Assignment 1 – Visualization

Volume Rendering

10-12-2017
Safa Dündar 0875719
Tsjarco Massa 0861036

Contents

1. Introduction

In computer graphics, volume visualization is a useful method for analyzing 3D volume data as it enables to get more insight in this data. Volume data are sampled functions of three spatial dimensions. With a software ray casting application, this spatial data can be visualized. Visualization is needed to play with this spatial data and conduct exploratory research to get interesting insights. In this report, a software ray caster application is extended to visualize different kinds of volume data and exploratory analysis is conducted to get interesting insights.

A traditional way to visualize volume data is to create a software ray caster manually. However, it is rather error-prone and time consuming. To visualize volume data, an initial setup of a software ray caster in Java called VolVis is provided to visualize the data. However, this VolVis application is not complete to get interesting and accurate insights.

In this report, our goal is to extend the VolVis application with additional ray casting techniques. By this the volume data can be visualized in different ways. The methodology to reach our goal is summarized as follows:

1. Study and analyze the VolVis application and understand how the software ray caster works.
2. Extend the VolVis application with additional ray casting techniques.
3. Visualize and analyze the volume data to get more insights and report them.

First of all, we need to understand what ray casting is about and what the different ray casting techniques are. The techniques that are implemented in the VolVis application are Slicer, Local Maximum Intensity Projection (LMIP), Compositing and 2D Transfer Function. Slicer is discussed in chapter 2.2. MIP and Compositing are discussed in Chapter 2.3. In Chapter 3, the 2D Transfer Function is discussed in addition with the Phong Shading Model. Lastly, exploratory data analysis is conducted to get insights in the volume data. This is discussed in Chapter 4.

2. Ray Casting

Ray casting is a volume rendering method and it is a commonly used projection algorithm in computer graphics. Volume rendering is the display of a 3D dataset onto a 2D screen (Ray, Pfister, Silver, & Cook, 1999).

In ray casting, rays are cast into the dataset. Each ray comes from the view point, penetrates a pixel on the screen and passes through the 3D dataset. Sample values are then computed by linear interpolation at evenly spaced intervals. (Tri) linear interpolation will be discussed in the next subsection. Finally, for each of these samples the voxel values are computed and are mapped to different properties by different ray casting techniques such as color and opacity. This is then visualized onto the 2D screen off the user. Like pixel represents values on a 2D screen, a voxel represents a value on a regular grid in a 3D space. Figure 1 shows the schematic visual representation of the ray casting technique.

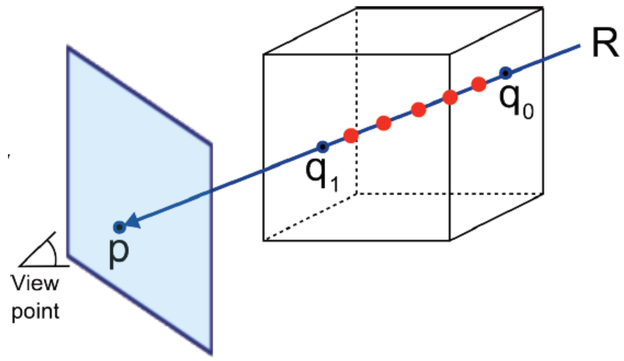

Figure 1: Schematic visual representation of the ray casting technique.

2.1. Tri-linear interpolation of samples along the viewing ray

From the given datasets we only know the pixel coordinates on the screen (the viewpoint of the user in Figure 1) and the corner points of the voxels in the 3D image. However the ray that is casting through the 3D image's voxels is not always exactly casting through a corner point of a voxel. Therefore it is necessary to be able to calculate the value within a voxel at any point. Therefore we use tri-linear interpolation, which makes it possible to calculate the value at every given point using the surrounding voxel coordinates. This is schematically represented in Figure 2.

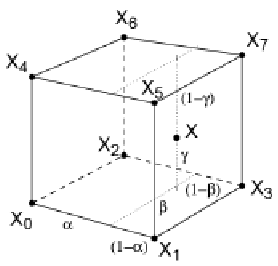

Figure 2: Schematic representation of tri-linear interpolation.

The formula used to calculate point ´X´ in Figure 2 is given below. Tri-linear interpolation is actually three times linear interpolation on the three dimensions of a 3D image. Tri-linear interpolation is used as default in the ray casting application.

$$S_X = (1-\alpha)(1-\beta)(1-\gamma)S_{X_0} + \alpha(1-\beta)(1-\gamma)S_{X_1} + (1-\alpha)\beta(1-\gamma)S_{X_2} + \alpha\beta(1-\gamma)S_{X_3} + (1-\alpha)(1-\beta)\gamma S_{X_4} + \alpha(1-\beta)\gamma S_{X_5} + (1-\alpha)\beta\gamma S_{X_6} + \alpha\beta\gamma S_{X_7}$$

2.2. Slicer

The slicer function selects the image at the center of the ray in the boundary box. The boundary box represents the edges of scanned data representing the image. The slicer is the most simplistic visualization function discussed and analyzed. The slicer could be compared to physically cutting though an object in order to see the 'slice' of your cut.

2.3. MIP and compositing ray functions

In the following chapters the MIP and the compositing ray functions are discussed.

2.3.1. MIP (maximum intensity projection) – LMIP

MIP stands for maximum intensity-projection. As the name suspects it presents the maximum intensity of a ray. This technique is commonly used to visualize specific objects with high intensity. An example for the application of the maximum intensity projection is the use of contrast fluid during surgery. The formula for the MIP function is given below.

$$I(p) = \max_t \; s_t$$

This formula follows the ray through the image and saves the voxel value with the highest intensity. All voxels with highest intensity are than displayed for form the image.

We extended the MIP function into the LMIP function, which stands for Local Maximum Intensity Projection (Sato, Shiraga, Nakajima, Tamura, & Kikinis, 1998). According to this article LMIP is more efficient in finding the maximum intensity, because it considers a value as maximum of the ray when it is above the threshold of 95% of the maximum intensity of the whole dataset. Through this approach the information used for depiction of the original image is improved. We implemented this by adding a break in the loop for finding the maximum intensity of a ray. When a voxel has an intensity higher than 95% of the maximum intensity, the break is triggered and the next pixel is investigated.

2.3.2. Compositing

Compositing is a ray casting technique which makes use of the color and the opacity of a transfer function to color the original image. Transfer functions are manually made for each image give the image more intuitive meaning. For each sample on the ray the voxel value is calculated by using the tri-linear interpolation introduced in section 2.1. From these voxel values, the color and opacity is extracted from the transfer function. The composite color is then calculated using the back-to-front composition formula. The new composite color is calculated by multiplying the voxel color and opacity and adding the multiplication of the one minus voxel opacity and the previous composite color. The opacity is used to make for example the outside of an image more opaque, whereby the inside becomes visible. The first formula is the general formula for volume rendering compositing. The second equation discretizes and simplifies the first formula. The third formula is the implementation for the back-to-front compositing.

$$I(t) = I_0 e^{-\int_0^t \tau(u)du} + \int_0^t c(s) e^{-\int_0^t \tau(u)du} ds$$

$$I(p) = \sum_{i=0}^{n-1} c_i \prod_{j=i+1}^{n-1} (1 - \tau_j)$$

$$where\ c_i = color\ and\ \tau_j = opacity$$

$$C_i = \tau_i c_i + (1 - \tau_i) C_{i-1}$$

$$where\ C_i = new\ composite\ color\ and\ C_{i-1} = previous\ composite\ color$$

2.4. Resolution interaction

The software ray caster application is quite slow during runtime. One cause is that every (sample) voxel value is calculated along the ray as it passes through the 3D dataset. This is necessary when the precise depiction of the image is required. Calculation of fewer samples on the ray, thus increasing the step size, will result in a smoother interaction with the ray caster application. However, increasing the step size could lead to images which are like slices and thus are not smooth. For example in the LMIP ray caster, the maximum intensity value could be missed with larger steps. We solved the slow interaction by decreasing the number of samples calculated only when the user toggles to change the image outlay. After the user releases the mouse button the image will be enhanced through using step size one again. Hereby the interaction with the ray caster application is faster, but the image does not lose any information. An example of the visualization during interaction and without interaction is represented in Figure 3.

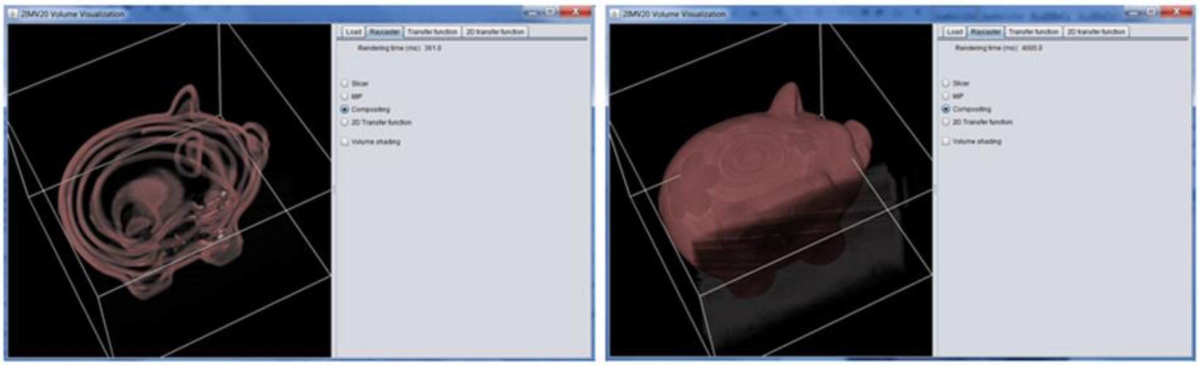

Figure 3: Example of resolution interaction. Left increased step size during toggle and right enhanced image after toggle.

3. 2-D transfer functions

3.1. Gradient-based opacity weighting

A 2-D transfer function is capable, more than a 1-D transfer function, of capturing complex combinations of material boundaries. In addition as we have seen in chapter 2.3.2 a 1-D transfer function has to be manually made, which is time consuming as it is trial-and-error based. 2-D transfer functions are more data centric as they present voxel intensity and gradient magnitude in a histogram. From this histogram the user could more easily customize the image to its needs.

The formula below calculates the gradient that is used by the gradient-intensity histogram.

$$\nabla f(x_i) = \nabla f(x_i, y_j, z_k) \approx \begin{pmatrix} \frac{1}{2}\Big(\nabla f(x_{i+1}, y_j, z_k) - \nabla f(x_{i-1}, y_j, z_k)\Big), \\ \frac{1}{2}\Big(\nabla f(x_i, y_{j+1}, z_k) - \nabla f(x_i, y_{j-1}, z_k)\Big), \\ \frac{1}{2}\Big(\nabla f(x_i, y_j, z_{k+1}) - \nabla f(x_i, y_j, z_{k-})\Big) \end{pmatrix}$$

Where X_i is a voxel location the x_i, y_j, z_k

3.1.1. Classification – Isovalue contour surfaces

Classification is used to classify different tissues in order to make them individually visible while they are overlapping. The formula below gives the opacity calculation used during classification.

$$\alpha(x_i) = \alpha_v \begin{cases} 1 & if\ |\nabla f(x_i)| = 0\ and\ f_v = f(x_i) \\ 1 - \frac{1}{r}\left|\frac{f_v - f(x_i)}{|\nabla f(x_i)|}\right| & if\ |\nabla f(x_i)| > 0\ and\ f(x_i) - r\,|\nabla f(x_i)| \ \le\ f_v \le f(x_i) + r\,|\nabla f(x_i)| \\ 0 & otherwise \end{cases}$$

Where r = radius (desired thickness in voxels of the transient region.)

3.2. Range of gradient magnitudes (Kniss)

Material boundaries defined by the 2-D transfer function often overlap. This could lead to erroneously colored parts of the image. To minimize this, values that are larger than the maximum gradient magnitude of a material are ignored (Transactions, Visualization, & Graphics, 2010). This way, parts of the erroneously colored parts are left out. The trianglewidget in the Java application is extended to implement this "Kniss" constraint.

3.3. Illumination model (Phong model)

The shading model by Phong provides better quality and increased realism in generated images (Phong, 1975). For this method it is assumed that the light source gives parallel light and the light source is coming from the user's location. The formula below is used for the shading technique.

$$I = I_a k_{ambient} + I_l k_{diff}(\boldsymbol{L} \cdot \boldsymbol{N}) + I_l k_{spec}(\boldsymbol{V} \cdot \boldsymbol{R})^{\alpha}$$

Symbol	explanation
L	normalized vector in direction of light source
N	server normal at voxel location
V	normalized vector in direction of observer
H	normalized vector in direction of maximum highlight
Kambient	ambient reflection coefficient
Kdiff	diffuse reflection coefficient
Kspec	specular reflection coefficient
Alpha	exponent used to approximate highlight

Table 1: Explanation of symbols used in Phongs shading model

The formula above can be simplified into the following formula.

$$I = I_a k_{ambient} + I_d k_{diff}(\boldsymbol{L} \cdot \boldsymbol{N}) + k_{spec}(\boldsymbol{V} \cdot \boldsymbol{H})^{\alpha}$$

As described already above parallel light is used and the light is coming from the user. For this reason the value of H is equal to the value of L and equal to the value of V.

4. Exploratory data analysis

This chapter discusses the 'carp8' and the 'pig8' datasets in detail. The parameters used for the Phong Shading model are Kambient=0.1, Kdiff=0.7, Kspec=0.2 and Alpha=10.

4.1. Carp8

Slicer

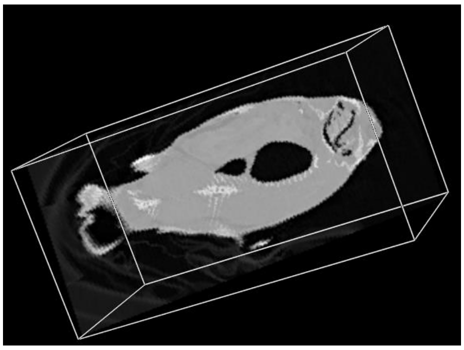

Figure 4: Carp8 slicer representation

Investigating the slice of the carp data set the first thing that immediately stands out are the two tissues. These are in the middle of the carp and are displayed in black. Secondly, there are two black lines in the head of the carp. Probably these are the gills of the carp. Thirdly, looking at the surrounding of the carp, it looks like that this carp is in some medium. Since it is a fish, we suspect the carp is in water. Lastly, the fish bones can slightly be seen. We think that by using the LMIP ray casting technique the fish bones should stand out. Using the slicer it is not possible to visualize for example the skin of the carper, because this is simple not in the middle (slice) of the data set.

LMIP

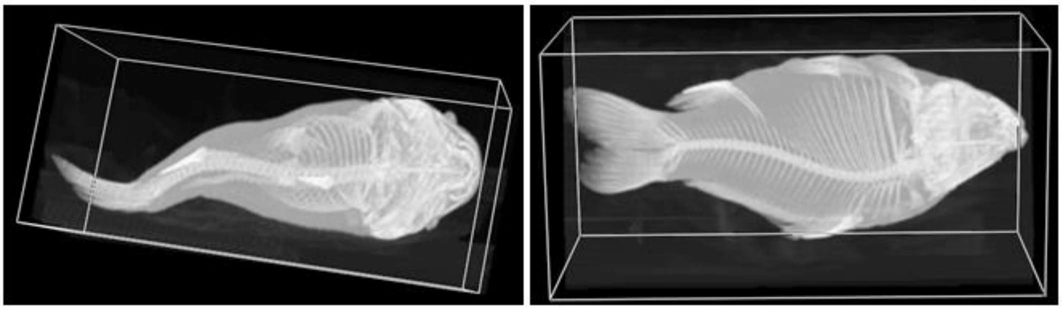

Figure 5: Carp 8 dataset LMIP representation

Using the Local Maximum Intensity projection the fish bones are standing out as expected. Furthermore the tail is surprisingly visable. Apparently the tail contains tissue with high intensity, which is surprising because the flexibility of the tail.

The most interesting part is the head of the carp. At the right side of Figure 5 the bones in the head are not distinguisable. The left side of Figure 5 provides better visualization of the head of the carp. Compared to the slicer, where only the shape of the carp is visable, the LMIP gives better insights in high intensity tissues in the fish such as bones.

Compositing

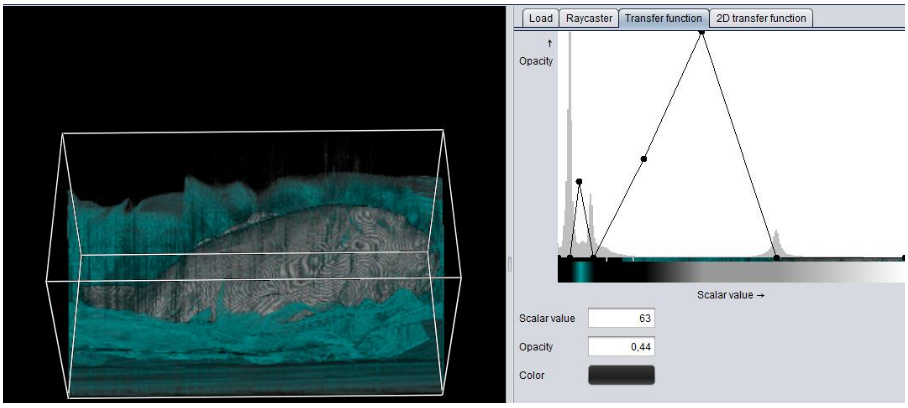

Figure 6: Carp8 dataset compositing surrounding representation

Using compositing we distinguish the carp and the surrounding using the transfer function shown in Figure 6. The surrounding of the carp is colored blueish, because intuitively carps swim in water. On the right side behind the transfer function the histogram is visible. The high peaks on the left side of the histogram are related to the surrounding of the carp. For the further analysis we will ignore these peaks in order to take a closer look at the carp only.

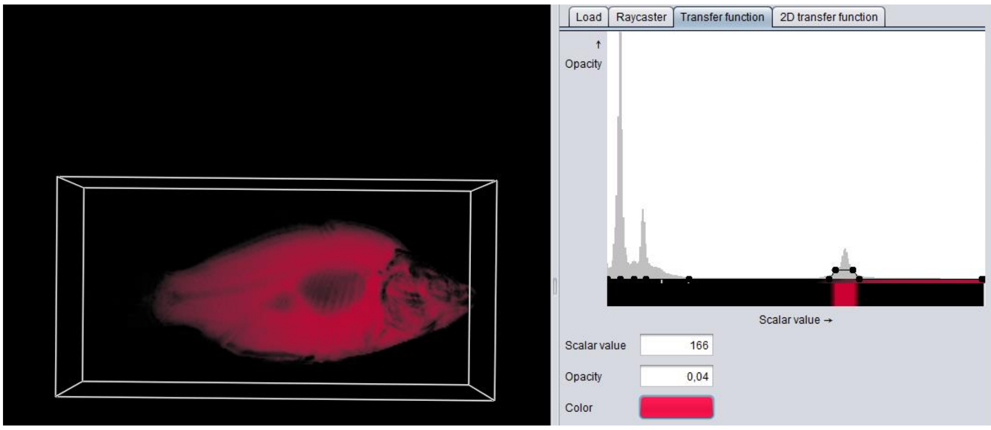

Figure 7: Carp8 dataset composition flesh tissue representation

From Figure 7 we observe that the small peak on the right side of the transfer function panel contains information about the flesh tissue of the carp. Looking closely, the forms of the eyes are visible and we see the two tissues inside the fish as well.

In Figure 8 we combined the important features of the carp in a single visualization.

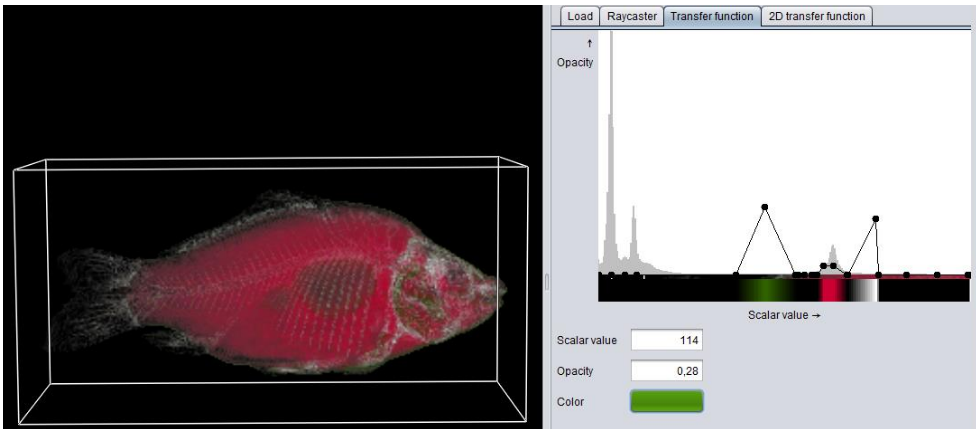

Figure 8: Carp8 dataset composition flesh and bones representation

In Figure 8 we observe the bones of the carp in white like the LMIP function did before. Furthermore, we tried to highlight the gills of the carp in green which is more opaque in the head of the carp.

2D Transfer Function

With the 2D Transfer Function, separating different materials is easier. With this function we tried to get insights about the tissues of the carp. In Figure 9 the picture of the carp is shown using the 2D Transfer Function.

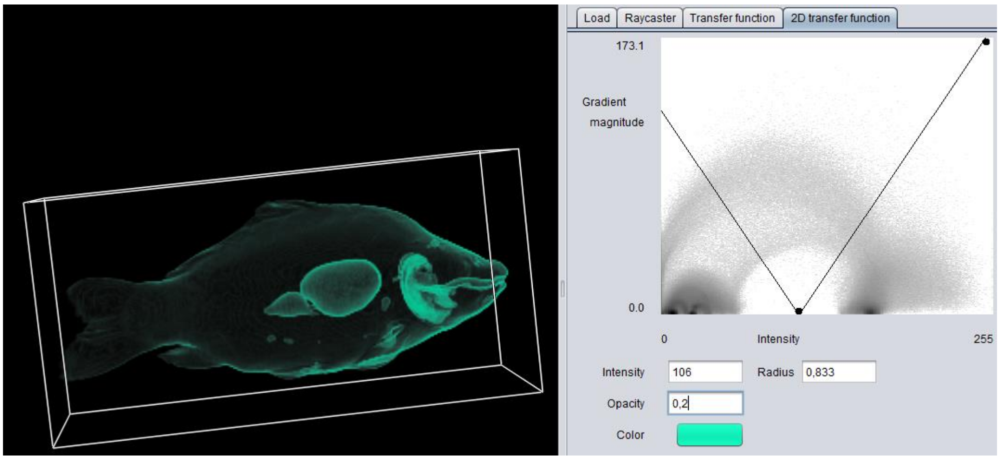

Figure 9: Carp8 dataset 2D Transfer Function representation

As can be immediately seen, the tissues are more distinguashable compared to the compositing function. First of all, there are two tissues in the corpus of the carp. We think that the bigger tissue is the stomach and the smaller tissue is the liver of the carp. In addition, the gills of the carp are more clear using the 2D Transfer Funtion. There are also even smaller tissues in the carp, especially the tissue on the bottom of the carp between the stomach and the gills is interesting for further investigation. Maybe the Phong Shading model will give us more insights.

Phong Shading Model

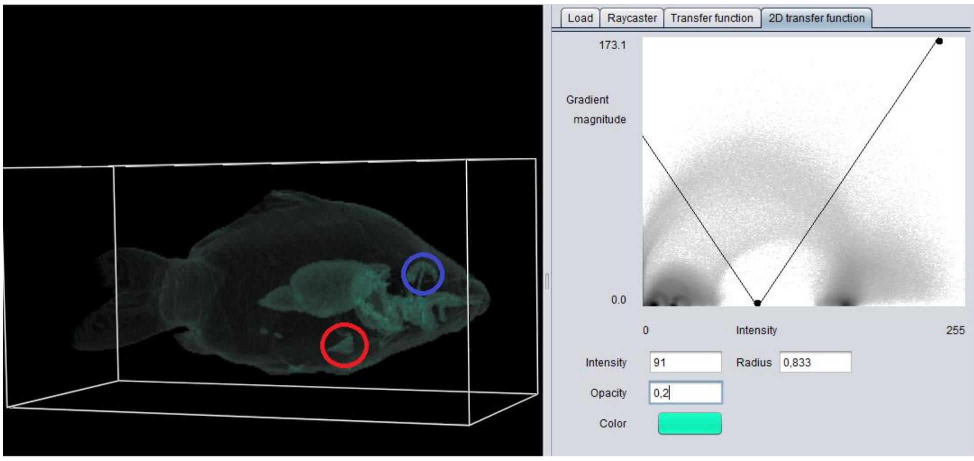

Figure 10: Carp8 dataset 2D Transfer Function representation

Enabling the shading function and looking from a slightly different view at the carp, more tissues are now visible. Conducting research about the anatomy of the carp, it appears that the tissue in the red circle is the heart of the carp and the tissue in the blue circle is the brain of the carp.

Lastly, let us look at the surface of the carp. We analyzed the surface from two different viewpoints. The first one is where the carp is looking to the right and the other one is where the carp is looking to the left. The figures are represented in Figure 11 and Figure 12 respectively.

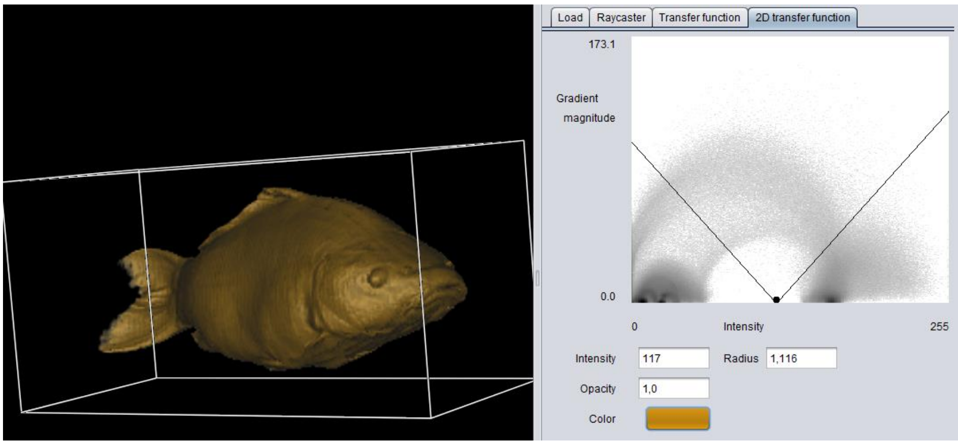

Figure 11: Carp8 dataset 2D Transfer Function representation

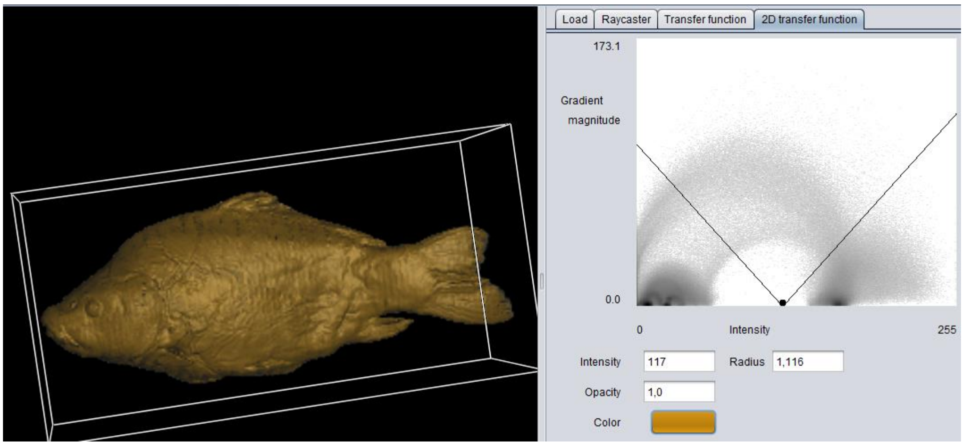

Figure 12: Carp8 dataset 2D Transfer Function representation

In both Figure 11 and Figure 12 the eyes of the carp are clearly visible. In Figure 11 the mouth of the carp is visible and it can be seen that the tail and the fin are curved. The difference between Figure 11 and Figure 12 is that the surface of the carp Figure 12 is not smooth. More contours can be seen in Figure 12 compared to Figure 11, which is interesting and unexplainable.

4.2. Pig8

Slicer and LMIP

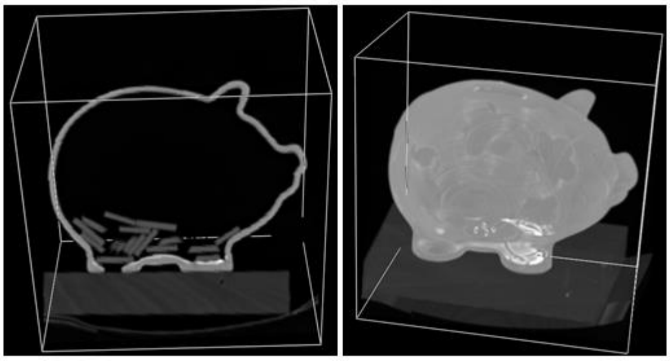
Figure 13: Pig8 dataset, left slicer and right LMIP

Looking at the slice of the pig dataset it is immediately clear that it is indeed a pig standing on a rectangular block. More precisely, it is a piggy bank. It is a piggy bank for the reason that there are coins inside the pig. Now, there is nothing more to explain just by looking with the slicer function. Let us explore more with the LMIP function.

Exploring the dataset with the LMIP function, high intensity values of the dataset are now clearly visible. The first thing that stands out, are the high intensity values at the bottom of the pig on the forelegs. Probably, this is some kind of material like rubber or metal to put some more weight on the front side of the pig so that it stands stable on the thick block. Secondly, there are four pins standing out at the mid-bottom of the pig. Probably, this is the slot where you can open the pig to get your money out. Thirdly, flower patterns on the pig can be observed. At the top of the pig the gap for inserting money is visible as well. Lastly, this LMIP visualization shows us that the block is actually not rectangular, but we observe that it is actually curved like a cot for babies.

Composition

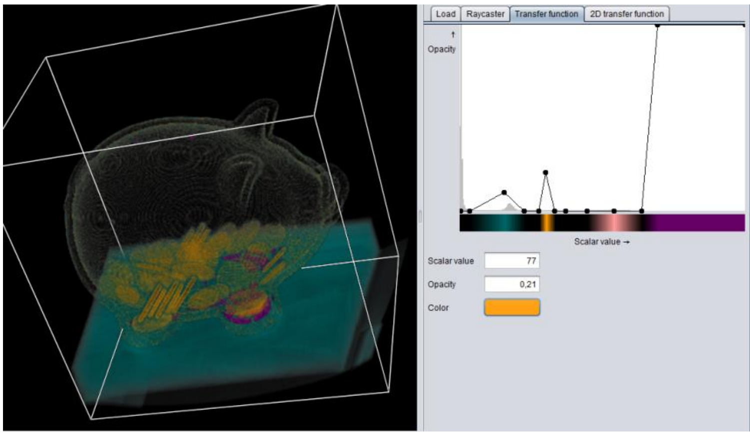

Figure 14: Pig8 dataset composition inside representation

In Figure 14, we visualize the block and the coins separately with different colors for each of them. A trade off has to be made for the opacity of the block and the coins, because if the opacity is set too high, the surface of the pig becomes yellow or blue as well. Furthermore, we colored the special thing on the forelegs of the pig purple. Looking at the histogram on the transfer funtion, this is indeed a different kind of material, because the scalar value is on the right side of the histogram. Probably, this is some kind of a rubber or metal where the pig is attached to the block.

In Figure 15 we visualized the surface fo the pig in pink. Hereby the pattern on the surface becomes more clear.

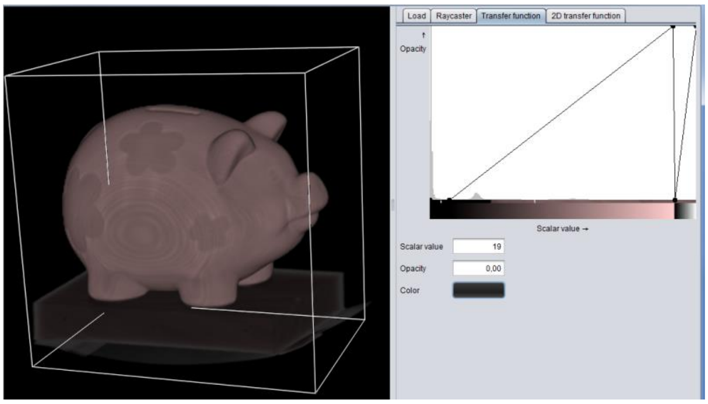

Figure 15: Pig8 composition surface representation

Looking at the surface of the pig, we can also see the flower patterns. It is remarkable that the flowers have a darker color than the rest of the surface. Probably, some difference on the contour and shadowing are present. This can be best explored by the 2D transfer function with the extension of the Phong Shading Model. Next to that, the gap where you can insert the money into the pig is now more visible compared to the LMIP technique. Therefore this object is definitely a piggy bank.

2D Transfer Function with the Phong Shading Model

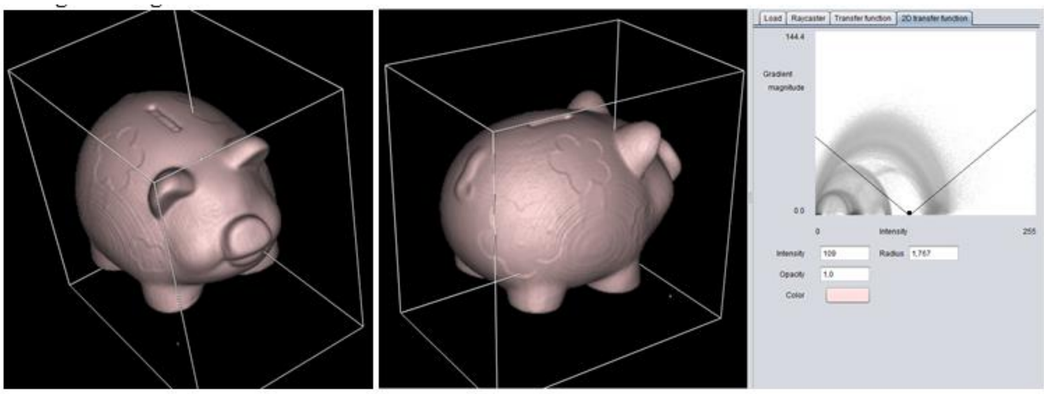

Figure 16: Pig8 dataset 2D Transfer Function

Looking at the piggy bank in Figure 16 we observe the contour surface of the pig is more smoothly. It can be seen that the ears are curved because of the shading and we can see the tail of the pig as well. In addition, it is remarkable that the flower patterns is carved in on the surface.

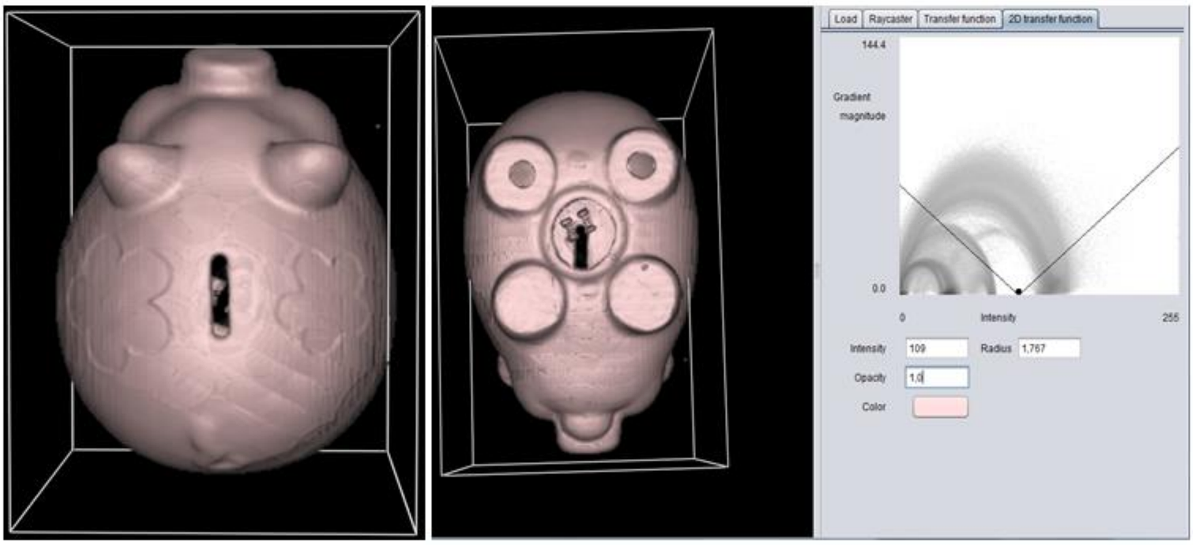

Figure 17: Pig8 dataset 2D Transfer Function

Looking from the top and the bottom to the piggy bank we can see interesting things. Looking from the top through the gap, we can see something which is not very clear. Investigating from the bottom it is clear that it is some kind of slot someone has to push together to open the piggy bank to get money out of it. In addition, the two circles on the hind legs are very remarkable. Probably these are materials where you can stick your piggy bank on the block. However, the forelegs got high intensity on the LMIP raycasting technique which is still not clear. The 2D transfer function without shading is used for further analysis, see Figure 18.

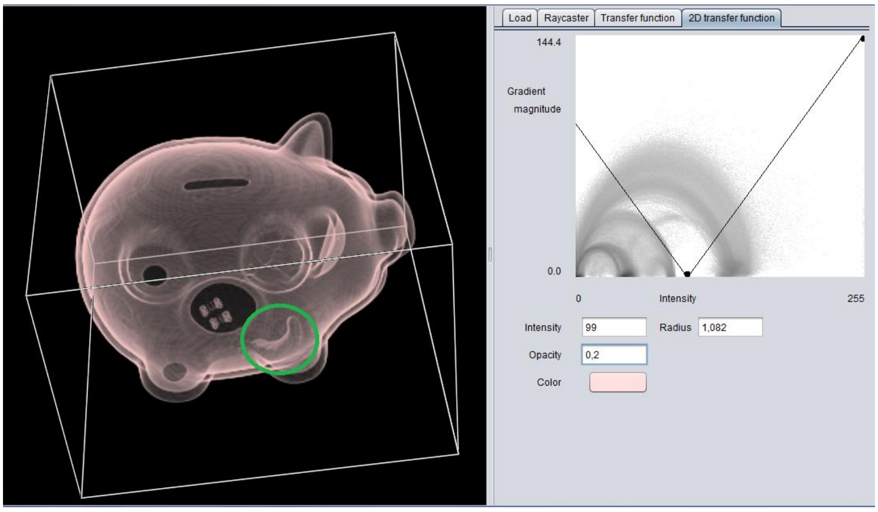

Figure 18: Pig8 dataset 2D Transfer Function without shading

Looking at the bottom of the piggy bank, it is clear that the slot is composed of two strings. So apparently, someone has to push these two strings to open the piggy bank to get the money out of it. There is also some remarkable beany like object in the foreleg of the piggy bank which is encircled in green. Apparently, this thing gives high intensity values which was standing out in the LMIP raycasting technique.

5. Conclusion

Concluding the analysis in this report the slicer is most efficient for fast, simple representation. The more advanced ray casting techniques are requiring a lot more computing power and thereby time. The transfer function for compositing gives the possibility to give different scaler regions different colors. However building these transfer functions is time consuming, trial-and-error and dataset specific. The 2D transfer function gives a better interpretation how to visualize the image. The Phong shading model always improves the image for interpretation, because details become clearer. After all, which ray casting technique to use for visualization depends on the goals and what the analyst wants to see in his visualization.

6. Bibliography

Phong, B. T. (1975). Illumination for computer generated pictures. *Communications of the ACM*, *18*(6), 311–317. http://doi.org/10.1145/360825.360839

Ray, H., Pfister, H., Silver, D., & Cook, T. A. (1999). Ray casting architectures for volume visualization. *IEEE Transactions on Visualization and Computer Graphics*, *5*(3), 210–223. http://doi.org/10.1109/2945.795213

Sato, Y., Shiraga, N., Nakajima, S., Tamura, S., & Kikinis, R. (1998). Local maximum intensity projection (LMIP): A new rendering method for vascular visualization. *Journal of Computer Assisted Tomography*, *22*(6), 912–917. http://doi.org/10.1097/00004728-199811000-00014

Transactions, I., Visualization, O. N., & Graphics, C. (2010). Multidimensional Transfer Functions for Interactive Volume Rendering, *8*(3).