ISSN: 1803-3814 (Printed), 2571-3701 (Online)
https://doi.org/10.13164/mendel.2022.2.083

Mathematical Methods for 3D Reconstruction of Cell Structures

Dalibor Martišek

Institute of Mathematics, Faculty of Mechanical Engineering,Brno University of Technology, Czech Republic
martisek@fme.vutbr.cz

Abstract

The study of the complicated architecture of cell space structures is an important problem in biology and medical research. Optical cuts of cells produced by confocal microscopes contain a lot of information, however, most of this is unsubstantial for human vision. Therefore, it is necessary to use mathematical algorithms for the visualization of such images. Present software tools such as OpenGL or DirectX run quickly in a graphic station with special graphic cards, run very unsatisfactory on PC without these cards and outputs are usually poor for real data. These tools are black boxes for a common user and make it impossible to correct and improve them. With the method proposed, more parameters of the environment can be set. The quality of the output is incomparable to the earlier described methods and is worth increasing the computing time. We would like to offer mathematical methods of 3D scalar data visualization describing new algorithms that run on standard PCs very well.

Keywords: *Cell structure, 3D reconstruction, Scalar data, Direct volume rendering, Illuminant model, Line segment rasterization.*

Received: 10 May 2022
Accepted: 05 December 2022
Online: 19 December 2022
Published: 20 December 2022

1 Introduction

Cells are obviously observed using a fluorescent scan confocal microscope. Data images acquired by this microscope are scalar fields from a mathematical point of view. Algorithms for their three-dimensional (3D) processing can be classified into two groups: surface-fitting algorithms (SF algorithms) and direct volume rendering algorithms (DVR algorithms). SF algorithms construct a geometric surface representation of the scalar field to be displayed and then construct this surface. DVR algorithms display the scalar field directly without surface representation.

The existing software tools used to reconstruct small objects usually do not employ all the features of state-of-the-art hardware. Since the existing software is run on high-level computers, the efficiency of the algorithms is not the issue. Technologies such as OpenGL or DirectX can only be used on graphic stations with special graphic cards with graphics accelerators. The software programs recommended by microscope manufacturers usually do not work on standard PCs. Data visualization based on OpenGL or DirectX does run on a PC, but only with great difficulties.

The main problem is the quality of output created by means of OpenGL and DirectX. With the method proposed, more parameters of the environment can be set, making it possible to apply 3D filters to set the output image sharpness in relation to the noise. The quality of the output is incomparable and is worth increasing the computing time.

For the study of the complicated architecture of 3D structure of cells, we have more and more quality instruments available but the needed software support falls behind slightly. Most of the programs provided with these instruments are not able to master the 3D object reconstruction at an appropriate level and, if they are, companies do not publish any applied methods or algorithms for commercial reasons. Therefore very little information can be found on these methods.

We would like to offer mathematical methods of 3D scalar data visualization describing new algorithms that run on standard PCs very well. Some publications ([8, 10, 11, 15, 1]) were some of the first attempts in this direction.

2 Current Methods

2.1 Primary Data

The primary data used in this paper have been provided by the Olympus Fluoview 1000 microscope by Prof. Roman Janisch of the Department of Biology, Faculty of Medicine, Masaryk University, Brno, Czech Republic and Prof. Josef Reishig of the Institute of Biology, Faculty of Medicine, Charles University in Plzeň, Czech Republic. A confocal microscope output is formed by a series of (usually a few tens of) optical cuts through an examined object and our ambition is to reconstruct the object on the basis of these cuts: twodimensional-ly and especially three-dimensionally. Biology and medical research use the fact that confocal microscopy is basically a noninvasive and nondestructive kind of study of the space structure of cells and tissues.

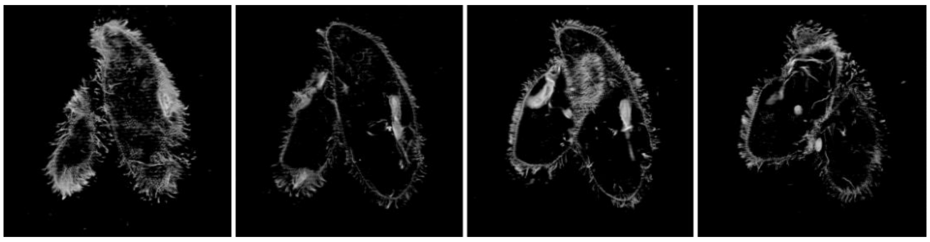

Figure 1: Optical cuts of Paramecium caudatum cells. The 4-th, 7-th, 11-th and 15-th cut of eighteen optical cuts has been choosen (8 bits grey scale, resolution of 490×490 pixels in each cut). Data provided by Prof. Roman Janisch.

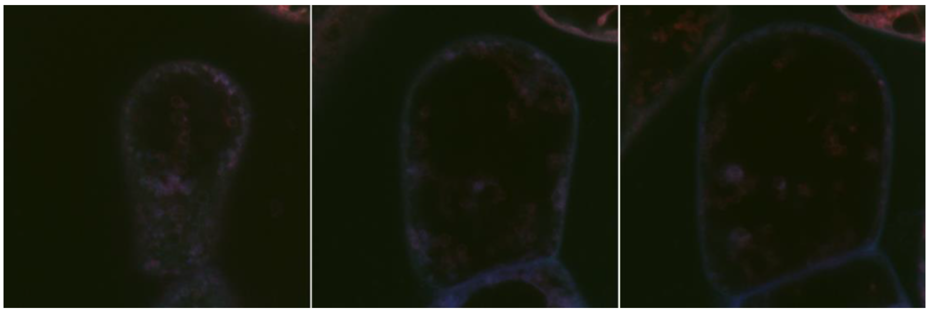

Figure 2: Optical cuts of Tobaco cells. The 20-th, 50-th and 80-th cut of one hundred optical cuts has been choosen (RGB true color, resolution of 800×800 pixels in each cut). Data provided by Prof. Josef Reichig.

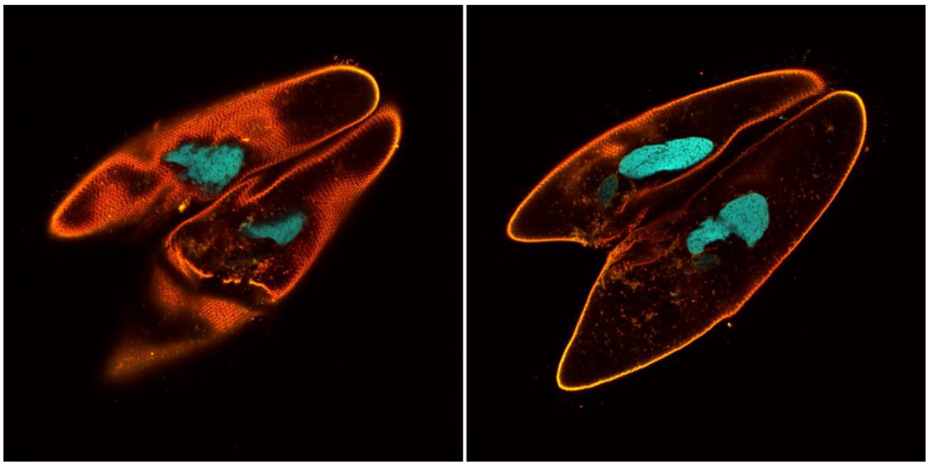

Figure 3: Optical cuts of Paramecium caudatum cells. The 20-th and 40-th cut of 56 optical cuts has been chosen (RGB true color, resolution of 1600×1600 pixels in each cut). Data provided by Prof. Josef Reischig.

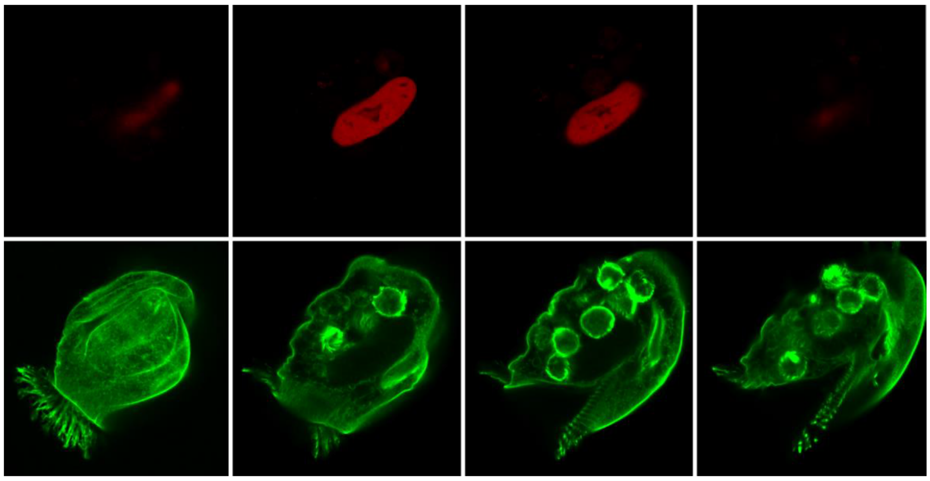

Figure 4: Optical cuts of Euplotes patela cells. The 7-th, 13-th, 18-th and 23-th cut of 27 optical cuts has been chosen (pseudocolors: 27 cuts of nucleus – red, 27 cuts of other structures - green; 16 bits grey scale, resolution of 450 ×460 pixels in each cut). Data provided by Prof. Roman Janisch.

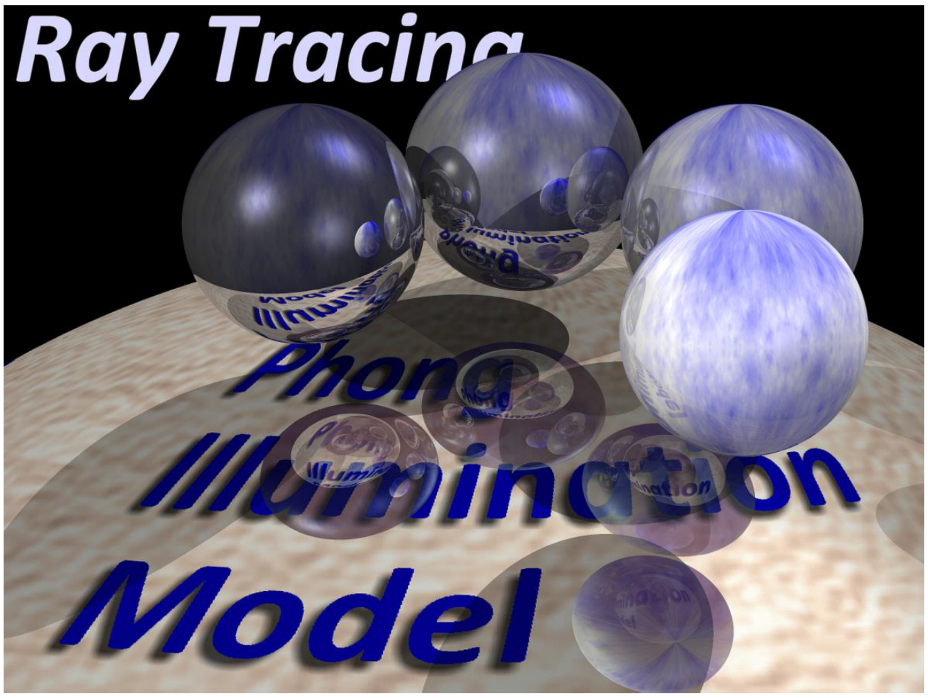

Figure 5: Output of the Ray Tracing algorithm where each point of the set has been evaluated by Phong Illumination Model. Recursion depth is five. Software was developed by the author of this paper.

In Fig. 1, we can see optical cuts of Paramecium caudatum cells (18 Images, each of them 8 bits grey scale, resolu-tion of 490×490 pixels in original data). In Fig. 2, there are illustrated optical cuts of Tobacco cell (100 optical cuts, RGB true color, 800×800 pixels).

Furthermore, Paramecium caudatum (56 optical cuts, RGB true color, 1200×1200 pixels) in Fig. 3 and Euplotes Patela cell (2×27 optical cuts, nucleus in red, other structures in green, color depth 16 bits, resolution of 450×460 pixels of each cut in original data (Fig. 4).

2.2 Current Common Principles of 3D Data Visualization

In computer graphics, either vector or raster data can be processed. In terms of graphic data, a vector is understood in the traditional sense, i. e. it is understood as an oriented line segment with an initial and end point.

While displaying vector data, we usually work with illumination as well. Light illuminates a scene (the space displayed) in a certain direction. If a planar optical interface had a microscopically ideal surface then both reflection and refraction would preserve parallelism. However, a real body does not have a totally smooth surface so both reflected and refracted beams have various directions. Asperities have a fractal character and the properties of reflected and refracted beams can be described only approximately.

The first (and the simplest) illumination model for n light sources was published by Bui Tuong Phong [4]. It is an empirical model of local illumination which combines the light direction L_m of m-th light source, I_m its intensity, normal vector N_P in calculated point P and direction V_P from which point P is observed. Furthermore, ambient intensity I_a, specular (s), diffuse (d) and ambient (a) part of the reflection and also its "shininess" (h) is estimated. These parameters gives following equation for illumination intensity I_P in the surface point P:

$$\begin{aligned} I_p = {} & a \cdot I_a + \sum_{m=1}^{n} L_m \cdot N_P \cdot I_m \cdot d + \\ & \sum_{m=1}^{n} (V_p \cdot (2 \cdot (N_p \cdot L_m) \cdot N_P - L_m))^h \cdot I_m \cdot s \end{aligned} \tag{1}$$

It is possible to achieve a realistic scene by combining this model with so called ray tracing technique. A light ray is sent through each pixel of output window and all optic interactions with group of surfaces and solids are recursively tracked. In Fig. 5, we can see the set of five balls with different optical properties illuminated by three light sources with intensities $I_1 = I_2 = 0.25$; $I_3 = 0.2$, ambient intensity is $I_a = 0.3$. For the big ball, it is: $s = 0.7$; $d = 0.2$; $a = 0.1$. For the smaller balls is (from the left): $s_1 = 0.8$; $d_1 = 0.15$; $a_1 = 0.05$; $s_2 = 0.6$; $d_2 = 0.3$; $a_2 = 0.1$; $s_3 = 0.4$; $d_3 = 0.4$; $a_3 = 0.2$; $s_4 = 0.1$; $d_4 = 0.4$; $a_4 = 0.5$.

Algorithms that process a scalar field can be classified into two groups: surface fitting algorithms SF algorithms) and direct volume rendering algorithms (DVR algorithms). SF algorithms construct a geometric surface representation of the scalar field to be displayed, and then construct this surface. DVR algorithms display the scalar field directly without surface representation.

Raster data do not contain any “vector” information on saved objects. From these data, it is not possible to find out in a simple way whether an image is composed of cubes, spheres or something else. A file contains information on the size of the image, on the way it may have been compressed and on the colour encoding used. The image itself is saved as a matrix and each element of this matrix represents one image pixel. Most of the displaying devices (monitors, printers, cameras, etc.) work on this principle.

Data provided by a confocal microscope are also raster. Separate optical cuts represent different height levels of the observed object. Therefore, its series provided 3D data grid – see Fig. 6.

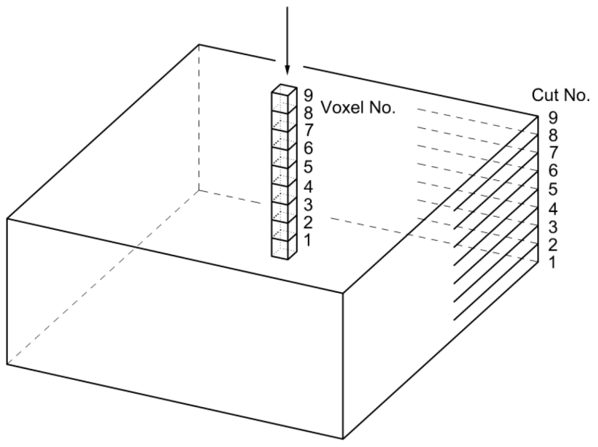

Figure 6: 2D processing of a series of optical cuts.

2D processing of this grid can be done in several ways. We can use the First intensity projection algorithm (FIP) which finds the first non-zero intensity in separate columns created by corresponding pixels, Maximum intensity projection algorithm (MIP) displays the maximum intensity in the processed column. Summed intensity projection algorithm (SIP) displays the sum intensity along the column and the average intensity projection algorithm (AIP) displays the arithmetic mean of intensity in the column.

Most of the image order methods are based on the approximation of volume rendering integral (VRI) which considers volume as participating medium. Participating medium is a cloud consisting of small particles where each particle can absorb, emit or scatter light.

The idea of volume rendering integral was first described by Blinn [5]. Infinitesimal particles are considered as intensity emitters as well as attenuation source

$$\frac{dI}{ds} = g(s) - \tau(s)I(s) \tag{2}$$

where s is a length parameter, $g(s)$ is emission intensity at distance s, $I(s)$ is light intensity at distance s. The coefficient $\tau(s)$ is called the extinction coefficient and expresses the rate of decline of light intensity along a ray.

For intensity $I(s_1)$ at point s_1, solution of (2) is

$$I(s_1) = I_0 e^{-\int_{s_0}^{s_1}\tau(t)dt} + \int_{s_0}^{s_1} g(s)e^{-\int_{s}^{s_1}\tau(t)dt}ds \tag{3}$$

or alternatively for $I(s_0)$

$$I(s_0) = I_1 e^{-\int_{s_1}^{s_0}\tau(t)dt} + \int_{s_1}^{s_0} g(s)e^{-\int_{s}^{s_0}\tau(t)dt}ds \tag{4}$$

We have to approximate these integrals. We employ discrete Riemann sum over the casted ray with discrete samples at distances spaced apart by Δs. By such discretization of (3), we get

$$\begin{aligned} I(s_1) &= I_0 e^{-\int_{s_0}^{s_1}\tau(t)dt} + \int_{s_0}^{s_1} g(s)e^{-\int_{s}^{s_1}\tau(t)dt}ds \\ &\approx I_0 e^{-\sum_{i=1}^{n}\tau(i\Delta s)\Delta s} + \sum_{i=1}^{n} g(i\Delta s)e^{-\sum_{j=i+1}^{n}\tau(j\Delta s)\Delta s} \\ &= I_0 \prod_{i=1}^{n} e^{-\tau(i\Delta s)\Delta s} + \sum_{i=1}^{n} g(i\Delta s)\prod_{j=i+1}^{n} e^{-\tau(j\Delta s)\Delta s} \end{aligned}$$

where $n = |s_1 - s_0|/\Delta s$. If we denote $T(s) \approx t(i\Delta s) = t_i$ and $g(i\Delta s) = g_i$, we obtain

$$I = I_0 \cdot \prod_{i=1}^{m} t_i + \sum_{i=1}^{m}\left(v_i \prod_{j=i+1}^{m} t_j\right) \tag{5}$$

(Back-to-Front algorithm - BtF) or from (4) alternatively

$$I = I_0 \cdot \prod_{i=1}^{m} t_i + \sum_{i=1}^{m}\left(v_i \prod_{j=1}^{i} t_j\right) \tag{6}$$

(Front-to-Back algorithm- FtB).

These algorithms are able to simulate the transparency of an object: $v_i$are values of voxels; t_i its transparencies and I_0 is the intensity of ambient light.

2.3 Projection

For 3D visualization, we can relatively easily use a parallel projection method which simulates a long-distance camera, and centre projection methods to model a camera situated closer to the object. In projective geometry, each point X in 3D space is described by a quartet of homogenous coordinates $X = (x; y; z; \omega)$ where $\omega = 0$ if and only if the point X is at infinity (so called non-eigen point), $\omega \neq 0$ in other cases (X is a “classical” point). The projection of the point X to the point X' is described as

$$\begin{pmatrix} x' \\ y' \\ z' \\ \omega' \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ m_{41} & m_{42} & m_{43} & m_{44} \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \\ \omega \end{pmatrix} \tag{7}$$
$$X'^T = MX^T$$

where M is a projection matrix. There is $m_{44} \neq 0$ and $m_{41} = m_{42} = m_{43} = 0$ in case of a parallel projection, $m_{44} \neq 0$ and $m_{41} \neq 0$ or $m_{42} \neq 0$ or $m_{43} \neq 0$ in case of a centre projection.

2.4 3D Processing of the Raster Data - SF Algorithms

Vector visualization algorithms are, by default, used for 3D vector data, but they can be used for raster data as well (including confocal microscopes data).

One of the most known algorithms which solved this problem is called the Marching Cubes Algorithm. It proceeds through each cube, then determines the polygon(s) needed to represent the part of the isosurface that passes through this cube. This is done by evaluation of each cube vertex by value $2^n, n = 0, 1, \dots, 7$ (see Fig. 7). If the scalar's value in n-th vertex is higher than the iso-value then 2^n is added to a counter. Its final value determines the polygon which needs to be constructed. In Fig. 7; the counter value is $2^0 + 2^1 + 2^3 + 2^4 + 2^5 = 59$. Note that there are 256 possibilities in all, however, this number can be reduced to fifteen using symmetries.

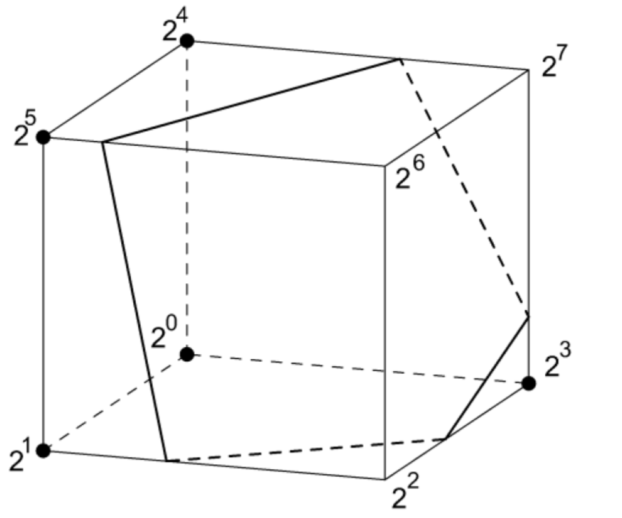

Figure 7: Principle of the Marching Cubes Algorithm.

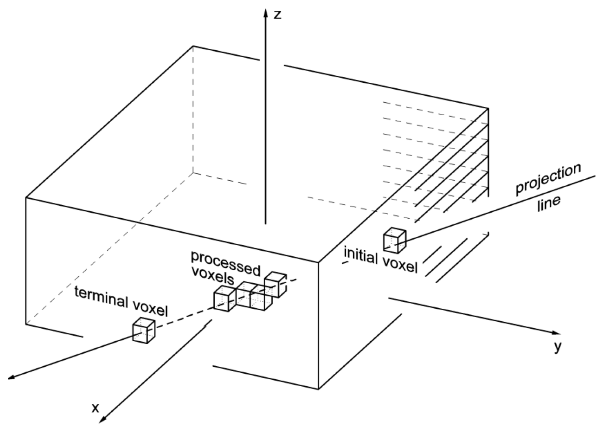

Figure 8: 3D generalization of 2D processing of the optical cuts series.

2.5 3D Processing of the Raster Data - DVR Algorithms

For a DVR algorithm, it is necessary to generalize 2D processing of 2D grid which is shown in Fig. 6. Its 3D generalization is illustrated in Fig. 8. The coordinates of a point where a projection line enters a system of cuts (initial voxel), are known and the coordinates of a point where the projection line leaves the system of cuts (terminal voxel), are known as well (see Fig. 8). It is necessary to find out which voxels lie on the line segment bounded by the input and output. However, planar Bresenham's algorithm deals exactly with this situation: the coordinates of the basepoint and endpoint of a line segment are known and we color the pixels lying between them. So it is necessary to generalize this algorithm for the 3-D space.

In literature, this procedure is often called the rasterization of a line segment. We can calculate the intersection of the projection line with the surface of each voxel – we can call it an intersection algorithm (IA). However, this process is very time-consuming. A fast graphic algorithm for 2D line construction comes from Bresenham — see [2] or [4]. Some generalizations of this algorithm for the 3D space are known as well — see [4, 3, 13, 9] for example. After 3D generalization, all methods mentioned the previous section (FIP, MIP, SIP, AIP, BtF, FtB) can be used as DVR algorithms.

In Fig. 9 – 12, a brief overview of the results of existing methods of cell structure visualization is illustrated. In Fig. 9, we can see the 3D processing of raster data from Fig. 1 using the Marching Cubes algorithm. Two cells of infusorian Paramecium caudatum with overlapped anterior, detail (micronucleus) were chosen by the rectangle on the 2D reconstruction using AIP (left) and MIP (right) in pseudocolors.

Fig. 10 illustrated DVR 3D processing of the data from Fig. 2. High noise in these data is a great problem for current methods. Noise can be obviously reduced by appropriate thresholding, however, in this case, this technique is not very successful. To low threshold implies drowning of the data in the brutal noise (on the left), slightly higher number results in a loss of useful information (on the right). Fig. 11 illustrated 3D visualization of raster data from Fig. 3.

Although the input data is of high resolution (1600×1600 pixels), the quality of the output (especially of the detail – cell nucleus on the right) is not very good.

In Fig. 12, 3D processing of data from Fig. 4. Due to low method resolution, the quality of the output (especially of the details - micronucleus) is poor, due to low input data resolution, and artifact grid structure is visible. The intersection algorithm (IA) was used for projection into 2D in Fig. 10, 11, and 12.

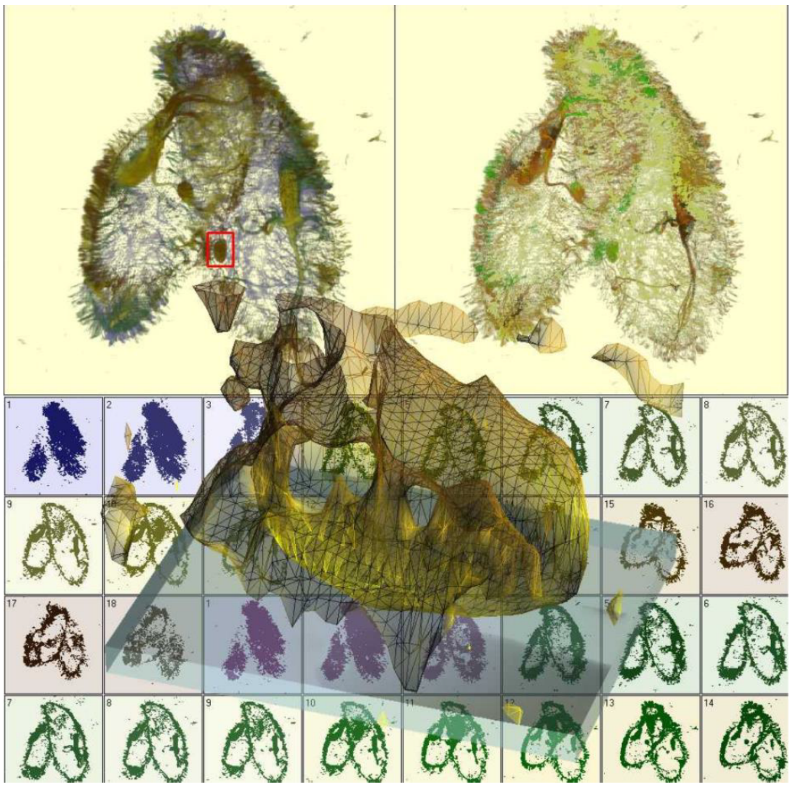

Figure 9: SF 3D processing of raster data from Fig. 1 using Marching Cubes algorithm. The central projection with viewing angle 25° and Phong illuminant model with one light source are used.

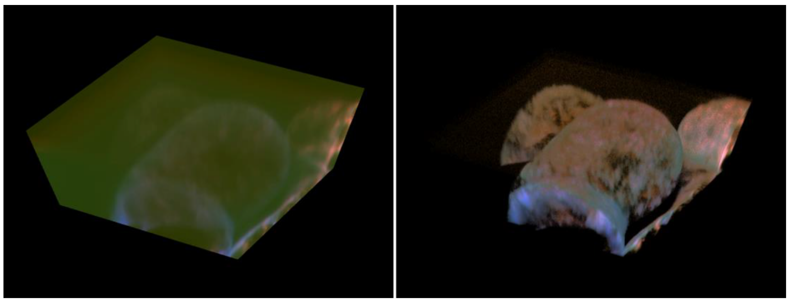

Figure 10: DVR 3D processing of raster data from Fig. 2 using FtB algorithm. The central projection with viewing angle 35°. The images differ in threshold value – 10 on the left, 20 on the right.

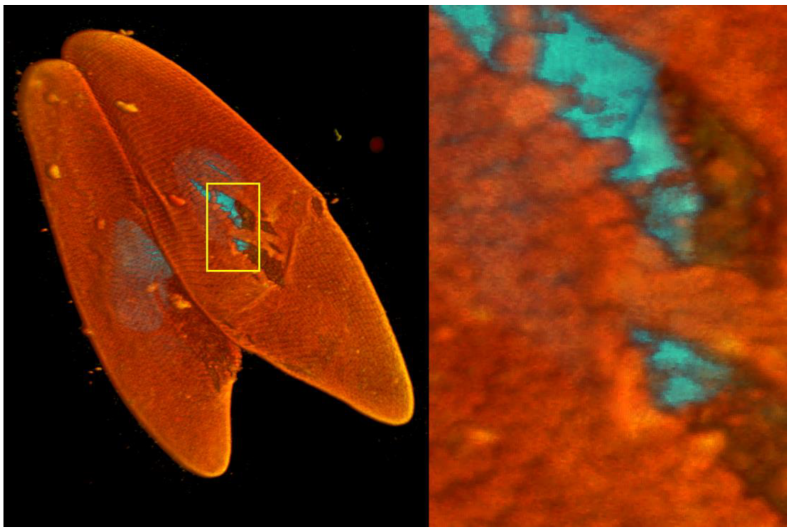

Figure 11: DVR 3D processing of raster data from Fig. 3 using common FtB algorithm. The central projection with viewing angle 10° is used.

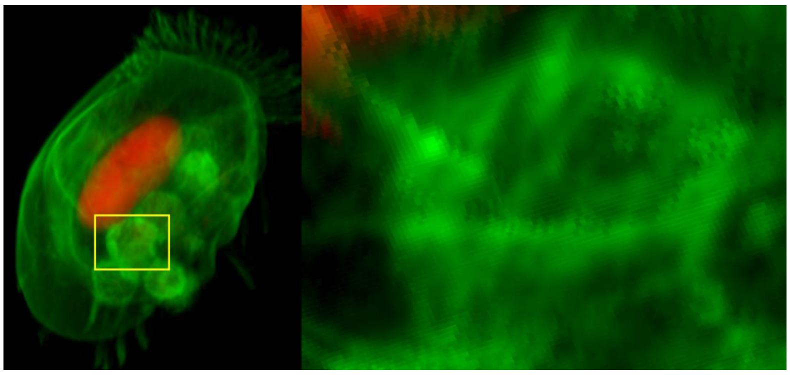

Figure 12: DVR 3D processing of raster data from Fig. 4 using common BtF algorithm. The central projection with viewing angle 10° is used.

3 Proposed Method for DVR 3D data Visualization

3.1 Rasterization of a Line Segment

We know from linear algebra that a set $W^{\perp} = \{v \in V \,|\, \forall w \in W : w \perp v\}$ where V is the linear space and w, v are vectors, is called the orthogonal complement of the space W in V. It is known, that for all $v \in V$ exists one and only one $p \in W$ and $c \in W^{\perp}$; where $v = p + c$. The vector p (or c) is called an orthogonal projection (or a component) of the vector v with respect to subspace W.

Let us denote by $x \in V^{(3)}$ an arbitrary vector of 3D vector space. A mapping that projects this vector into a general plane with normal vector $n = (n_1; n_2; n_3)$ (an orthogonal supplement of space $\langle n\rangle^{\perp}$) is given by

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} = \begin{pmatrix} 1-n_1^2 & -n_1n_2 & -n_1n_3 \\ -n_1n_2 & 1-n_2^2 & -n_2n_3 \\ -n_1n_3 & -n_2n_3 & 1-n_3^2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{8}$$

The vector $p = (p_1; p_2; p_3)$ is the orthogonal projection of x, the component $c = (c_1; c_2; c_3)$ is given by

$$c = (x; n) \cdot n = \left(\sum_{i=1}^{3} x_i n_i \right) \cdot n \tag{9}$$

Let us denote by $\langle e_1; e_2; e_3\rangle$ the vector space generated by orthogonal base $\{e_1; e_2; e_3\}$. We use orthogonal projections $p_{1;2}$; $p_{1;3}$ and $p_{2;3}$ of a direction vector $u = (u_1; u_2; u_3)$ into subspaces $\langle e_1; e_2\rangle$; $\langle e_1; e_3\rangle$ and $\langle e_2; e_3\rangle$. Our situation is simplified by the fact that the direction of a projection line is collinear with some base vector in all cases. Fig. 13 illustrates the projection of the vector u into the subspace $\langle e_1; e_2\rangle$. Thus, the general formula (5) need not be used because, if $u = (u_1; u_2; u_3)$ and $\{e_1; e_2; e_3\}$ is the orthogonal base, then

$$\begin{aligned} p_{1;2} &= u_1e_1 + u_2e_2 = (u_1; u_2; 0) \\ p_{1;3} &= u_1e_1 + u_3e_3 = (u_1; 0; u_3) \\ p_{2;3} &= u_2e_2 + u_3e_3 = (0; u_2; u_3) \end{aligned} \tag{10}$$

The coordinates $u_1; u_2; u_3$ of the direction vector $u \in V^{(3)}$ can be obtained by two of the three projections

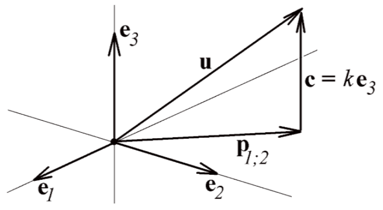

Figure 13: Projection $p_{1;2}$ of the direction vector u of a projection line into the subspace $\langle e_1, e_2\rangle$.

(4) which can be generated on an output device by Bresenham's algorithm described in Section 2. Thus, the generalization for a space results in Bresenham's algorithm being used twice. The plane in which it is supposed to work constitutes its additional parameter.

3.2 Illuminant Model for the Raster Data

Some improvements of the methods described in Sec. 2.2 were proposed — see [5, 6, 7, 12, 14] for example. It is recommended to apply some illumination models for improving previous outputs.

As was said in Sec. 2.2, there must be given a norm of surface point for illumination model calculating. But there exists no surface in a scalar field; the norm is therefore replaced by the gradient of this field in the point P. It is defined as

$$N_P \approx \text{grad } G(P) = \left(\tfrac{\partial G}{\partial x}(P); \tfrac{\partial G}{\partial y}(P); \tfrac{\partial G}{\partial z}(P) \right) \tag{11}$$

for the differentiation able function G which defines the value of the scalar field in dependence on the point P – see [12], [15] for example.

In the case of discrete data, this expression must be replaced by a vector of first central differences

$$N_P \approx \text{grad } G(P) \approx \left(\Delta_x^1 G(P), \Delta_y^1 G(P), \Delta_z^1 G(P) \right) \tag{12}$$

where $P = [x_i, y_i, z_i]$ and

$$\begin{aligned} \Delta_x^1 G(P) &= \frac{1}{2}(G(x_{i+1}, y_i, z_i) - G(x_{i-1}, y_i, z_i)) \\ \Delta_y^1 G(P) &= \frac{1}{2}(G(x_i, y_{i+1}, z_i) - G(x_i, y_{i-1}, z_i)) \\ \Delta_z^1 G(P) &= \frac{1}{2}(G(x_i, y_i, z_{i+1}) - G(x_i, y_i, z_{i-1})) \end{aligned} \tag{13}$$

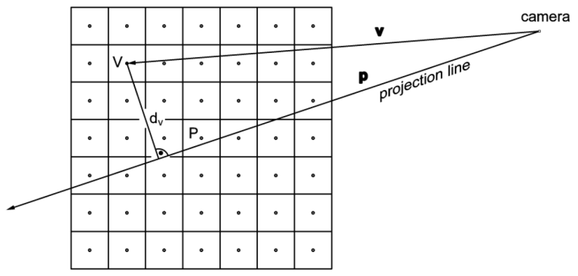

Figure 14: Distance d of the voxel V from the projection line.

However, the application of an illumination model cannot solve two problems said in the previous section: noise reduction and compensation for insufficient data resolution. Thus the question arises: is it possible to find a method by means of which we will be able to abolish at least one of these insufficiencies? The response to this question is positive. A suitable 3D filter is possible to solve both of these problems.

For each projection line p and each processed voxel P (see Fig. 8), the cubic neighborhood N which contains $(2n+1)^3$ voxels is setting. This situation is illustrated in Fig. 14 for $n = 3$ in 2D for simplicity. Distance d of each voxel V of this neighborhood from the projection line is calculated as:

$$d_V = \frac{||p \times v||}{||p||} \tag{14}$$

The weight c_V of the voxel V is specified using the Hanning window function

$$c_V = \begin{cases} \frac{1}{2} \cdot \left(1 + \cos \frac{\pi \cdot d_V}{r_{\max}}\right), & d_V < r_{\max} \\ 0, & d_V \geq r_{\max} \end{cases} \tag{15}$$

where $r_{\max}$ is the "radius of activity" of the processed voxel P. The estimation of total density d_P and the normal vector n_P in processed voxel P is then calculated as the weighted average of all voxels in a given neighborhood $\mathcal{N}$

$$d_P = \frac{\sum_{V \in \mathcal{N}} c_V v_v}{\sum_{V \in \mathcal{N}} c_V}; \quad n_P = \frac{\sum_{V \in \mathcal{N}} c_V n_v}{\sum_{V \in \mathcal{N}} c_V} \tag{16}$$

4 Results and Discussion

DVR method proposed in Sec. 3 works as 3D low-pass filter, i. e. it is possible to decrease high frequencies in the sense of the Fourier transform. It means that they effectively clear an additional noise. In Fig. 15 we can see a 3D model which is constructed using data from Fig. 2. Although this data contains a lot of noise, the 3D model is clear and relatively good quality (compare with Fig. 10). Video with some results is available for download from the author's website[1].

[1] https://dmartisek.cz/Veda/Paramecium_Euplotes_En.m4v

5 Conclusion

Cells observed by conventional or confocal microscopes are often highly transparent or translucent and do not produce visible shadows due to the transparent geometry of the structure inside the cell. Therefore, the current methods assume that an illumination model designed for surfaces cannot be used to visualize of cell structures. In this paper, it has been shown that the illuminant model can also be used for transparent or translucent objects described by raster data. After some modification, it is even possible to use models originally developed for CAD (Computer Aided Design) systems. These systems are usually used in engineering. The modified Phong model formulas were applied to each voxel of processed three-dimensional data. Our method allows using central projection not only for vector data but also for raster data. This is another important step for cellular representation. Our new method can be characterized by a "free camera". We can place the camera not only outside the cell (Fig. 15, 16, and 17) but also inside the cell (Fig. 18).

Acknowledgement: The author acknowledges support from Private Institute of Applied Mathematics, Šlapanice, Czech Republic. The author would like to thank prof. Roman Janisch from the Department of Biology, Faculty of Medicine, Masaryk University Brno, Czech Rep. and Assoc. Prof. Josef Reischig from the Department of Biology Charles University Plzen, Czech Rep. for the provided data.

References

[1] Blinn, J. F. Light reflection functions for simulation of clouds and dusty surfaces. *Acm Siggraph Computer Graphics 16*, 3 (1982), 21–29.

[2] Bresenham, J. E. Algorithm for computer control of a digital plotter. *IBM Systems journal 4*, 1 (1965), 25–30.

[3] Cohen-Or, D., and Kaufman, A. 3d line voxelization and connectivity control. *IEEE Computer Graphics and Applications 17*, 6 (1997), 80–87.

[4] Foley, J. D., Van, F. D., Van Dam, A., Feiner, S. K., and Hughes, J. F. *Computer graphics: principles and practice*, vol. 12110. Addison-Wesley Professional, 1996.

[5] Glassner, A. S. Space subdivision for fast ray tracing. *IEEE Computer Graphics and applications 4*, 10 (1984), 15–24.

[6] Kaufman, A. An Algorithm for 3D Scan-Conversion of Polygons. In *EG 1987-Technical Papers* (1987), Eurographics Association.

[7] Kaufman, A., and Shimony, E. 3d scan-conversion algorithms for voxel-based graphics. In *Proceedings of the 1986 workshop on Interactive 3D graphics* (1987), pp. 45–75.

[8] Martišek, D. Methods of detection of equipotential surfaces in cell structures. *Cells IV, Kopp Publ. České Budějovice* (2002), 47–54.

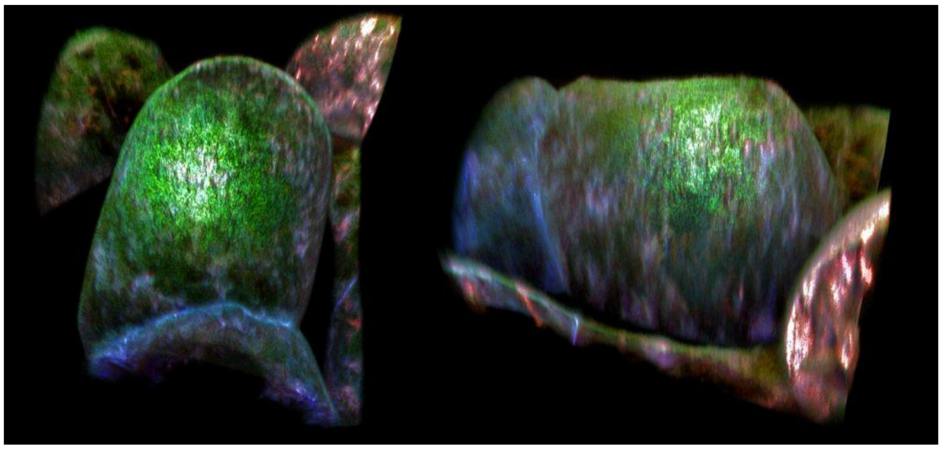

Figure 15: DVR 3D reconstruction of the Tobaco cells from Fig. 2. BtF algorithm; $I_0 = 0.2$; $t_i = 0.85$, cubic neighborhood for $n = r_{\max} = 3$, central projection with viewing angle 15°. Phong illuminant model with one light source, $I_a = 0.2$; $s = d = 0.4$; $a = 0.2$ (compare with Fig. 10).

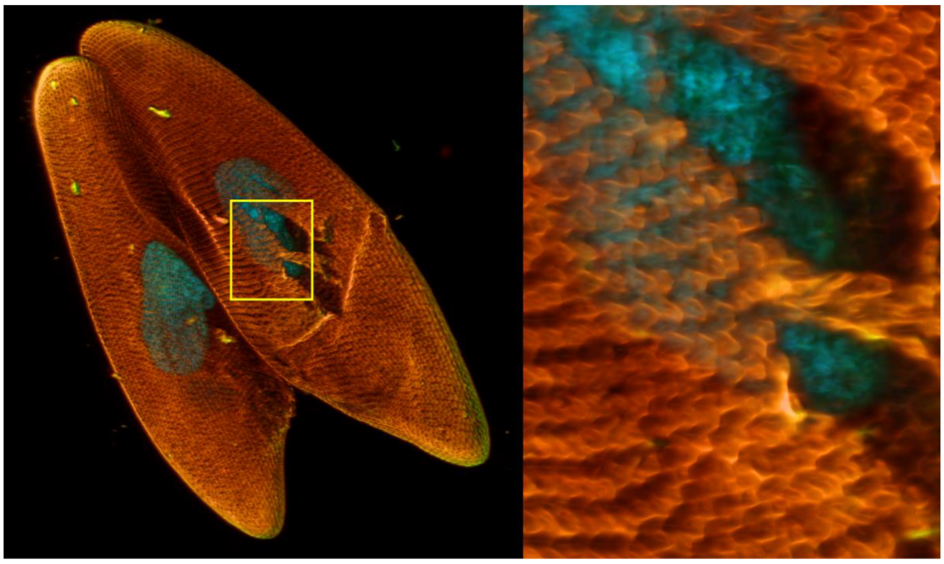

Figure 16: DVR 3D reconstruction of Paramecium caudatum cells from Fig. 3. BtF algorithm; $I_0 = 0.1$; $t_i = 0.9$, cubic neighborhood for $n = r_{\max} = 3$, central projection with viewing angle 15°. Phong illuminant model with one light source, $I_a = 0.2$; $s = d = 0.4$; $a = 0.2$ (compare with Fig. 11).

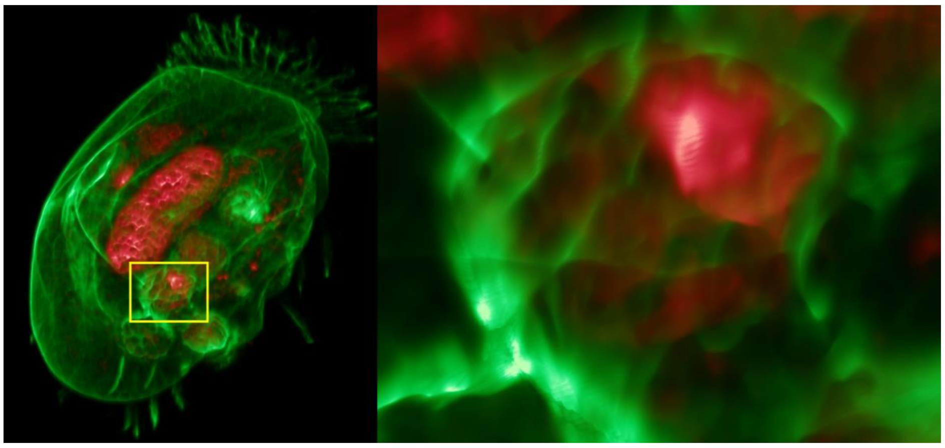

Figure 17: DVR 3D reconstruction of the Euplotes patella cell from Fig. 2. BtF algorithm; $I_0 = 0.2$; $t_i = 0.85$, cubic neighborhood for $n = r_{\max} = 3$, central projection with viewing angle 15°. Phong illuminant model with one light source, $I_a = 0.2$; $s = d = 0.4$; $a = 0.2$ (compare with Fig. 12).

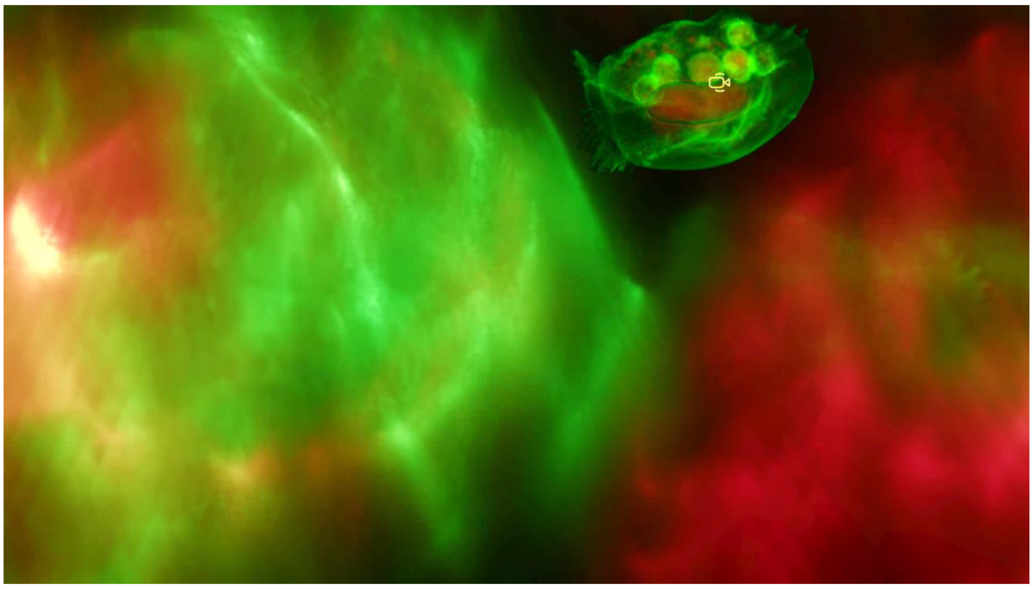

Figure 18: Virtual camera inside the Euplotes patella cell - DVR 3D reconstruction of the cell from Fig. 2. BtF algorithm; $I_0 = 0.2$; $t_i = 0.85$, cubic neighborhood for $n = r_{\max} = 3$, central projection with viewing angle 40°. Phong illuminant model with one light source, $I_a = 0.2$; $s = d = 0.4$; $a = 0.2$.

[9] Martisek, D. Computer estimation of jrc index using function moments. *MENDEL Journal 27*, 2 (2021), 51–58.

[10] Martišek, D., and Martišek, K. Direct volume rendering methods for cell structures. *Scanning 34*, 6 (2012), 367–377.

[11] Martišek, K. *Numerical methods of multispectral confocal microscopy.* PhD thesis, Diploma Thesis, Brno University of Technology, Brno, Czech Republic. p 1–65, 2007.

[12] Max, N. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics 1*, 2 (1995), 99–108.

[13] Mokrzycki, W. Algorithms of discretization of algebraic spatial curves on homogeneous cubical grids. *Computers & Graphics 12*, 3-4 (1988), 477–487.

[14] Nielson, G. The volume rendering equations. *tutorial for inclusion in the notes for CSE 573* (1993).

[15] Phong, B. T. Illumination for computer generated pictures. *Communications of the ACM 18*, 6 (1975), 311–317.