Automatic Generation of Volumetric Transfer Functions

Kai Uwe Barthel

Hochschule für Technik und Wirtschaft Berlin
Wilhelminenhofstraße 75A, 12459 Berlin
Germany

ABSTRACT

For high quality 3D volume renderings transfer functions have be defined. Typically 1D transfer functions are used that assign a color and an alpha value to every intensity value of the volume data. The choice of transfer functions often is quite difficult and users have to go through a trial and error approach to obtain "good" visualizations. 2D transfer functions use the gradient of the volume data as additional dimension to assign colors and alpha values. In this paper the new features of the ImageJ Volume Viewer plugin will be presented, among others it will be possible to get automatic proposals for 1D or 2D transfer functions that will lead to pleasing volume renderings. These transfer functions serve as a starting point but they still can be modified to change the way the volume data is rendered.

1. INTRODUCTION

The Volume Viewer [1] was started in 2005 as a simple ImageJ plugin that did not use any Java 3D support for hardware-acceleration. Its purpose was to enable 3D visualizations of image stacks of volume data. The idea to build the plugin without Java 3D resulted from the fact that quite a few ImageJ users with no computer science background found it cumbersome to install the required Java 3D framework. In addition the author liked the idea of writing a volume viewer totally from scratch. A third motivation was the ability to try new visualization concepts in an own framework without the restrictions of 2D texture slices or a 3D texture on a GPU.

Further author information: (Send correspondence to Kai Uwe Barthel) E-mail: barthel@HTW-Berlin.de

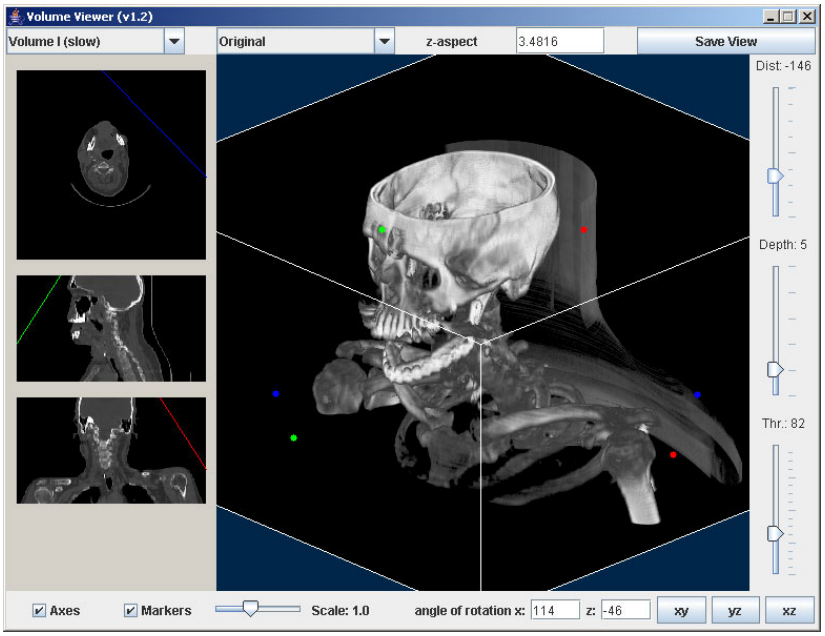

Figure 1. Volume Viewer Plugin

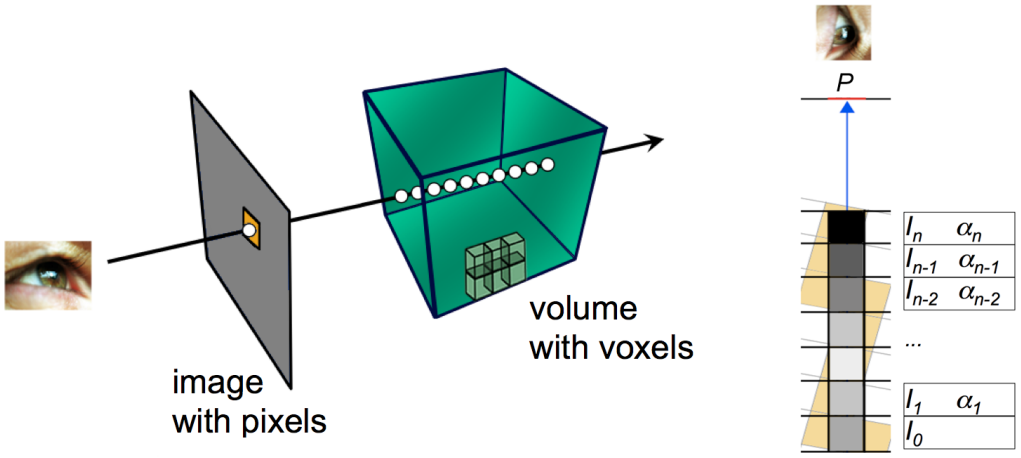

Figure 2. Principle of direct volume rendering, intensities and alpha value of successive layers of the volumetric data

Although the first version of the plugin (fig. 1) was rather simple, many people used it and there have been quite a few requests for improvements and extensions. This paper first will explain the basic idea of direct volume rendering. Then the added features of the newest version of the plugin will be presented. One of the most important tasks for visually pleasing volume renderings is the easy generation of good transfer functions. Here a good compromise between a not too complicated usage and a maximum of freedom in setting the parameters has to be determined. The new version of the plugin solves these contradictory requirements by offering different modes of operation. For each mode an automatic proposal for the parameter settings is made, but the user still has the possibility to modify them.

2. DIRECT VOLUME RENDERING

2.1 1D transfer functions

Figure 2 shows the basic principle of direct volume rendering. Direct volume rendering methods generate images of a 3D volumetric data set without explicitly extracting geometric surfaces from the data [2]. For each pixel of the image to be rendered a ray is cast through the volume. Along the ray the volume data is sampled at evenly spaced locations. If the volume is not aligned with the image plane the voxel values at the sampling positions have to be interpolated. For every sample a color and an alpha value have to be assigned. By alpha-blending the pixels of successive layers the final visualization of the volume is rendered. Very simple approaches use the voxel intensity as color and a linear function of the intensity to determine an alpha value. More sophisticated approaches allow an individual assignment of color and alpha as functions of the intensity values. Figure 3 compares two 1D transfer functions and the corresponding renderings.

With carefully chosen transfer functions high quality renderings can be achieved. However the choice of transfer functions often is quite difficult and users have to go through a trial and error approach to find “good”

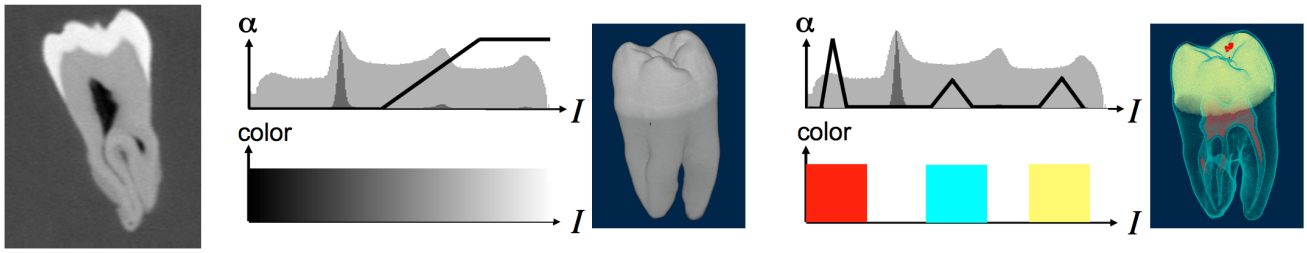

Figure 3. Two examples for 1D transfer functions and their corresponding renderings

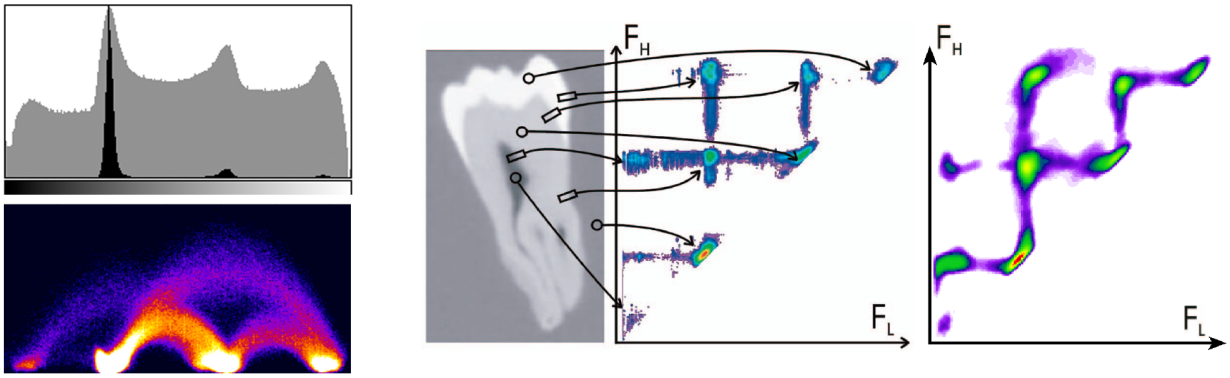

Figure 4. Intensity histogram, intensity-gradient histogram an LH-histograms from [4] und [5]

visualizations. Analyzing the histogram of the volume data may help but there is no easy intuitive way that guarantees to find suiting transfer functions.

2.2 2D transfer functions

Using only the intensity values of a volume often is not sufficient to get good renderings. Typically the boundaries of an object are the most important regions that need to be rendered well. If other "unimportant" regions within the volume share the same intensities values as the "important" boundaries, then these different regions cannot be distinguished well. To be able to treat these regions differently, higher dimensional transfer functions have been proposed [3]. For 2D transfer functions typically the gradient values of the volume data are used in addition to the intensity values. By assigning alpha and color values according to this 2D intensity-gradient histogram, the rendering process can be controlled more exactly.

Instead of using gradients which are often affected by noise, Sereda et al. [4] introduced the LH space as transfer function domain. They assume that every voxel either lies inside a homogeneous material or on a boundary between two materials with lower and higher intensity. The LH histogram is a 2D histogram that is built from the data set by accumulating boundary voxels with the same (FL,FH) coordinates, which are retrieved by analyzing the intensity profile across a boundary. This technique gives good results but requires complex computations that have to be performed in a preprocessing step. In [5] a simplified approach was proposed to calculate the boundaries more efficiently. In the new version of the Volume Viewer plugin an even simpler approach is used that accomplishes the boundary detection based on a simple clustering approach.

3. NEW FEATURES OF THE VOLUME VIEWER PLUGIN

Compared to the first version of the Volume Viewer plugin many changes, bug fixes and enhancements were implemented. As one of the most requested features the plugin now finally is resizable in order to be able to generate larger renderings. A virtual trackball model is used to allow an easy rotation of the volume in all directions using a mouse. Finally all three axes may be set to specific rotation angles with text fields. In addition to the existing nearest neighbor and bilinear interpolation modes two tricubic interpolation methods (polynomial and spline) were implemented for higher visual quality of the slices and volume renderings. For very small volumes with only very few layers rendering artifacts are often visible. To handle this problem higher sampling frequencies along the ray may be used, in addition the starting points of the rays are dithered. Maximum and mean projections were added as additional rendering modes. Due to optimizations and multithreaded calculations the plugin has also become much faster.

To choose and modify the transfer functions four different modes have been realized. In addition to the simple slider based setting of a threshold that was used in the first version of the plugin, now also 1D transfer functions can be used where all RGBA function values may be drawn individually with the mouse. Another

possibility is to edit the 2D transfer function within the intensity-gradient histogram where colors and alpha values may be assigned to particular histogram regions. As the forth mode LH-histograms may be used. The LH-histograms are calculated very efficiently using integral images. Due to the integral images it may be checked very efficiently whether a voxel might be part of a boundary. For those voxels a simple two-class clustering algorithm is performed to check if the voxel really is from a boundary. For each of the four transfer function modes the plugin may calculate a proposal for the parameter settings. These settings then may be changed in order to improve or modify the rendering.

4. CONCLUSION

In this paper we have presented the new version of the Volume Viewer plugin, which has been improved in many ways. The most important change is the possibility to generate high quality transfer functions for volume rendering very easily. Four different modes to choose a transfer function have been implemented. For each volumetric stack, the plugin analyses the data and proposes a suiting transfer function.

5. REFERENCES

[1] http://rsbweb.nih.gov/ij/plugins/volume-viewer.html

[2] Marc Levoy. Display of Surfaces from Volume Data. IEEE Computer Graphics & Applications, 8(5):29–37, 1988.

[3] J. Kniss, G. Kindlmann, C. Hansen, Multidimensional transfer functions for interactive volume rendering, IEEE Transactions on Visualization and Computer Graphics, 2002

[4] P. Sereda, A. V. Bartoli, I. W. O. Serlie, F. A. Gerritsen
Visualization of boundaries in volumetric datasets using lh histograms.
IEEE Transactions on Visualization and Computer Graphics 12, 2 (2006), 208–218.

[5] J. Praßni, T. Ropinski, K. Hinrichs
Efficient Boundary Detection and Transfer Function Generation in Direct Volume Rendering,
Proceedings of the 14th International Fall Workshop on Vision, Modeling, and Visualization (VMV09), 2009