always been the same: Abstraction is the key to condense the information in the data set to a level that allows quick and intuitive understanding. Paradoxically, it turns out that the condensed result often reveals more information than a whole view of the data.

Today we face similar challenges as artists 500 years ago. Still we are trying to understand complex shapes and to visually communicate relevant features to help viewers relating these features to the entire data. This problem is further aggravated by the fact that the amount of information available today has sheerly exploded during the last decades. At the same time it has become possible to interactively visualize large and highly detailed volumetric data sets. However, volume rendered imagery often includes a barrage of 3D information like shape and appearance of complex structures, and it thus quickly overwhelms the viewer. In particular, when focusing on a specific region a viewer cannot understand the relationship between various structures, unless he has a mental picture of the entire data set by navigating around and observing the component parts. Many users of interactive volume rendering software do not have the required visualization skills for this mental exercise, and many others are not willing to invest the time and effort. The result is that a large portion of users have difficulties in understanding non-trivial data sets and in finding what they are looking for in those data sets. Consequently there is a dire need for novel techniques that can support users in this task which are intuitive and simple enough to be accepted in practice.

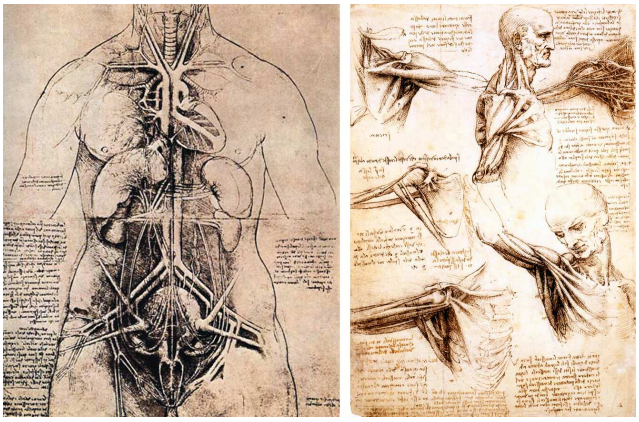

Fig. 1. Already Leonardo DaVinci [7] followed the focus+context paradigm to convey relative positions of inner organs and muscles in the human body.

In this paper we present *ClearView*, an interactive and intuitive volume visualization tool that provides the user with a simple exploration metaphor. Following traditional technical illustrations, several 3D layers of volumetric data sets are extracted using texture-based ray-casting. They are composed to produce high-quality images at high frame rates. The user guides the exploration process by moving the hotspot, a lens-like yet distortion free region, in which additional layers of the data set are augmented to convey relevant features. The proposed GPU system exploits feature-based techniques to improve the understanding of complex 3D data sets, and it utilizes image-based deferred shading to maximize performance. *ClearView* does not require any additional feature volumes, making the method suitable for the interactive rendering of high-resolution data sets on recent GPUs. Several shaders to intuitively convey material and shape properties are integrated into the system. To keep the user interface slim, only few parameters abstract from the technical realization. The user simply positions the hotspot on the data set and selects an amount of transparency to continuously blend between focus and context information.

- *Jens Krüger, E-mail: jens.krueger@in.tum.de.*
- *Jens Schneider, E-mail: schneidj@in.tum.de.*
- *Rüdiger Westermann , E-mail: westermann@in.tum.de. all authors are with Computer Graphics & Visualization Group, Technische Universität München*

Manuscript received 31 March 2006; accepted 1 August 2006; posted online 6 November 2006.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

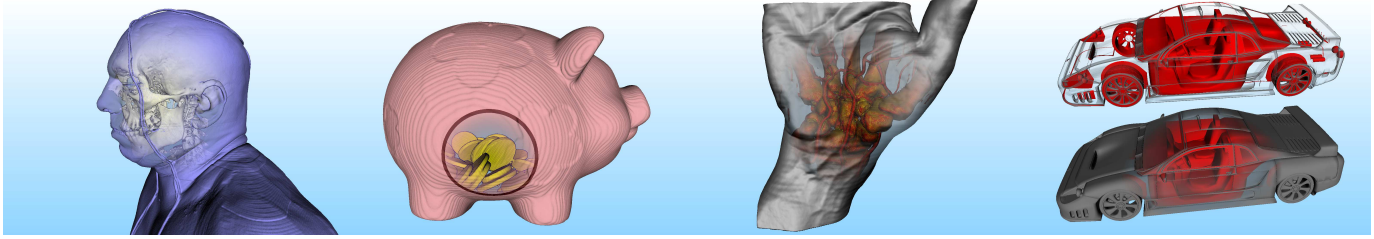

Fig. 2. The *ClearView* system can be used to visually explore complex data sets at interactive frame rates by using a focus+context metaphor. Even the leftmost image of the 512^3 visible human data set with multiple iso-surfaces was generated at about 15 fps on a 800x600 viewport.

2 Related Work

Psychological studies indicate that humans process information perceived as a single visual event intuitively, while the cognitive understanding of distinct events, such as multiple images, requires much greater effort [11, 26]. The focus+context paradigm seeks thus to combine multiple aspects of the data into a single visual event by assigning portions of one image to different aspects of the data. The general idea to emphasize certain aspects of the data in an intuitive way is also the basis of many non-photorealistic rendering (NPR) techniques [13]. Consequently, NPR has been applied to both volume rendering [25, 16, 17, 10, 28, 24, 31] and focus+context techniques in visualization [32]. In the following we will shortly summarize previous approaches.

2.1 Distortion Lenses

Early work on automated focus+context originated from the need to visualize vast amounts of information using the limited space of 2D screens. Depending on the type of data to be displayed, different methods have been proposed. Shaw et al. [30] used a lens metaphor to filter and visualize scattered, high-dimensional information. For multiple 2D Layers, Bier et al. [1] suggested the *Toolglass* interface also supporting lenses. For single-layered 2D data, several techniques seeking to assign more screen space to important regions were suggested [22]. The generalization to handle 3D data requires some effort in order to ensure an unobstructed line of sight to the focus [4]. For volume rendering however, even if the focus is clearly visible, non-linear distortions in combination with fuzzy, semi-transparent structures are potentially counter-intuitive. Nevertheless, distortion lenses have been applied to 3D volume rendering successfully [20, 15, 34, 6]. Some unresolved issues remain, though. For instance, it is not yet clear if a transition region [34] around the lens helps the user to understand the data set, since it deforms the context [15]. In medical applications the distortions associated with lenses might only be acceptable in selected cases.

2.2 Cutaway Illustrations

In cutaway illustrations a selective view on important details in the interior of an object is provided by omitting extraneous details. Conceptually these illustrations cut away parts of the outer hull using simple geometries such as clip planes or simple, convex objects. Similar to 3D lenses, special care has to be taken in order to provide an unobstructed view through the cut onto the inner structures, making image based methods appealing. For polygonal models, the GPU-based methods by Diepstraten et al. [9, 8] are promising, but were not generalized to volume data. Weiskopf et al. [36] performed GPU-accelerated clipping for volumes with arbitrary clip geometry. McGruffin et al. [27] proposed an interactive system to browse pre-classified iso-surfaces in volume data by deforming them according to simple and intuitive metaphors.

2.3 Multiresolution focus+context

Naturally the user is most interested in the focus region, while the context is needed only to provide positional cures. Consequently, various multi-resolution techniques have been proposed not only gain a speedup by reducing the resolution in the context region, but also to point out a particular feature in still images. Levoy et al. [23] coupled an early eye tracking device with a volume renderer to select the focus. Cignoni et al. [5] used a 3D *MagicSphere* to define the focus for triangle models and to guide appropriate remeshing. For the focus+context reconstruction of iso-surface, a similar concept was later used by Westermann et al. [37]. Lee et al. proposed to find an optimum view based on saliency [21]. Weiler et al. [35] implicitly used a focus+context technique by gradually decreasing the resolution of volumetric data with increasing distance to the camera. Lately, Ropinski et al. [29] applied focus+context techniques to select shaders for seismic data sets.

2.4 Context-Preserving Volume Rendering

Recently, several authors have recognized the need for volume visualization to perform abstaction beyond the established clip-plane rendering. Interrante et al. [17, 16] suggested to augment semi-transparent iso-surfaces using curvature-directed strokes and 3D Line Integral Convolution, which provides the user with intuitive cues about the shape of these surfaces. Viola et al. [33] suggested *importance driven volume rendering* to highlight interesting structures in volume data. Starting with a pre-segmented volume, a semantic importance value is applied to the segments, which affects the final image. Bruckner et al. [3] suggested a whole toolbox of automatic illustration methods to efficiently provide the user with insights about volumetric data. Bruckner et al. [2] also propose *context preserving volume rendering*, a fully automated illustration technique trying to detect interesting structures in volume data using a sophisticated, high dimensional, data-dependent transfer function. Most of these approaches provide the user only with indirect control of the screen-space location of the focus by global and/or data-dependent parameters.

The remainder of this paper is organized as follows. In Section 3 we provide an overview of our approach. Then, in Section 3.1 we describe the procedure of extracting 3D focus and context layers, including positional and normal information. The compositing pass described in Section 3.2 blends these layers together based on user-defined parameters. This section also adresses the various rendering modes and shaders available in *ClearView*. Section 4 presents results and timings to demonstrate the interactivity of our approach. Finally, we conclude and discuss some directions for future research in Section 5.

3 ClearView

ClearView is designed with respect to the user's needs by providing an interactive and intuitive means for focus+context-based exploration of large volume data sets. One of the key requirements is to instantly enable a less experienced user to explore complex volumetric data sets without extensive training and preliminary knowledge of the data. Therefore we have based our visualization modes on established techniques, often used in man-made illustrations (see Figure 3). Semantically such illustrations are composed of two segments: the focus region - in case of the camera in Figure 3 this is the digital circuit - and the context region - the camera body in this image. In the images shown, the context region is faded out using transparency to reveal the inner focus region. At the same time characteristic parts of the context are visualized to accommodate better understanding of spatial relationships between structures in the data. From these examples the following building blocks for generating technical focus+context illustrations can be deduced:

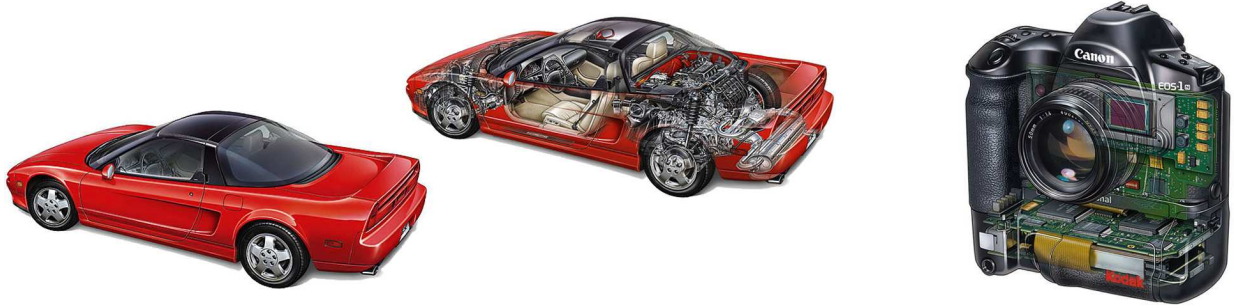

Fig. 3. Technical Illustration of Acura NSX and a Kodak digital Camera, Images courtesy of Kevin Hulsey Illustration, Inc.

- **Context**: the images contain one or more *semi-transparent* context layers.
- **Focus**: the images contain one or more *opaque* focus layers.
- **Shading**: *important features* in the context layers are emphasized to indicate the spatial context into which the focus region is embedded.
- **Compositing**: the transparency of non-important regions in the context layers is increased to reveal the opaque focus layers.

In the following section we explain the techniques developed in *ClearView* to generate focus+context-based volume rendered imagery taking into account the aforementioned paradigms. We assume that a segmented volume data set, or the data set and a set of iso-values is given. Each context layer either consists of an iso-surface corresponding to a user-defined iso-value or a user-selected segment, or it consists of all structures in front of the focus layer that is closest to the viewer.

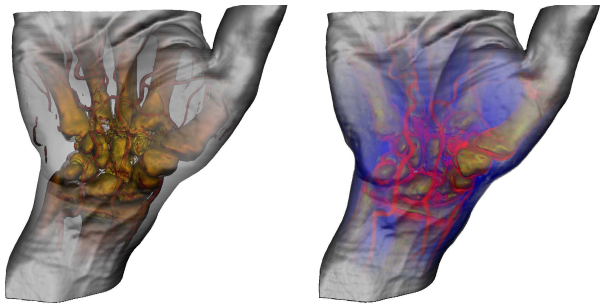

Fig. 4. This image compares the two different types of context layers surface only (left) and surface plus volume (right).

In this way not only surfaces, but also volumetric structures exhibiting a certain thickness can be displayed to provide context. Two examples demonstrating the different types of context layers supported by *ClearView* are shown in Figure 4.

The information in every layer is generated using GPU-based volume ray-casting [19], where the user selects the material properties of each layer. Since we are only interested in rendering iso-surfaces (or a relatively thin layer of structures in front of an iso-surface), ray-casting can benefit extremely from early-ray termination implemented via the early-z test on recent GPUs.

The entire image is generated in two stages. In the first stage the context and focus layers are rendered, while in the second stage these layers are composed into the final image. During the compositing stage image-based, deferred shaders are applied to each layer to enhance features or to suppress non-relevant structures.

3.1 Context and focus extraction

To render a context layer we distinguish between surface and volume layers. If the user selects an iso-value greater than zero the iso-surface corresponding to this value is rendered. Only the object space position of the first ray-surface intersection with this iso-surface is kept and stored in a floating point render target, i.e. in a geometry image. If the iso-value selected is equal to zero, all matter up to the first context or focus surface is accumulated using alpha-compositing according to the user-defined transfer function. Instead of point coordinates, accumulated color values are now stored in the render target.

After all context layers have been rendered, as many textures as layers have been generated and stored on the GPU. In the upcoming rendering pass *ClearView* generates a normal for every texel in a surface layer. This is done by using the coordinates stored in the respective texture map as texture coordinates into the volume texture. At this position the volumetric gradient is approximated by central differences along the object coordinate axes. Note that this pass - although eight fetches are performed - is still very efficient as it can effectively take advantage of texture caches on the GPU.

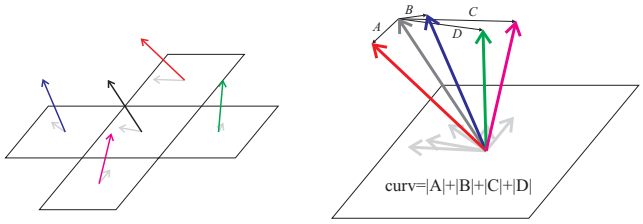

Fig. 5. Curvature estimation: The left image illustrates the normal at a surface pixel and at its four neighbors. To estimate the curvature (right) the sum of distances from the center normal to all adjacent normals is computed.

In a final pass, for every texel in a surface layer *ClearView* computes a curvature measure similar to the umbrella operator on discrete triangular meshes [18]. At each texel the summed distances between neighboring surface normals are computed. This results in values close to zero in regions of low normal variation and large values otherwise (see Figure 5). Since this operation is performed in image space, at surface silhouettes pixels containing no normal information can be considered in the curvature estimate. However, as all render targets are initially set to zero a very high curvature value is computed. This feature will later be used to highlight object silhouettes in the context layers.

The rendering pipeline for context surface layers is illustrated in Figure 6. If more than one context surface is selected by the user the algorithm is invoked multiple times. In each pass the positions stored in the current texture are used as starting points for the ray-caster. It is the users responsibility to ensure that structures in the i-th context layer are behind the structures in the layers 1 to $i-1$ with respect to the viewing direction. If structures contained in different layers intersect each other the correct blending of these structures is not guaranteed.

Figure 7 shows the different textures generated during surface layer extraction for a particular example. Texels of a volume layer are treated similarly by applying an image-based high-pass filter to enhance regions exhibiting high color gradients thus highlighting heterogeneous material. The rendering pipeline to generate a focus layer is exactly the same as the one used to generate a context surface layer.

3.2 Compositing

The compositor blends all the different textures generated during context and focus extraction. It takes into account a user-defined focus

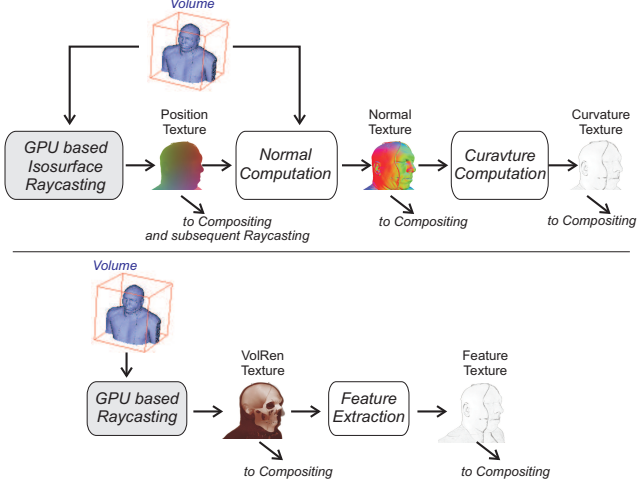

Fig. 6. Schematic overview of the surface- (top) and volume rendering (bottom) pipelines used to generate context layers as it is implemented in *ClearView*.

point, a focus region, curvature-based importance measures, and different rendering shaders. The focus point corresponds to the screen-space position of the mouse cursor projected onto the closest focus layer. This is simply the coordinate of the respective texel in the corresponding position texture. The user can also change the size of the spherical region around this 3D position in which the context information fades out from one to zero (focus point). The importance measure controls the visibility of the context layers within the focus regions via transparency modulation. Finally the user can select from a number of pre-defined shaders the one that yields the most appropriate results. Actually we have implemented four different shaders, which will be explained in Sections 3.2.5 to 3.2.7, but additional shaders can be added with ease.

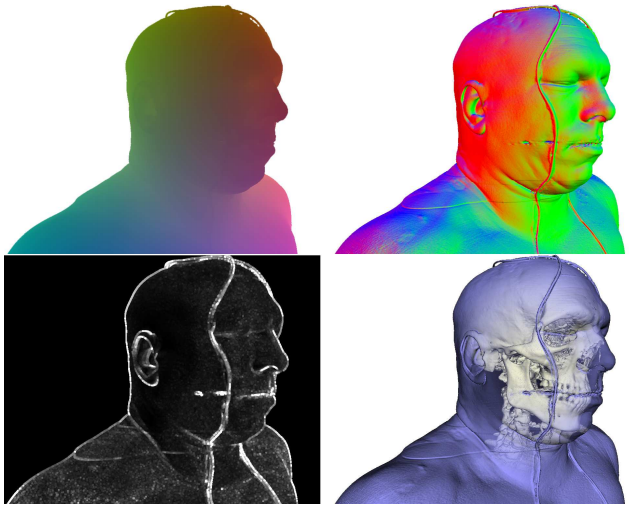

Fig. 7. These images shows the position, normal, and curvature textures, as well as a final image including a focus layer.

In the following we will discuss the particular importance measures we have integrated into *ClearView*. Technically the importance measures control the transparency of structures in the data. Here the challenge is to *automatically* and *interactively* find out what exactly is "important" and what can be neglected.

3.2.1 Curvature-based importance

As can be seen in Figure 3 artists often use the shape and in particular sharp features of the context layer as an importance measure. Especially areas exhibiting high curvature are necessary to convey the global shape of the object [17]. Therefore our first importance shader uses the curvature directly to modify transparency. Given the focus point C, the size of the focus region s, and the surface position P within the data set, the transparency is computed as

$$trans = 1 - saturate\left(max\left(\frac{|C-P|}{s}, curvature(P)\right)\right)$$

where *saturate* clamps its parameter to $[0\ldots1]$. Especially when used for technical and anatomical visualizations this importance measure gives excellent results, and it was thus chosen to be the default measure in *ClearView*. In particular compared to a clip region this approach gives significantly better results as it reveals the focus at the same time indicating the spatial context surrounding the structures in focus. Such a comparison is shown in Figure 8.

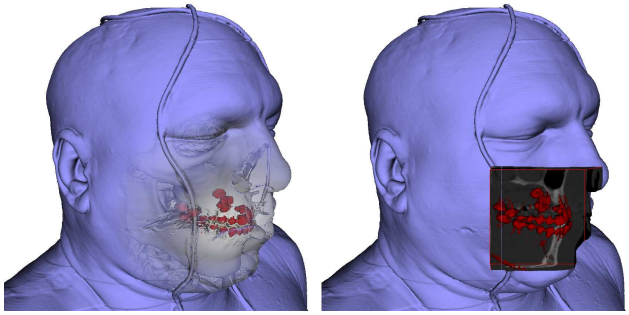

Fig. 8. The left image shows *ClearView's* curvature-based importance shader. Skin and bone are selected as context and the dental enamel is in focus. It is obviously clear that by using a clip geometry, as shown in the right, a similar context preserving visualization of the bones can hardly be achieved.

3.2.2 Distance-based importance

Instead of using the curvature-based importance measure it is often of interest to visualize how close two structures are to each other. The distance-based importance measure computes the distance from the context surface to the focus surface in the direction of the context normal (see left of Figure 9). This render mode is motivated by the observation that context surfaces running in parallel and in close proximity to the focus surface often convey no vital information whereas context surfaces with a diverging normal often relate to important features. Finally, context structures further away from any focus element - regardless of their normal - have a lowered probability of occluding important focus structures and are thus being drawn opaque. By using the identifiers from above the transparency is computed as

$$trans = 1 - saturate\left(max\left(\frac{|C-P|}{s}, normalDistance(P)\right)\right)$$

This importance measurement is often useful for technical and medical analysis as well as for pre-operative planning or structure optimization, since proximities are effectively emphasized. The integration of such a distance-based measure into the ray-casting approach is straight forward. It only requires one additional rendering pass that starts the rays at the surface points stored in the position texture and traverses these rays into the direction of the respective normal in the normal texture.

3.2.3 View-distance-based importance

The view-distance-based importance measure computes the transparency of a structure in a very similar way as the distance-based measure. In contrast, instead of computing the distance from the context surface to the focus surface into the direction of the normal the distance into the view-direction is computed (see right of Figure 9). This seemingly minor change has two important implications. On the one hand it simplifies the computation. As the intersection points between the rays of sight and the context and focus surfaces are already known for a given view, the computation of the distance into the view direction does not require an additional rendering pass. More important to the user is the visual effect of this render mode. As can be nicely

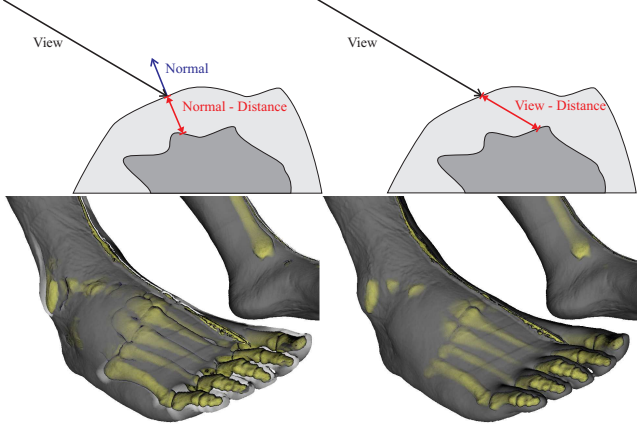

Fig. 9. This image shows the difference between the distance-based (left) and the view-distance-based (right) importance measure.

seen in Figure 10, this particular importance measure seems to hide the bone structure in a "fog of context" providing the user with an intuitive depth cue. It is particularly useful if the user wants to focus on close-to-surface structures. In the example shown transparency is computed as

$$trans = 1 - saturate\left(max\left(\frac{|C-P|}{s}, |surfF(P) - surfC(P)|\right)\right)$$

where *surfF* and *surfC* denote the extracted focus and context surfaces, respectively.

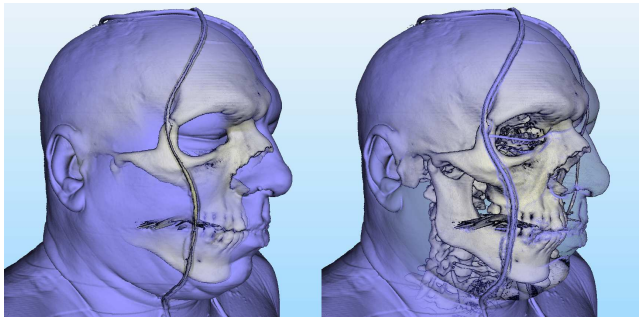

Fig. 10. In this image the view-distance-based measure (left) is used to focus only on the structures close to skin. The right image shows the curvature-based measure applied to the same data set.

3.2.4 Focus Border

To enhance the border of the focus region, and thus to guide the view of the user, *ClearView* allows to draw a border aligned to the context surface. To achieve this effect the compositor uses the distance d

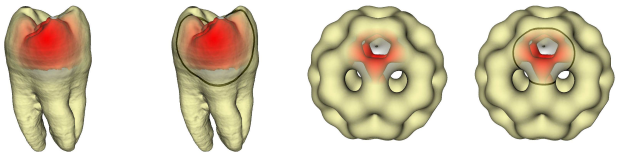

Fig. 11. The tooth and C_{60} molecule data sets are shown with and without the focus border.

of each surface point in the position texture to the focus point C. It decreases the luminance of those pixels in the focus layer for which holds: $d \in [0.95 \cdot s, s]$, where s denotes the size of the focus region. This idea is essentially equivalent to intersecting a sphere around the focus center with the surface. As can be seen in Figure 11 this extension is particularly useful in still images, as it attracts the user's attention to the focus region even more.

3.2.5 Diffuse Illumination

The simplest shader in our system evaluates just deferred diffuse lighting. Normals are evaluated as described before and are used to evaluate the diffuse dot product between normal and light direction. An ambient term is added based on the user's preferences. By default, *ClearView* positions the light at the camera. The reason is that this in general yields best contrast. Another beneficial effect of co-inciding light and view direction is that enhancement of silhouettes comes for free, since zero-crossings of the dot product between view and normal are usually identified as silhouettes.

3.2.6 Cool to Warm Shading

Based on the observation that in nature objects facing the sun are colored in a warm tone, while shadows have a bluish hue due to scattering effects, Gooch et al. [12] mandated the use of cool to warm shading for technical illustrations. This method keeps relative luminance constant, resolving the problem of low contrast in very dark and very bright areas. In *ClearView* such shading is accomplished by storing the cool to warm shades in a 1D texture. A lookup using the diffuse lighting contribution as a texture coordinate is then performed to yield the final color.

3.2.7 Skin And Bone

Though diffuse illumination or cool to warm shading yield optimum contrast and resolve the object's shape well, the resulting images may still not be fully intuitive. The reason is that the human brain not only interprets an object's shape, but also texture and material appearance. Skin and bone are readily recognized and qualities like *hard* or *soft,warm* are immediately associated. However, the scattering processes that eventually lead to the final appearance of natural materials are extremely involved and complex. Luckily, very simple approximations are sufficient to provide the brain with enough evidence to recognize the depicted material as skin or bone, resulting in intuitive images.

We provide two simple and fast empirical shaders for skin and bone. Both are based on a single 1D lookup table (LUT) that maps lighting contributions to colors. For performing the lookup, several weights A, B, C, D are computed. In the following L denotes the light direction, N the normal and V the view direction at a certain pixel. All vectors are normalized to unit length.

$$\begin{aligned} A_{bone} &= 0.5 \cdot L \cdot N + 0.5 \\ B_{bone} &= 0.7 - 0.3 \cdot L \cdot V \\ C_{bone} &= B_{bone} \cdot (0.5 - 0.5 \cdot V \cdot N) \end{aligned}$$

A_{bone} reflects the amount of diffuse light received, B_{bone} is an intermediate estimate of how much light is coming from behind the object, and C_{bone} combines B_{bone} with a silhouette weight. A weighted average of A_{bone} and C_{bone} can then be used to fetch the final color from the LUT. In both shaders, a brightening of silhouette edges serves as a very rough approximation to subsurface scattering of light behind the object, improving contrast and plausibility. For bone, a single lookup at the position $saturate(0.6A_{bone} + 0.5C_{bone})$ is performed, no additional specular or ambient component is evaluated.

For the skin shader, one lookup into the same texture is performed for each of the diffuse, specular, and silhouette components. Additional weights are computed, where $reflect(a,b)$ denotes the vector a reflected about b.

$$\begin{aligned} A_{skin} &= 0.5 \cdot L \cdot N + 0.45 \\ B_{skin} &= 0.6 - 0.4 \cdot L \cdot V \\ B'_{skin} &= saturate(reflect(V,N) \cdot L)^4 \end{aligned}$$

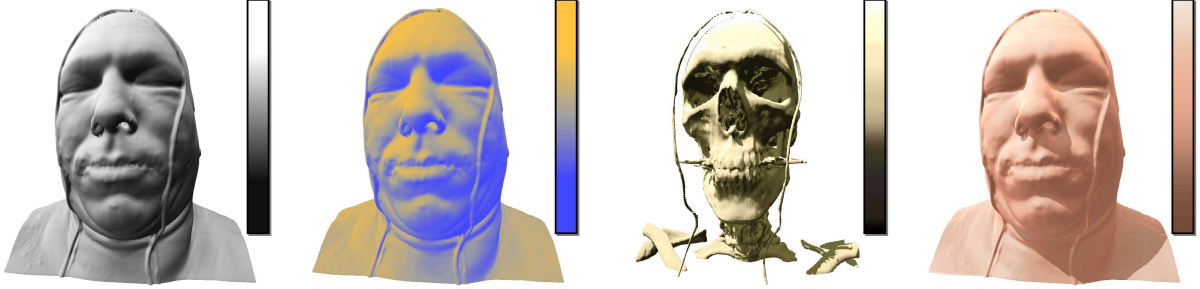

Fig. 12. From left to right: Diffuse lighting, cool to warm shading, bone, and skin shader, along with the LUT used. Bone corresponds to a different iso-value.

$$C_{skin} = 0.5 \cdot B_{skin} \cdot B'_{skin} + 0.5$$
$$D_{skin} = 0.9 \cdot saturate(1.0 - N \cdot V)^3$$

A_{skin} thru C_{skin} have the same interpretation as for the bone shader, but now C_{skin} is based on a more specular term. The new parameter D_{skin} is used to model an additional highlight at the silhouettes. Then, the three fetches into the LUT are performed.

$$Color_{diff} = LUT(A_{skin})$$
$$Color_{side} = (1.0 - N \cdot V) \cdot LUT(C_{skin})$$
$$Color_{hi} = LUT(D_{skin})$$

To accommodate for the red-shift in shadows that can be perceived in real skin due to subsurface scattering through blood (see also Figure 12, rightmost image), we further modulate the diffuse color component $Color_{diff}$ towards red if $A_{skin} \leq 0.5$. The final color is a weighted average of the above three contributions, using the intermediate B'_{skin} expanded to *rgb* as a specular term.

$$Color_{final} = 0.8 \cdot C_{diff} + 0.25 \cdot C_{side} + 0.15 \cdot C_{hi} + 0.1 \cdot B'_{skin}$$

In all of these techniques, shadows can help to make the results even more believable, especially for the purpose of still images. The shaders described here are illustrated in Figure 12

4 Results and Discussion

In the following we present some results of our algorithm, and we give timings for different parts of it. All test were run on a single processor Pentium 4 2.8 GHz equipped with an ATI X1800 XT GPU with 512 MB local video memory.

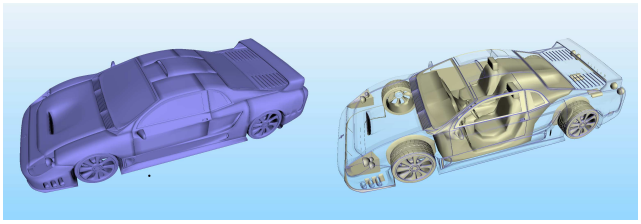

Fig. 13. This image shows *ClearView's* application to a triangle mesh.

One advantage of *ClearView* is that it is not restricted to volumetric data sets, since it consists of two stages: the extraction stage and the image-based shading and compositing stage. Hence any renderable representation, such as point-based structures, or triangle meshes can be used. Figures 13, 15, and 16 show show a selection of different data-sets being explored with *ClearView*.

To demonstrate the effectiveness of *ClearView* we have used the 512^3 thorax-abdomen-segment data set from the visible human male CT scan (see Figure 14). Two different surface layers are extracted and visualized to reveal focus and context information.

When changing parameters that only affect the appearance of the structures being extracted, such as as material properties, shaders, light

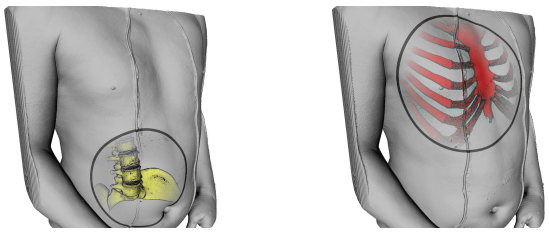

Fig. 14. Thorax/abdomen segment of the visible human male CT scan.

position and color, or focus position and size, already extracted layers can be re-used. In this case only the image-based shading and compositing stage is executed. This provides the user with the rapid visual feedback needed for interactive data exploration, while at the same time offering full control over the focus region. For a 800x600 viewport, the aforementioned shading and compositing stage is carried out at roughly 1700 fps, while still maintaining highly interactive 300 fps when increasing the resolution of the viewport to 1600x1200 pixels. Compositing three instead of two layers reduces the performance by approximately 15%. It is interesting to note that this is independent of whether an iso-surface or a volume layer is extracted. It is clear that these timings are independent on the data-set being visualized, and only depend on the image resolution as well as the number of layers to be combined.

Layer extraction is only executed when the user changes the camera position or layer-specific properties. The extraction of one single layer runs at about 30 fps on the low and 7 fps on the high resolution viewport. While the performance of the compositing stage is independent of the structure of the data set, these timings can vary due to empty-space skipping or early ray-termination. For a thorough discussion of these performance impacts we refer the reader to [19], where the underlying GPU ray-casting algorithm is discussed in detail. As multiple layers can be extracted in one single volume rendering pass by writing the layer information into multiple render targets, the time to generate images is about 21 fps for the 800x600 viewport and about 5 fps on the 1600x1200 viewport.

5 Conclusion and Future Work

We presented *ClearView*, an intuitive and interactive focus+context visualization method. In *ClearView*, the user controls the appearance of the final visualization using only a few, clearly defined parameters such as the size and location of the focus, a weight for the context, and color/material properties for the regions. Consequently, no extensive training is required for users to successfully use the system. Because *ClearView* allows for interaction with the focus+context parameters at several hundred fps even for large viewports, users are further supported by rapid visual feedback.

Since we believe that *ClearView* is a very useful tool for the rapid generation of high-quality visualizations, we would like to extend the system further in various ways. In this work, we discussed the visualization of volumetric, optionally pre-segmented data. In the future we would like to evaluate how multi-modal data fits into the sys-

tem. We would expect that using different modalities such as CT and MRI scans for different layers will convey important structures that cannot be found from visualizing either one of the modalities alone. As *ClearView* only depends on *renderable primitives*, which can be points, volumes, polyhedral surfaces etc., the restriction of combining only volumetric data is void. Furthermore, since the required features are extracted on-the-fly, we do not need additional feature data sets, allowing to investigate more compact, renderable data representations to cope with today's and tomorrow's gigantic data sets. To demonstrate the potential that lies within this flexible input data interface, we already rendered triangular meshes (see Figure 13) using *ClearView*, and we would like to further extend the palette of accepted input formats. Parallelizing the system also seems to be very promising to render the gigantic, potentially time-resolved data sets emerging lately as a result of improved numerical simulation capabilities. A trivial parallelization would just extract focus and context layers on separate GPUs. A more promising approach would be to further parallelize the extraction of each layer. Either image-space or object-space partitioning of the data is possible, since layers are extracted with full positional information and can be composed taking occlusion into account. For time-resolved data, additional research is needed as it is not obvious how the focus+context paradigm is best generalized to data sequences.

Acknowledgements

We would like to thank Kevin Hulsey Illustration, Inc. for Figure 3, The Visible Human Project for the Visible Human male CT scan, the Computer Graphics Group Erlangen for the Piggy Bank data set, the Institute of Computer Graphics and Algorithms of Vienna University for the stag beetle and www.volvis.org for the backpack data set. Some data sets were segmented by Thomas Schiwietz using the tool described in [14].

References

[1] E.A. Bier, M.C. Stone, K. Pier, W. Buxton, and T.D. Rose. Toolglass and magic lenses: The see-through interface. In *ACM SIGGRAPH*, pages 73–80, 1993.

[2] S. Bruckner, S. Grimm, A. Kanitsar, and E. Gröller. Illustrative context-preserving volume rendering. In *EuroVis*, pages 69–76, 2005.

[3] S. Bruckner and E. Gröller. Volumeshop: An interactive system for direct volume illustration. In *IEEE Vis*, pages 671–678, 2005.

[4] M.S.T Carpendale, D.J. Cowperthwaite, and F.D. Fracchia. Extending distortion viewing from 2d to 3d. *IEEE Computer Graphics and Applications*, 17(4):42–51, 1997.

[5] P. Cignoni, C. Montani, and R. Scopigno. Magicsphere: An insight tool for 3d data visualization. In *Eurographics*, pages 317–328, 1994.

[6] M. Cohen and K. Brodlie. Focus and context for volume visualization. In *Theory and Practice of Computer Graphics*, pages 32–39, 2004.

[7] L. DaVinci. Dell'anatomia fogli A et B, Quaderni d'anatomia I-IV. Collection Windsor Castle, 1478–1518.

[8] J. Diepstraten, D. Weiskopf, and T. Ertl. Transparency in interactive technical illustrations. In *Eurographics*, pages 317–325, 2002.

[9] J. Diepstraten, D. Weiskopf, and T. Ertl. Interactive cutaway illustrations. In *Eurographics*, pages 523–532, 2003.

[10] D.S. Ebert and P. Rheingans. Volume illustration: Non-photorealistic rendering of volume models. In *IEEE Vis*, pages 195–202, 2000.

[11] B.D. Fisher and Z.W. Pylyshyn. The cognitive architecture of bimodal event perception: A commentary and addendum to Radeau. *Cahiers de Psychologie Cognitive/Current Psychology of Cognition*, 13(1):92–96, February 1994.

[12] A. Gooch, B. Gooch, P. Shirly, and E. Cohen. A non-photorealistic lighting model for automatic technical illustrations. In *ACM SIGGRAPH*, pages 447–452, 1998.

[13] B. Gooch and A. Gooch. *Non-Photorealistic Rendering*. AK Peters Ltd., 2001.

[14] L. Grady, T. Schiwietz, S. Aharon, and R. Westermann. Random walks for interactive organ segmentation in two and three dimensions: Implementation and validation. In *MICCAI*, 2005.

[15] M. Ikits and C.D. Hansen. A focus and context interface for interactive volume rendering. http://www.cs.utah.edu/ ikits, 2004.

[16] V.L. Interrante. Illustrating surface shape in volume data via principal direction-driven 3D line integral convolution. In *ACM SIGGRAPH*, pages 109–116, 1997.

[17] V.L. Interrante, H. Fuchs, and S. Pizer. Illustrating transparent surfaces with curvature-directed strokes. In *IEEE Vis*, pages 211–218, 1996.

[18] L. Kobbelt, S. Campagna, J. Vorsatz, and H.P. Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *ACM SIGGRAPH*, pages 105–114, 1998.

[19] J. Krüger and R. Westermann. Acceleration techniques for GPU-based volume rendering. In *IEEE Vis*, pages 287–292, 2003.

[20] E. LaMar, B. Hamann, and K.I. Joy. A magnification lens for interactive volume visualization. In *Pacific Graphics*, pages 223–232, 2001.

[21] C.H. Lee, A. Varshney, and D. Jacobs. Mesh saliency. In *ACM SIGGRAPH*, 2005.

[22] Y.K. Leung and M.D. Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer-Human Interaction*, 1(2):126–160, 1994.

[23] M. Levoy and R. Whitaker. Gaze-directed volume rendering. In *Utah Symposium on Interactive 3D Graphics*, pages 217–223, 1990.

[24] A. Lu, C.J. Morris, D.S. Ebert, P. Rheingans, and C. Hansen. Non-photorealistic volume rendering using stippling techniques. In *IEEE Vis*, pages 211–218, 2002.

[25] E.B. Lum and K.L. Ma. Hardware-accelerated parallel non-photorealistic volume rendering. In *International Symposium on Non-photorealistic Rendering and Animation (NPAR)*, June 2002.

[26] D.W. Massaro. Attention and perception: An information integration perspective. *Acta Psychologica, Special Issue: Action, attention and automaticity*, (2–3):211–243, December 1985.

[27] M.J. McGruffin, L. Tancau, and R. Balakrishnan. Using deformations for browsing volumetric data. In *IEEE Vis*, pages 401–408, 2003.

[28] P. Rheingans and D.S. Ebert. Nonphotorealistic rendering of volume models. *IEEE TVCG*, 7(3):253–264, 2001.

[29] T. Ropinski, F. Steinicke, and K. Hinrichs. Visual exploration of seismic volume datasets. *Journal of WSCG*, 14:73–80, 2006.

[30] S.D. Shaw, J.A. Hall, D.S. Ebert, and D.A. Roberts. Interactive lens visualization techniques. In *IEEE Vis*, pages 155–159, 1999.

[31] A. Stompel, E.B. Lum, and K.L. Ma. Feature-enhanced visualization of multidimensional, multivariate volume data using non-photorealistic rendering techniques. In *Pacific Graphics*, 2002.

[32] I. Viola, E. Gröller, M. Hadwiger, K. Bhler, B. Preim, M.C. Sousa, D.S. Ebert, and D. Stredney. Illustrative visualization. IEEE Vis 2005, Tutorial #4.

[33] I. Viola, A. Kanitsar, and E. Gröller. Importance-driven volume rendering. In *IEEE Vis*, pages 139–145, 2004.

[34] L. Wang, Y. Zhao, K. Mueller, and A. Kaufman. The magic volume lens: An interactive focus+context technique for volume rendering. In *IEEE Vis*, pages 47–54, 2005.

[35] M. Weiler, R. Westermann, C. Hansen, K. Zimmermann, and T. Ertl. Level-of-detail volume rendering via 3D textures. In *IEEE VolVis*, 2000.

[36] D. Weiskopf, K. Engel, and T. Ertl. Volume clipping via per-fragment operations in texture-based volume visualization. In *IEEE Vis*, pages 93–100, 2002.

[37] R. Westermann, L. Kobbelt, and T. Ertl. Real-time exploration of regular volume data by adaptive reconstruction of iso-surfaces. In *The Visual Computer*, 1999.

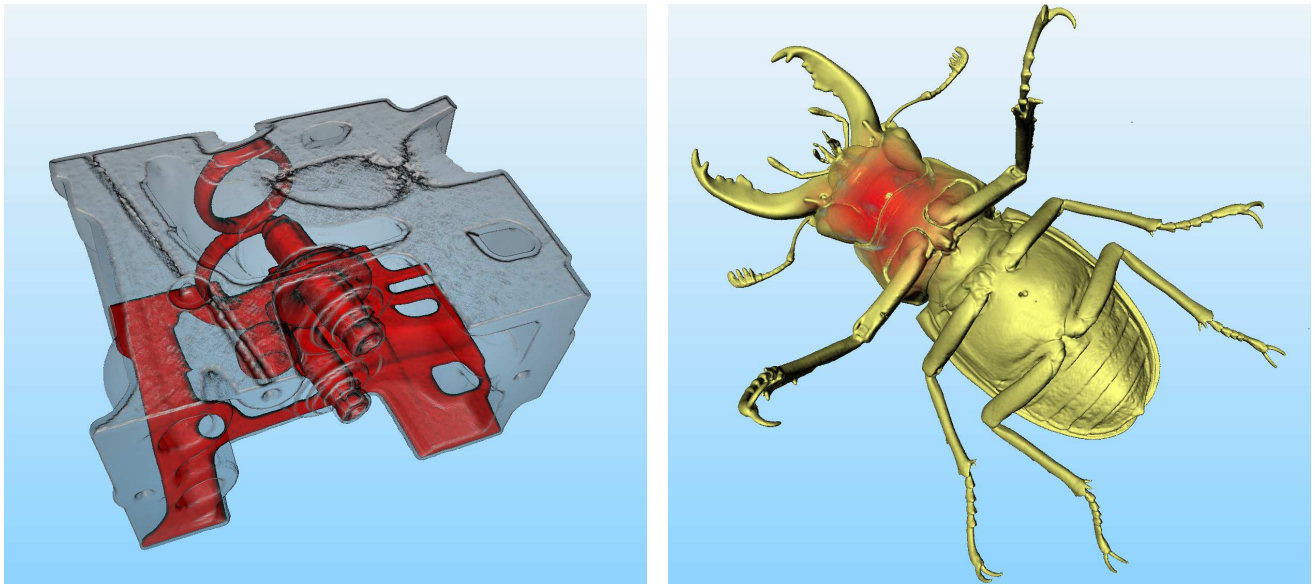

Fig. 15. This image shows the Engine data set rendered with one focus and one context iso-surface, and the stag beetle data set with a volume rendered focus.

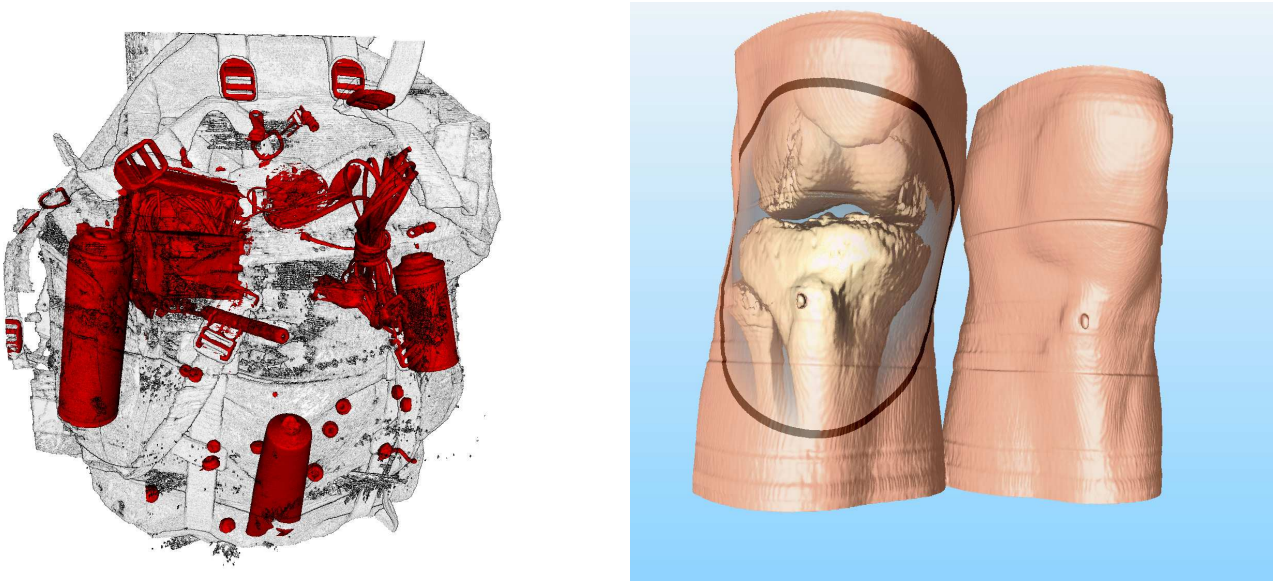

Fig. 16. In this image of the backpack data set the focus region covers the the entire image. To the right, the focus on the knee data set was set to highlight the joint between tibia and femur.