Interactive Volume Rendering based on a “Bubble Model”

Balázs Csébfalvi Eduard Gröller

Institute of Computer Graphics and Algorithms
Vienna University of Technology
Vienna, Favoritenstrasse 9-11, A-1040, Austria
{csebfalvi, groeller}@cg.tuwien.ac.at

Abstract

In this paper an interactive volume rendering technique is presented which is based on a novel visualization model. We call the basic method “bubble model” since iso-surfaces are rendered as thin semi-transparent membranes similarly to blown soap bubbles. The primary goal is to develop a fast previewing technique for volumetric data which does not require a time-consuming transfer function specification to visualize internal structures. Our approach uses a very simple rendering model controlled by only two parameters. We also present an interactive rotation technique which does not rely on any specialized hardware, therefore it can be widely used even on low-end machines. Due to the interactive display, fine tuning is also supported since the modification of the rendering parameters has an immediate visual feedback.

Key words: Volume rendering, shear-warp projection, fast previewing.

1 Introduction

Basically, there are two alternatives for high quality visualization of volume data sets. One alternative is iso-surface extraction using the “marching cubes” [1] surface-reconstruction method and the other one is direct volume rendering [2][3]. The first approach requires a time-consuming preprocessing in order to generate a polygonal mesh. Although such a mesh can be rendered interactively using conventional 3D graphics hardware, this method is limited to certain iso-surfaces defined by modality-dependent threshold parameters. Without any a priori knowledge about the data distribution it is not obvious which iso-surfaces represent the content of the volume best without significant loss of information. Furthermore, whenever a threshold value is changed the entire reconstruction process has to be repeated.

Another alternative is direct volume rendering which is a more flexible approach. Theoretically, every single voxel contributes to the final image, therefore the internal structures can also be rendered. In practice, it is rather difficult and time-demanding to specify an appropriate transfer function. Because of the exponential attenuation, the objects which are hidden by several other semi-transparent objects are hardly recognizable. Even if low opacities are assigned to the voxels only a limited number of semi-transparent iso-surfaces can be rendered at the same time. There are techniques to determine optimal threshold parameters automatically [12], and methods for effective transfer function design also exist [14][15]. The transfer function specification, however, is still data-dependent, and often requires user interaction [20].

Physics-based direct volume rendering is also limited because of the computational cost. Without using any specialized hardware device, it is not possible to render a large volume data set interactively, although it would be rather important in transfer function specification to have an immediate feedback.

Application oriented visualization models like maximum intensity projection (MIP)[5], local maximum intensity projection (LMIP)[6], or frequency-domain volume rendering [7][8] do not need time-consuming user interaction to specify the rendering parameters. Nevertheless, they also suffer from computational cost and they are limited to the medical imaging application field. Using MIP some internal features can be hidden by higher density regions and using Fourier volume rendering, which is equivalent to density accumulation, similar problems arise.

The application of well-known non-photorealistic rendering (NPR) techniques [9][11][16] is a new direction in volume rendering research. Previously, these NPR methods have been proposed for polygonal surface models, therefore their application to iso-surfaces extracted from volume data seems to be obvious. Interrante [13] uses principal-direction driven 3D line integral convolution for illustrating surface shapes in volume data. Saito proposes an NPR technique for real-time previewing of volumes [10]. His approach is also restricted to an iso-surface, where the surface is uniformly sampled and the sample points are projected onto the image plane as geometrical primitives like cross lines. The orientation of these primitives depends on the local inclination of the surface. Ebert [21] proposes a volume illustration framework combining direct volume rendering with NPR techniques. Most of the features of their general model are gradient and viewpoint dependent. Current volume rendering hardware devices do not support this model so interactive rendering is not possible.

In a certain sense, our method can also be considered as an NPR technique, since we do not concentrate on physically plausible rendering. Our major goal is to avoid a visual overload of the final image and preferably, to render interactively all the important details. The next section presents our simplified visualization model. In Section 3 the interactive rendering technique based on the bubble model is discussed. In Section 4 we present a simple volume-previewer application and we also show some rendering time measurements. Finally in Section 5 the contribution of this paper is summarized.

2 The "Bubble Model"

Using a certain volume rendering application, especially in the medical imaging area, it is rather important to reduce the time of the user interaction which is necessary to tune the rendering process. For instance, a radiologist applying a 3D diagnostical system usually does not have too much time to find the most appropriate transfer function.

In order to avoid a time-consuming specification of rendering parameters and to develop a technique for fast previewing of volumetric data we use a simplified visualization model called "bubble model". The main idea is to render several iso-surfaces as thin membranes similarly to the visual appearance of soap bubbles. We do not aim at a physically plausible model, therefore the light refraction effects are neglected. The most important feature of the model is that such thin membranes do not hide too much information behind them. In traditional direct volume rendering [4], because of the exponential attenuation only a limited number of iso-surfaces can be rendered at the same time. Decreasing the opacity assigned to an iso-surface, the objects behind it become more visible but its own visual contribution is reduced as well.

Taking these aspects into account we propose the following simplified rendering model. The "surfaceness" of a voxel is characterized by the gradient magnitude at the voxel position. If the gradient magnitude is high then the voxel belongs to an iso-surface rather than a homogeneous region. The gradient vector is estimated by calculating central differences:

$$\nabla f(x_i, y_j, z_k) \approx$$

$$\frac{1}{2} \cdot \begin{bmatrix} (f(x_{i+1}, y_j, z_k) - f(x_{i-1}, y_j, z_k)) \\ (f(x_i, y_{j+1}, z_k) - f(x_i, y_{j-1}, z_k)) \\ (f(x_i, y_j, z_{k+1}) - f(x_i, y_j, z_{k-1})) \end{bmatrix}, \quad (1)$$

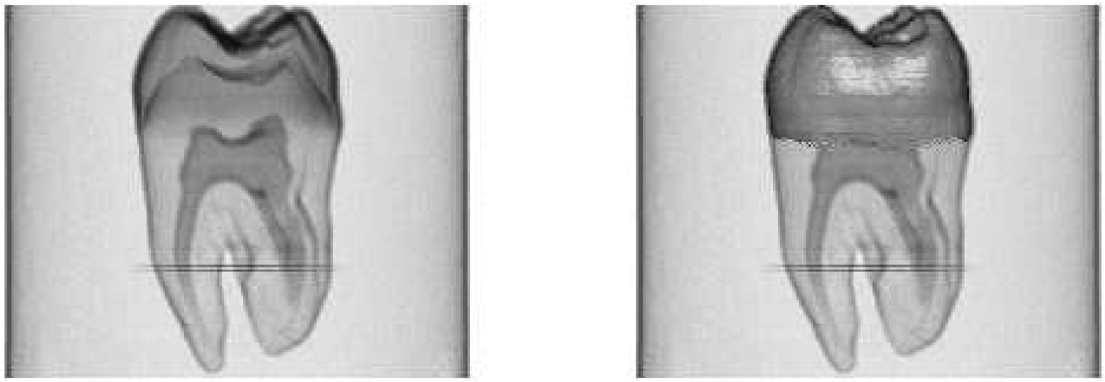
where $f(x, y, z)$ is the spatial density function. In order to avoid the staircase artifacts of this approximation a more sophisticated gradient estimation method [19] can also be used but it increases the preprocessing time as well. After having the gradient vectors calculated, opacities proportional to the gradient magnitudes are assigned to the voxels:

$$\alpha_{i,j,k} = |\nabla f(x_i, y_j, z_k)| \cdot s, \quad (2)$$

where $\alpha_{i,j,k}$ is the opacity of voxel $v_{i,j,k}$ and s is a constant scaling factor. This idea is similar to Levoy's approach [3], who proposed a 2D opacity function weighted by the gradient magnitudes in order to enhance the iso-surfaces. In contrast, we use only a 1D opacity function depending on gradient magnitudes rather than density values. This simplification has a special visual effect and it will be exploited in the interactive rendering method discussed later. Unlike the conventional light transport equation [3] we do not assign own colors to the voxels. In this sense, the voxels do not have a light reflection contribution. We assume only a constant ambient background light, and each voxel with non-zero gradient magnitude attenuates this background light. Thus, each pixel intensity is calculated as an accumulated transparency multiplied by the ambient light.

This so called bubble model can be considered as a simplified special case of the general optical model presented by Max [4]. We will show that the simplification has several advantages. Due to the opacity function weighted by the gradient magnitudes the number of voxels contributing to the image is reduced, therefore the visual overload can be avoided. Furthermore, the data reduction can be exploited in the optimization of the rendering procedure. Last but not least, the user interaction for tuning the rendering parameters is shorter and because of the fast display an immediate visual feedback is ensured.

The visual effect is illustrated in Figure 1. Those viewing rays which are nearly tangential to the iso-surface have a longer intersection segment with the region of high gradient magnitudes, therefore the corresponding accumulated transparency is lower. The rays nearly perpendicular to the iso-surface have minimal attenuation because of the short intersection segment, thus from these



(a) (b)

Figure 2: A CT scan of a tooth rendered using the bubble model (a) and the combined model (b).

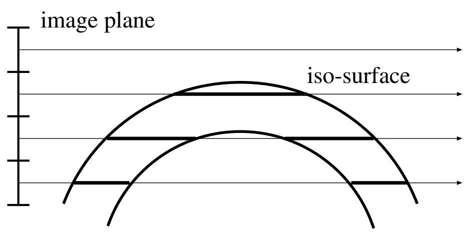


Figure 1: Opacity accumulation in the bubble model.

viewing angles the background is just slightly occluded. This model effects sharp silhouette lines at the object boundaries. This is similar to the visual appearance of a blown soap bubble, where almost just the silhouettes are visible. Nevertheless, the smooth transition at the boundaries characterizes the inclination of the surface improving the spatial impression. Figure 2a shows the CT scan of a tooth rendered using the bubble model. Note that, you can see all the important details the nerves and the complete internal structure of the tooth.

The bubble model can be combined with traditional surface shaded display. In order to avoid the visual overload of the generated image we propose only one additional shaded iso-surface. The viewing rays are evaluated according to the following algorithm:

```
double RayCasting(Volume volume,
                  Vec3D origin,
                  Vec3D direction)
{
 double transparency = 1.0;
 for(int i = 0; i < i_max; i++) {
  Vec3D sample = origin + direction * i;
  Voxel voxel = volume.Resample(sample);
  if(voxel.density > threshold)
   return Shading(voxel) * transparency;
  else {
   double opacity =
    voxel.gradient_magnitude *
    scaling_factor;
   transparency *= 1.0 - opacity;
  }
 }
 return ambient_light * transparency;
}
```

The upper formal ray-casting routine resamples the volume along a viewing ray defined by parameters *origin* and *direction*. In each *sample* point the *density* value and the *gradient magnitude* are calculated from the eight closest voxels using trilinear interpolation. If the *density* is greater than a predefined *threshold* then the lighting conditions are evaluated and the function returns the shaded color of the hit intersection point multiplied by the accumulated *transparency*. Otherwise the opacity of the current sample is multiplied by a *scaling factor* (denoted by s in Formula 2) and contributes to the accumulated *transparency*. If the ray does not have an intersection point with the iso-surface defined by the *threshold* then the function returns the *ambient light* multiplied by the accumulated *transparency*. This approach assumes that the internal structures have higher densities, like the bone in medical CT data sets. Generally, we can also use a confidence interval $[t - \epsilon, t + \epsilon]$ around threshold t to define the voxels belonging to an iso-surface.

The image in Figure 2b has been generated using the combined model. The upper part of the tooth has the highest density values, therefore setting an appropriate threshold it can be rendered separately using an arbitrary shading model. The root of the tooth has been visualized applying the bubble model.

Figure 3 shows a CT scan of a human body rendered using the bubble model (a) and the combined model (b). These images also contain almost all the internal details like the lungs, the ribs, the spine and the pelvis. Figure 3b illustrates that one additional shaded iso-surface is really a useful extension of the basic method since there can be organs with less drastic density transitions at the boundaries. For instance, the kidneys represent such a case, which can be easily rendered using surface shaded display.

3 Interactive Rendering based on the Bubble Model

In this section we present an interactive rendering technique which supports our combined visualization model. This is a pure software-only acceleration method, therefore it does not rely on any specialized hardware. The first step is a preprocessing, where at each voxel location a gradient vector is calculated. Afterwards, we extract the voxels having higher gradient magnitudes than a predefined threshold value. This results into a sparse volume which is stored in an appropriate data structure optimized for fast shear-warp projection.

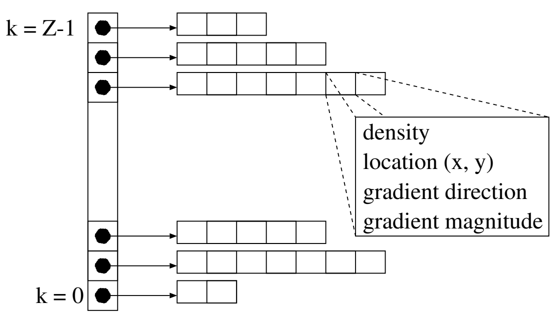


Figure 4: The data structure storing the sparse volume.

For the sake of clarity but without loss of generality, we assume that the principal component of the viewing direction is the z-coordinate and the resolution of the volume is $X \times Y \times Z$. In this case, the sparse volume is stored in a data structure illustrated in Figure 4.

The extracted voxels are stored sorted by their z-coordinates. The voxels having the same z-coordinate are stored in variable length arrays, where the length depends on the number of voxels extracted in the given z-slice. The entries contain all the information necessary for the rendering process, like the density, coordinates x and y (the z-coordinate is stored implicitly), the gradient magnitude, and the gradient direction. The gradient direction is required for the view-dependent shading of an iso-surface. This is represented by twelve bits, where the upper and lower six bits store the two polar coordinates of the gradient direction. This is used as an address into a lookup table, which contains the precalculated shaded colors under certain lighting conditions. Whenever the viewing direction or the lighting conditions are modified the lookup table has to be refreshed.

The addresses of these arrays containing the voxels with the same z-coordinates are stored in a separate pointer array of size Z. The extracted voxels are mapped onto the image plane using an efficient shear-warp projection [17] (Figure 5). The viewing transformation M_{view} is factorized into a 3D shear transformation M_{shear} and a relatively cheap 2D warp transformation M_{warp}:

$$M_{view} = M_{warp} \cdot M_{shear}$$

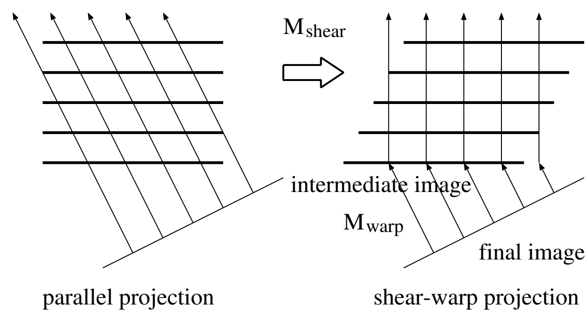
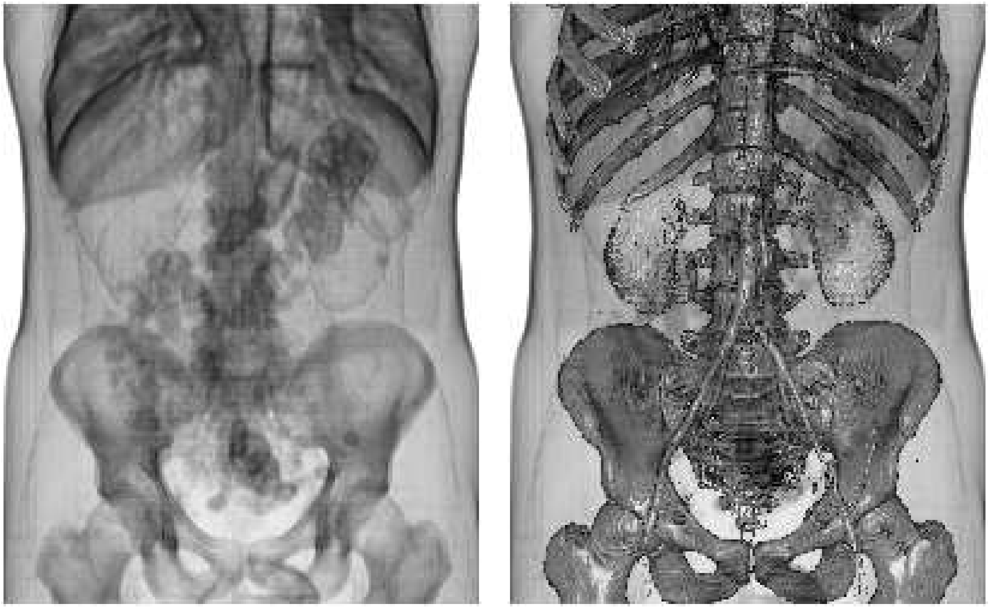


Figure 5: Shear-warp factorization of the viewing transformation.

The voxels are projected onto the intermediate image plane in back-to-front order. Initially, the pixel values of the intermediate image are set to the intensity of the background or ambient light. The current pixel values are multiplied by the transparency of the projected voxel. Whenever the density of the projected voxel is higher than the density threshold (defining a fully opaque iso-surface to be shaded), the current pixel value is overwritten by its shaded color. The shaded color is read from a precalculated lookup table, using the twelve-bit representation of the gradient direction as an address.

For the sake of efficiency, our method maps each voxel to one pixel, but in order to avoid holes, an intermediate image of size $(X + Z) \times (Y + Z)$ is generated, where the neighboring voxels are mapped to neighboring pixels. This mapping is very simple in sheared space. To each voxel location the 2D offset vector of the given slice is added. This offset is calculated for each slice in advance and only once whenever the viewing direction is changed. Assuming that the principal component of the viewing direction is the z-coordinate the maximum absolute translation of a boundary voxel is $Z/2$ in the direction of the



(a) (b)

Figure 3: A CT scan of a human body rendered using the bubble model (a) and the combined model (b).

x-axis or the y-axis. Therefore, all the projected voxels fall onto the intermediate image.

The slice offsets are calculated using a symmetric DDA line drawing method like Bresenham's algorithm[18]. The approximation of viewing rays with discrete 3D lines produces the same result as ray casting using nearest-neighbor resampling. In order to compute more accurate pixel values bilinear interpolation can also be applied taking into account the exact translations of the slices. This modification improves the image quality but drastically decreases the rendering speed. Therefore, in a practical implementation for continuous rotation the faster nearest-neighbor resampling can be used and after setting the appropriate viewing direction the more accurate bilinear interpolation is applied.

Having the intermediate image generated, it has to be mapped onto the final image using a 2D warp operation. The scaling factors can be built into the warp matrix, thus the size of the final image does not necessarily depend on the size of the original volume. In fact, the intermediate image is used as a texture map, since the final image is scanned pixel-by-pixel mapping the locations onto sample points in the intermediate image. These color samples are computed from the four closest pixels in the intermediate image using bilinear interpolation.

The upper rendering technique can be easily extended to arbitrary viewing directions, since for all the three principal directions a separate data structure can be generated. The appropriate data structure is selected during the rotation according to the principal component of the current viewing direction.

4 Limitations

The bubble model assumes that the volume contains clear boundaries between adjacent regions, therefore they can be easily detected by gradient thresholding. In case of medical CT scans this requirement is fulfilled, therefore the proposed method works well. However, there might be volumes, where a voxel belonging to an iso-surface does not necessarily have high gradient magnitude like in an electron density map or in a noisy MRI data set. In such cases our method cannot be effectively applied.

On the other hand, the appropriate threshold depends on the data set to be visualized. Therefore, one might consider it a drawback, that the gradient threshold, which strongly influences the performance, is a predefined parameter. This problem can be easily solved by slightly increasing the preprocessing time. Since each slice is represented by a voxel list, the voxel elements can be sorted by their gradient magnitudes. Assuming that the gradient magnitudes are quantized to bytes, sorting can be performed very efficiently based on the histogram of the given slice. In the preprocessing, each voxel having gradient magnitude quantized to a non-zero value is extracted from the volume. For each slice represented by a voxel list, a pointer indicates the position of the first voxel

volume	resolution	data reduction rate	frame rates
tooth	$256 \times 256 \times 161$	3.48 %	14.28 - 20.37 Hz
body	$202 \times 152 \times 255$	16.89 %	4.74 - 7.14 Hz
small head	$128 \times 128 \times 113$	25.19 %	14.08 - 20.12 Hz
big head	$256 \times 256 \times 225$	16.91 %	2.44 - 3.71 Hz

Table 1: Data reduction rates and frame rates for different data sets.

having higher gradient magnitude than the current threshold. During the rendering, each voxel list is processed only from this starting position. The gradient threshold can be interactively modified, since the new starting positions can be determined by a simple linear search. Assuming that the threshold is changing continuously, the new starting position is closed to the old one, therefore only a couple of voxels need to be checked. By selecting an appropriate gradient threshold the user can make a compromise between the rendering speed and the loss of information.

5 Implementation

The presented interactive rendering technique has been implemented in C++ under Windows NT and has been tested on a 800MHz Pentium III PC with 512M RAM. The interface of the application is shown in Figure 6. The two rendering parameters, the opacity scaling and the threshold defining a shaded iso-surface, can be controlled by two sliders. Because of the fast rendering procedure, an immediate visual feedback is ensured. The image can be rotated by moving the mouse pointer on the display window according to the drag and drop convention. Figure 7 shows the front and side views of a human head rendered using the bubble model (a, c) and the combined model (b, d). Table 1 shows the frame rates for different data sets. Note that, the rendering speed depends on the data reduction rate rather than the resolution of the volume. For example, in case of the *tooth* only 3.48% of the data is extracted. Although the volume *small head* has lower resolution, because of its worse data reduction rate (25.19%) it can be rendered approximately as fast as the *tooth*.

6 Conclusion

In this paper a new interactive volume rendering technique has been presented. We proposed a simplified visualization model that we call bubble model, since the iso-surfaces are rendered as thin semi-transparent membranes. Our opacity function weighted by the gradient magnitude reduces the number of voxels which contribute to the final image. Such an opacity mapping has two advantages. On one hand the visual overload of the image can be avoided, without significant loss of information. On the other hand the data reduction can be exploited in the optimization of the rendering process. We propose our model for fast volume previewing, which does not require a time-consuming transfer function specification. The rendering procedure is controlled by only two parameters and due to the optimization an immediate visual feedback is ensured. Since our acceleration technique is a pure software based method it does not rely on any specialized hardware to achieve interactive frame rates.

Acknowledgements

This work has been funded by the V^{is}M^{ed} project (`http://www.vismed.at`). V^{is}M^{ed} is supported by *Tiani Medgraph*, Vienna (`http://www.tiani.com`), and by the *Forschungsförderungsfond für die gewerbliche Wirtschaft*. The CT scans of the human head was obtained from the Chapel Hill Volume Rendering Test Dataset. The data was taken on the General Electric CT scanner and provided courtesy of North Carolina Memorial Hospital. The CT scan of the human body was provided by *Tiani Medgraph*. The industrial CT scan of the tooth was downloaded from web site: `http://visual.nlm.nih.gov/evc/`.

References

[1] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21, 4, pages 163–169, 1987.

[2] R.A. Drebin, L. Carpenter and P. Hanrahan. Volume rendering. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22, pages 65–74, 1988.

[3] M. Levoy. Display of surfaces from CT data. *IEEE Computer Graphics and Application*, 8(5):29–37, 1988.

[4] N. Max. Optical Models for Direct Volume Rendering. *Journal IEEE Transactions on Visualization and Computer Graphics*, Vol. 1, No. 2, pages 99–108, 1995.

[5] G. Sakas, M. Grimm, and A. Savopoulos. Optimized Maximum Intensity Projection (MIP). *Workshop on Rendering Techniques*, pages 51–63, 1995.

[6] Y. Sato, N. Shiraga, S. Nakajima, S. Tamura, and R. Kikinis. LMIP: Local Maximum Intensity Projection. *Journal of Computer Assisted Tomography*, Vol. 22, No. 6, 1998.

[7] T. Totsuka and M. Levoy. Frequency Domain Volume Rendering. *Proceedings Computer Graphics, Annual Conference Series*, pages 271–278, 1993.

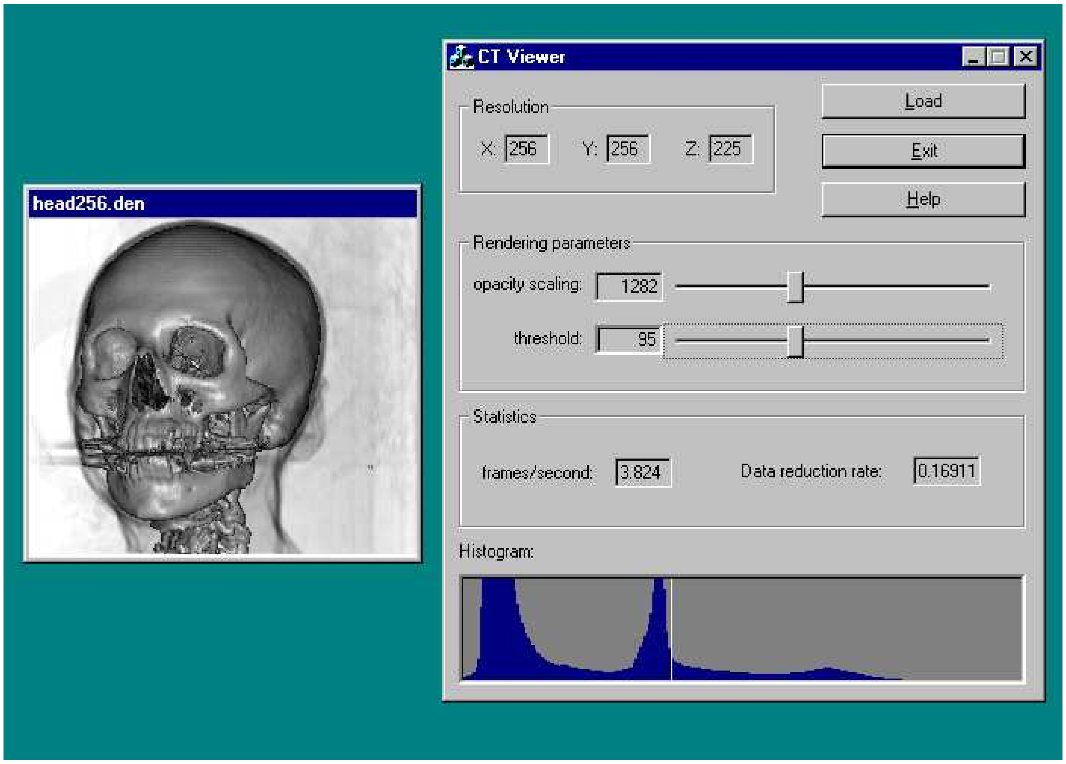


Figure 6: The graphics interface of the application.

[8] L. Lippert and M. H. Gross. Fast Wavelet Based Volume Rendering by Accumulation of Transparent Texture Maps. *Computer Graphics Forum (Proceedings EUROGRAPHICS '95)*, pages 431–443, 1995.

[9] T. Saito and T. Takahashi. Comprehensible Rendering of 3-D Shapes. *Computer Graphics (SIGGRAPH '90 Proceedings)*, pages 197–206, 1990.

[10] T. Saito. Real-time previewing for volume visualization. *Symposium on Volume Visualization*, pages 99–106, 1994.

[11] J. Lansdown and S. Schofield. Expressive Rendering: A Review of Non-Photorealistic Techniques. *IEEE Computer Graphics and Applications*, Vol. 15, No. 3, pages 29–37, 1995.

[12] G. Kindlmann and J. W. Durkin. Semi Automatic Generation of Transfer Functions for Direct Volume Rendering. *IEEE Symposium on Volume Visualization '98*, pages 79–86, 1998.

[13] V. L. Interrrante. Illustrating Surface Shape in Volume Data via Principal Direction-Driven 3D Line Integral Convolution. *Computer Graphics (SIGGRAPH '97 Proceedings)*, pages 109–116, 1997.

[14] S. Fang, T. Bifflecome and M Tuceryan. Image-Based Transfer Function Design for Data Exploration in Volume Visualization. *IEEE Visualization '98*, pages 319–326, 1998.

[15] I. Fujishiro, T. Azuma and Y. Takeshima. Automating Transfer Function Design for Comprehensible Volume Rendering Based on 3D Field Topology Analysis. *IEEE Visualization '99*, pages 467–470, 1999.

[16] G. Elber. Interactive Line Art Rendering of Freeform Surfaces. *Computer Graphics Forum (Proceedings EUROGRAPHICS '99)*, pages 1–12, 1999.

[17] Philippe Lacroute and Marc Levoy. Fast volume rendering using a shear-warp factorization of the viewing transformation. *Computer Graphics (SIGGRAPH '94 Proceedings)*, pages 451–457, 1994.

[18] L. Szirmay-Kalos (editor). *Theory of Three Dimensional Computer Graphics*. Akadémia Kiadó, Budapest, 1995.

[19] L. Neumann, B. Csébfalvi, A. König, E. Gröller. Gradient Estimation in Volume Data using 4D Linear Regression. *Computer Graphics Forum (Proceedings EUROGRAPHICS 2000)*, pages 351–358, 2000.

[20] A. König and E. Gröller. Mastering Transfer Function Specification by using VolumePro Technology. *Tecnical Report TR-186-2-00-07, Institute of Computer Graphics and Algorithms, Vienna University of Technology*, 2000.

[21] D. Ebert and P. Rheingans. Volume Illustration: Non-Photorealistic Rendering of Volume Data. *Proceedings of IEEE Visualization 2000*, pages 195–202, 2000.

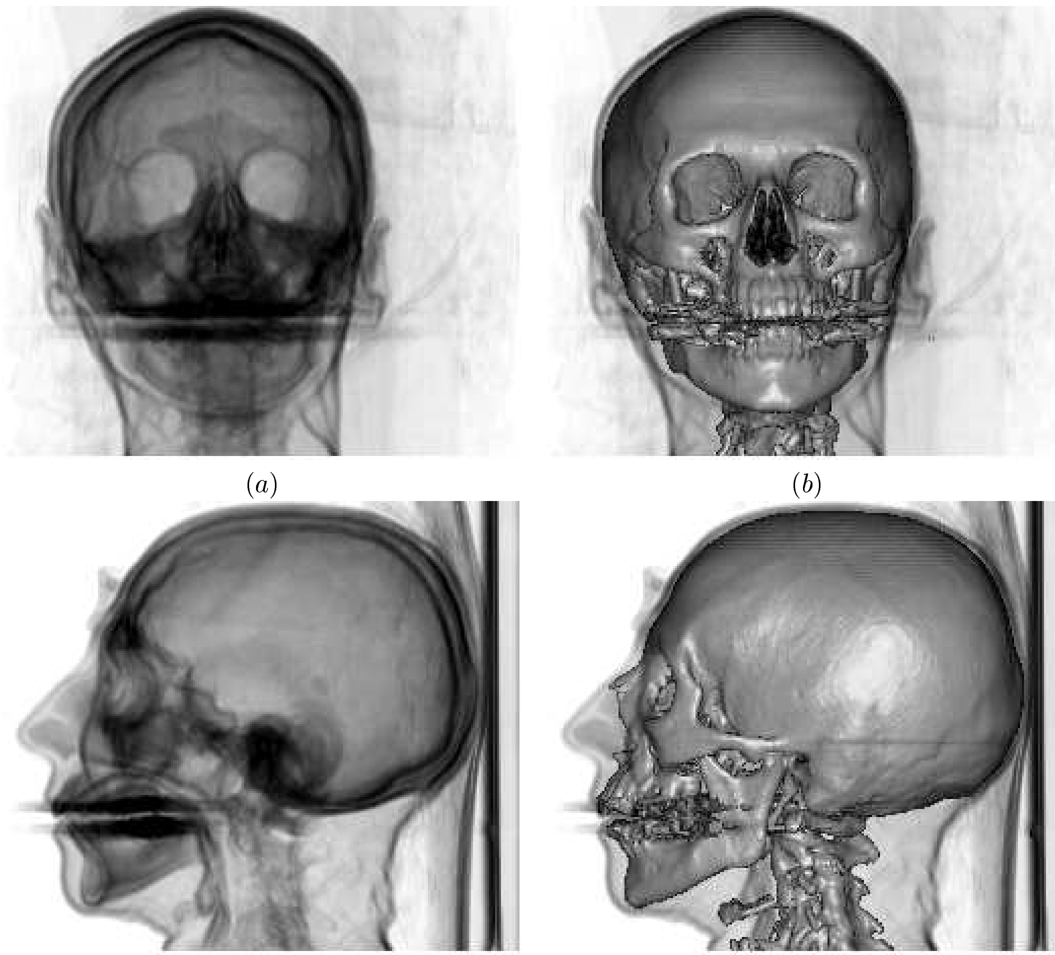


Figure 7: A CT scan of a human head rendered using the bubble model (a, c) and the combined model (b, d).