2IV35 Visualization Set 2

Jeroen van Oorschot Student number: 0721913
j.v.oorschot@student.tue.nl

January 11, 2015

Contents

1 Volume Rendering **3**
1.1 Tri-linear interpolation . . . 3
1.2 Gaining speed . . . 3

2 Maximum intensity projection **3**
2.1 Backpack . . . 3
2.2 Bonsai . . . 4
2.3 Carp . . . 4
2.4 Orange . . . 5
2.5 Pig . . . 5
2.6 Human . . . 6
2.7 Tomato . . . 6
2.8 Tooth . . . 7

3 Transfer function **8**
3.1 Carp . . . 8

4 Compositing **8**
4.1 Backpack . . . 8
4.2 Bonsai . . . 8
4.3 Carp . . . 8
4.4 Orange . . . 8
4.5 Pig . . . 8
4.6 Human . . . 8
4.7 Tomato . . . 8
4.8 Tooth . . . 8

5 Opacity weighting **9**
5.1 Backpack . . . 9
5.2 Bonsai . . . 9
5.3 Carp . . . 9
5.4 Orange . . . 10
5.5 Pig . . . 11
5.6 Human . . . 12
5.7 Tomato . . . 12
5.8 Tooth . . . 13

1 Volume Rendering

1.1 Tri-linear interpolation

To make sure we can also get data from between points we want to do some interpolation on the data. So I use the following function to get a value on point (x, y, z) in a field F with the value $F[i][j][k]$ for the integer values i, j and k by calculating the value val with:

$$\begin{aligned}
\alpha = \quad & x - \lfloor x \rfloor \\
\beta = \quad & y - \lfloor y \rfloor \\
\gamma = \quad & z - \lfloor z \rfloor \\
val = \quad & (1-\alpha) * (1-\beta) * (1-\gamma) * F[\lfloor x \rfloor][\lfloor y \rfloor][\lfloor z \rfloor] \\
& +\alpha * (1-\beta) * (1-\gamma) * F[\lceil x \rceil][\lfloor y \rfloor][\lfloor z \rfloor] \\
& +(1-\alpha) * \beta * (1-\gamma) * F[\lfloor x \rfloor][\lceil y \rceil][\lfloor z \rfloor] \\
& +\alpha * \beta * (1-\gamma) * F[\lceil x \rceil][\lceil y \rceil][\lfloor z \rfloor] \\
& +(1-\alpha) * (1-\beta) * \gamma * F[\lfloor x \rfloor][\lfloor y \rfloor][\lceil z \rceil] \\
& +\alpha * (1-\beta) * \gamma * F[\lceil x \rceil][\lfloor y \rfloor][\lceil z \rceil] \\
& +(1-\alpha) * \beta * \gamma * F[\lfloor x \rfloor][\lceil y \rceil][\lceil z \rceil] \\
& +\alpha * \beta * \gamma * F[\lceil x \rceil][\lceil y \rceil][\lceil z \rceil]
\end{aligned}$$

1.2 Gaining speed

We implemented two ways to speed up the program, one is by making a low and a high resolution maximum intensity projection(MIP), in the lower resolution version (simply called MIP) we start at one and take steps of two in calculating pixel colour, while setting pixels (x, y),$(x-1, y)$,$(x, y-1)$ and $(x-1, y-1)$ to that color. We also added the option to manually adjust the number of samples. Results of tests on the backpack dataset can be found in this table, the numbers represent the rendering time in ms:

samples	hi	low
223	11904	2933
22	1280	372

2 Maximum intensity projection

In maximum intensity projection (MIP) we traverse trough the data and project at the point where that ray hits the screen the maximum of the samples on that ray. After implementing maximum intensity projection we opened all data sets with to get the following results.

2.1 Backpack

This worked fine, it was easy to see there were objects resembling a bullet, a drug needle, a Swiss army knife and a spray can in the backpack, so this technique would be suitable on an airport, because we can pick out the backs worth examining. Without rotating the data the contents can be obscured, but after rotating you could get to see everything clearly as can be seen in Figure 1.

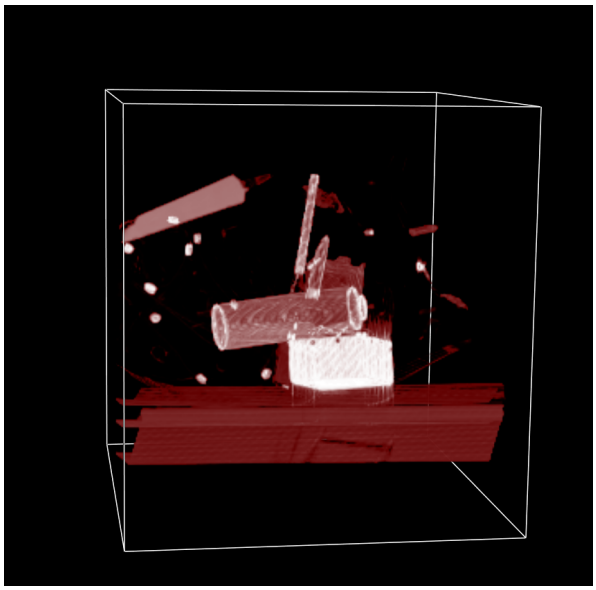

Figure 1: Resulting picture for backpack with MIP

2.2 Bonsai

This worked fine, we could see there was a tree. A picture is included in Figure 2.

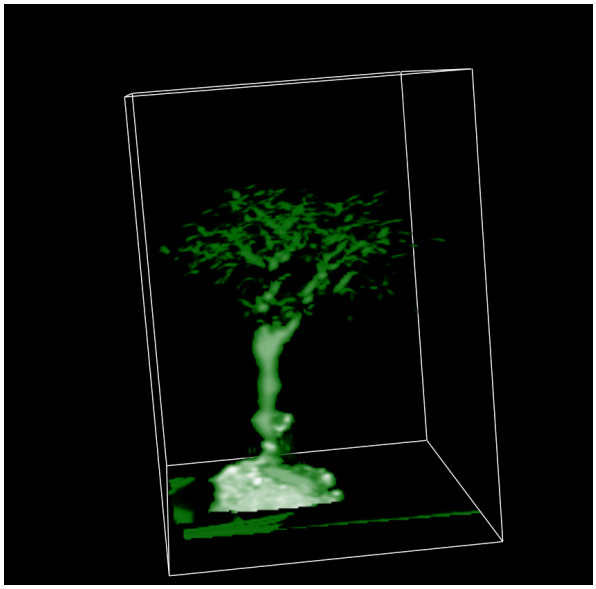

Figure 2: Resulting picture for Bonsai with MIP

2.3 Carp

This worked fantastic, we could see all the bones in the carp. A picture of the tail is included in Figure 3, the side is in Figure 4.

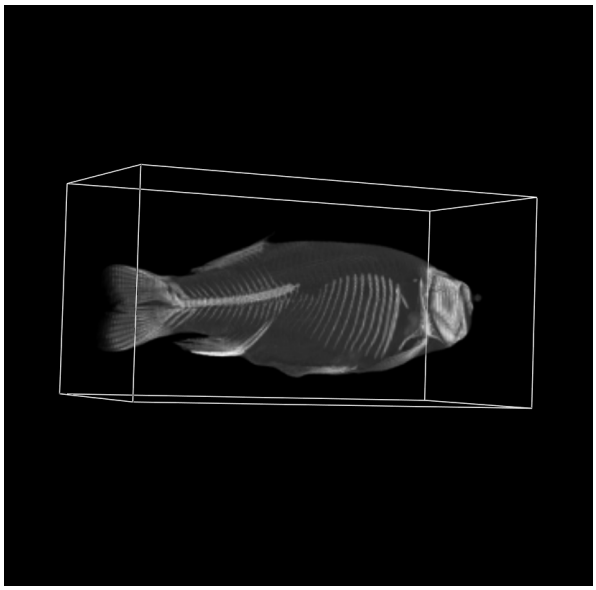

Figure 3: Resulting picture for tail of the carp with MIP

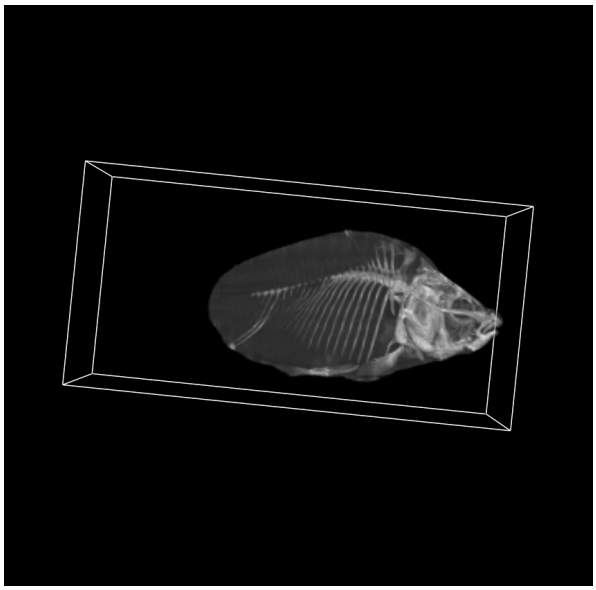

Figure 4: Resulting picture for side of the carp with MIP

2.4 Orange

This dataset leant itself to MIP too, we even were able to give the peel an orange color, while giving the inside a blue color. A picture is included in Figure 5.

2.5 Pig

In this example we were able so show the coins in yellow and the pig in pink, also the hole is visible. A picture is included in Figure 6.

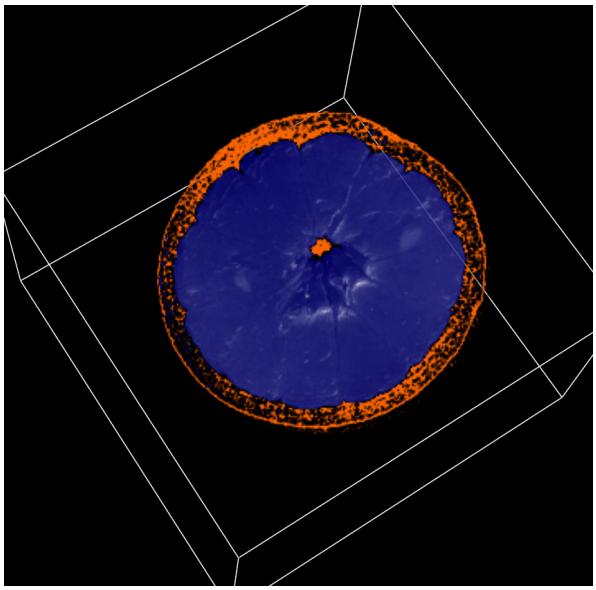

Figure 5: Resulting picture for orange with MIP

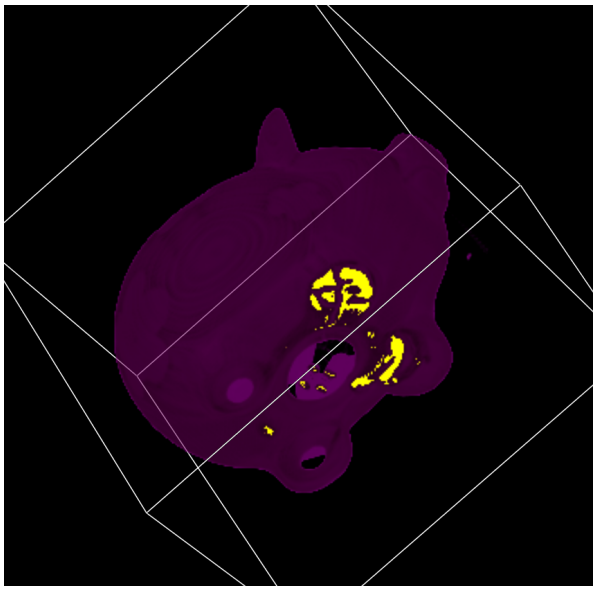

Figure 6: Resulting picture for pig with MIP

2.6 Human

Again a clear picture of a skeleton. A picture is included in Figure 7.

2.7 Tomato

The tomato did not look very good as a picture because its quite homogeneous and quite similar to the orange, therefor we decided to omit a picture of it.

Figure 7: Resulting picture for human with MIP

2.8 Tooth

The tooth data also produced nice results as can be seen in Figure 8.

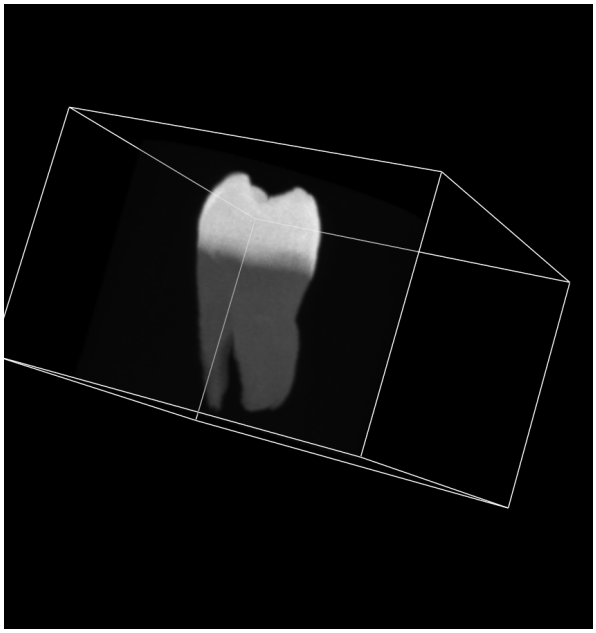

Figure 8: Resulting picture for tooth with MIP

3 Transfer function

In the transfer function we made at every sample i the color C_i is calculated, but there is also some color C_{i-1} added behind it. This is done through the function $C_i = C(i) + (1 - \tau)C_{i-1}$.

3.1 Carp

Because its more dificult to find the correct colours we included a single picture, Figure 21. It is nice to see the difference between the different tissues, we can see the green skin, the yellow brains, the red muscles and the grey bones.

4 Compositing

For compositing we implemented the standard formula given in slides of the course 2IV35 (Visualization) of the TU/e [**?**]. Compositing is a process where the values of all voxels along the ray are merged in such a way that one should be able to also look at the insides, where masses which are big will have a heavier impact on the picture then smaller masses. This can be remedied by setting the transfer function carefully but it could be that the transferfunction needs to be readjusted after rotating the picture, since now the ray has to traverse more or less of the material then it did previously.

4.1 Backpack

4.2 Bonsai

4.3 Carp

4.4 Orange

4.5 Pig

4.6 Human

4.7 Tomato

4.8 Tooth

5 Opacity weighting

Opacity weighting is implemented as discribed in a paper by Marc Levoy [**?**]. Since there is a need for scaling the gradient magnitude, we implemented a scaling factor using a spinner, which can be used to scale the magnitude. We decided to use this method since it enables a person to have granulated control of the scaling which enables the selection of optimal values for making the borders visible. Opacity weighting remedies the problem of compositing since it voxels which define edges will be made much more visible then voxels in the middle of (homogenuous) masses. This enables more precise viewing of contents of volumes.

5.1 Backpack

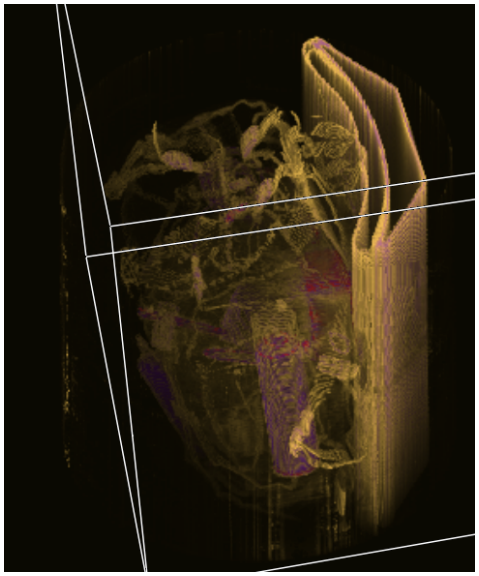

Figure 9: Backpack similar to composition

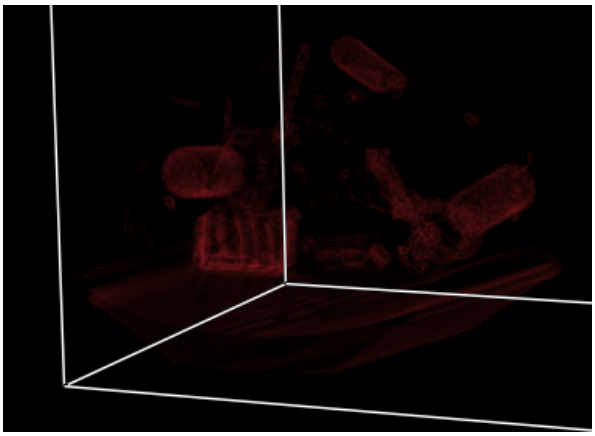

Figure 10: Backpack with a look into the contents

5.2 Bonsai

5.3 Carp

When we applied opacity weighting to the carp dataset we could very quickly see that the carp was laid flat on some sheets which is presumable paper

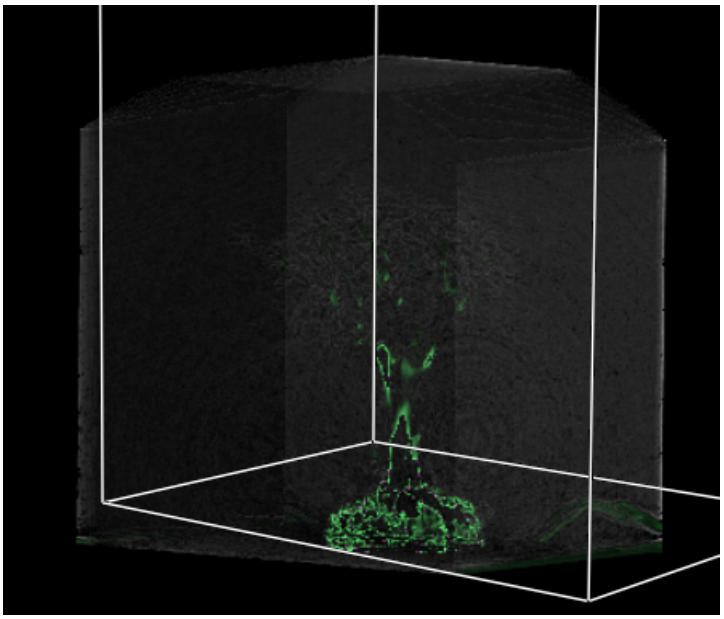

Figure 11: The bonsai with the protective covering slightly visible

5.4 Orange

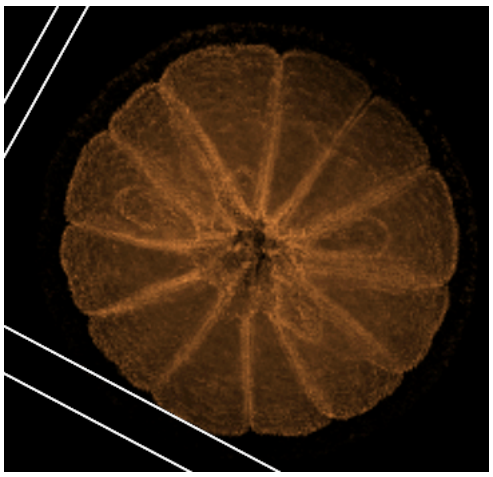

Figure 13:

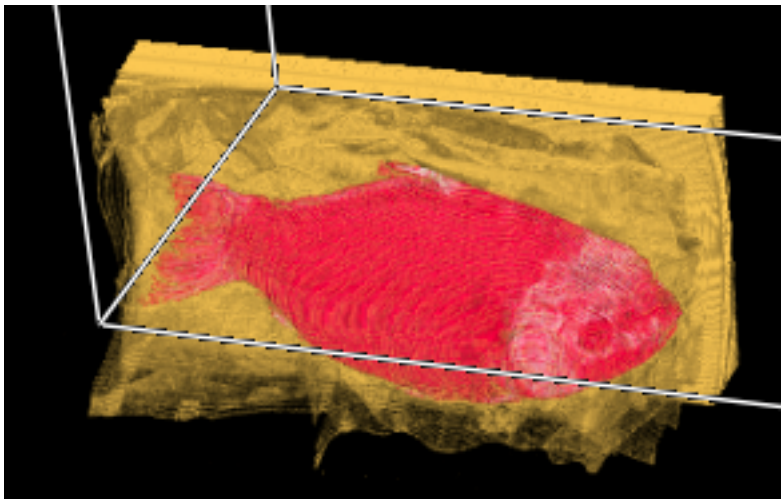

Figure 12: The carp with the protective surface visible

5.5 Pig

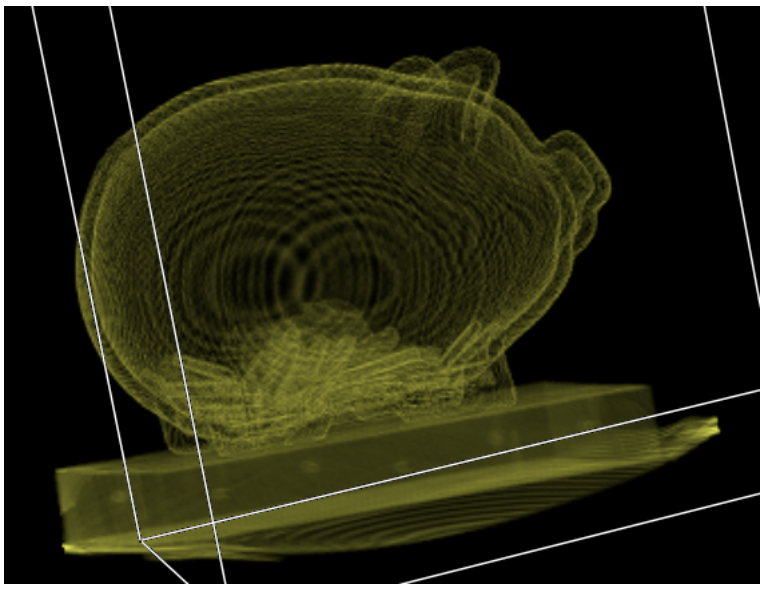

Figure 14: The piggybank with the coin contents made visible

5.6 Human

Figure 15:

5.7 Tomato

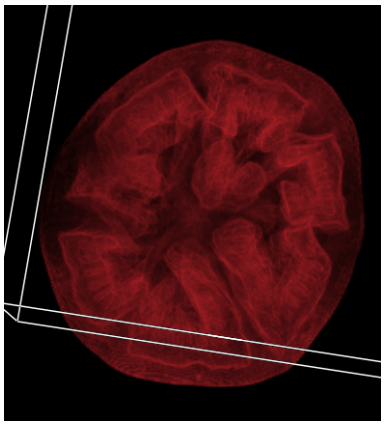

Figure 16: Mixture of the pulp, the skind and water

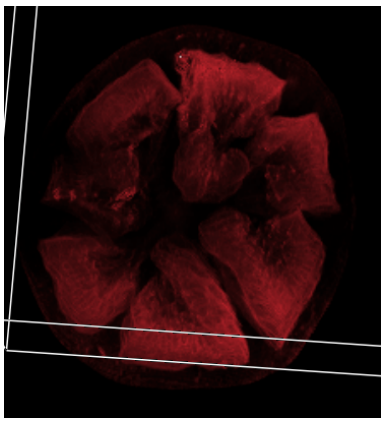

Figure 17: Only the pulp visible.

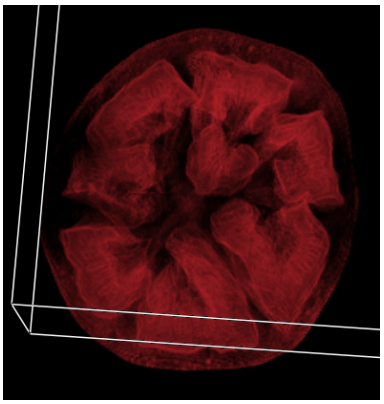

Figure 18: Tomato with some seeds visible

5.8 Tooth

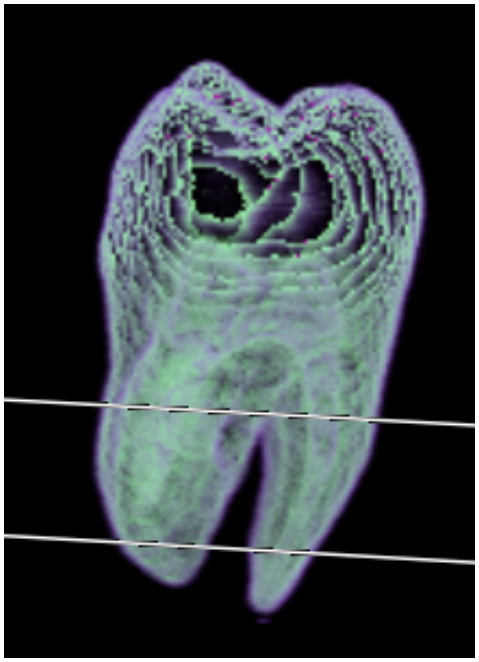

Figure 19: The tooth with the root

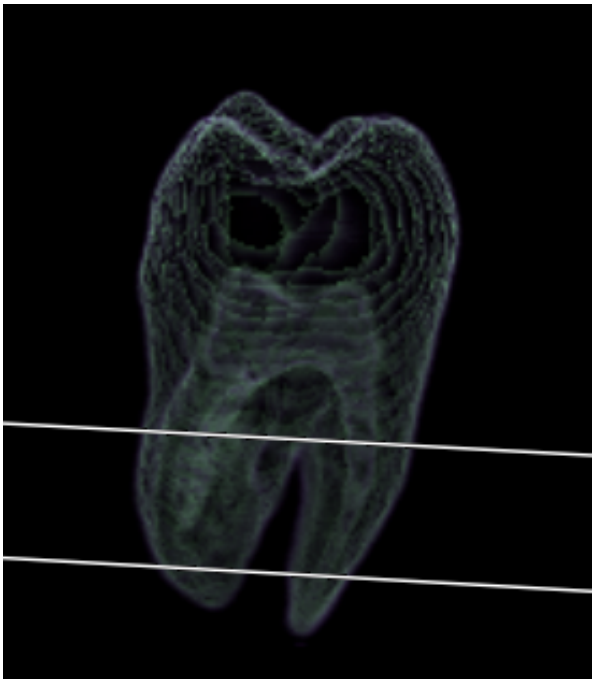

Figure 20: Tooth with a more visible root

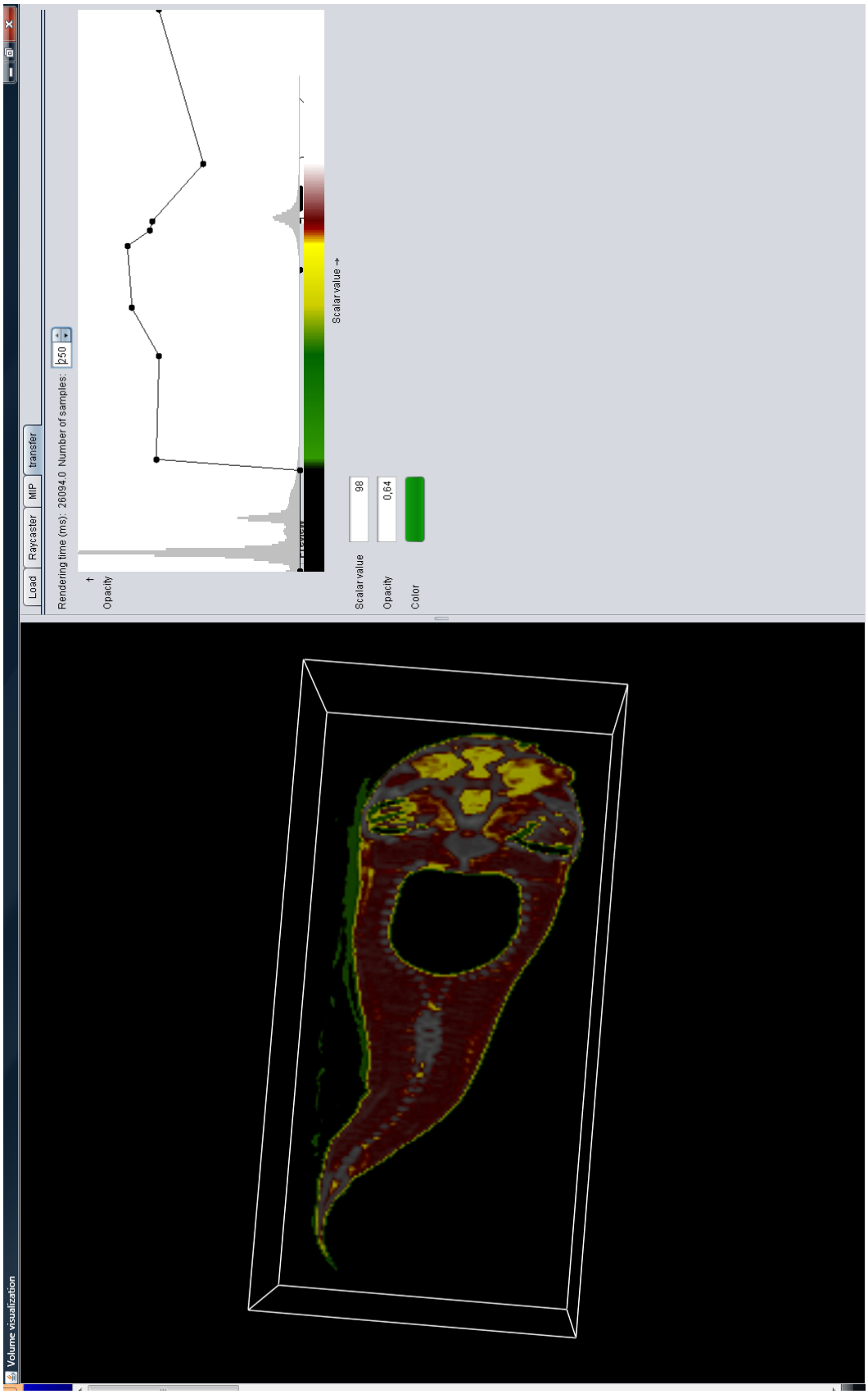

Figure 21: Resulting picture for the carp with transfer function