personálním počítači.

Literatura:

[1] Koenderink,J.J. - van Doorn,A.J.: Local structure of movement parallax of the plane. J.Opt.Soc.Am. 66 (1976) No 7 pp 717-723

[2] Koenderink,J.J. - van Doorn,A.J.: Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. Optica Acta 22 (1975) No 9 pp 773-291

[3] Koenderink,J.J. - van Doorn,A.J.: The singularities of the visual mapping. Biological Cybernetics 24 (1976) pp 51-59

[4] Koenderink,J.J. - van Doorn,A.J.: The Internal Representation of Solid Shape with Respect to Vision. Biological Cybernetics 32 (1979) pp 211-216

[5] Andronov,A.A. - Leontovich,E.A. - Gordon,I.I. - Maier,A.G.: Qualitative theory of second-order dynamic systems. New York: J.Wiley Sons 1973

[6] Plantinga, W.H. - Dyer, C.R.: An algorithm for constructing the aspect graph. Proc. IEEE 27th Symp. on Foundations of Comput. Sci. (1986) pp 123-131

[7] Maripuri,S.R. - Zeid,I.: Generating aspect graphs for nonconvex polyhedra. Computer-Aided Design 22 (1990) No 5 pp 258-264

[8] Preparata,F.P. - Shamos,M.I.: Computational Geometry - an introduction. New York, Berlin, Heidelberg, Tokyo: Springer-Verlag 1985

Ray tracing volume data with subvoxel precision

Miloš Šrámek

Institute of Measurement Science

Slovak Academy of Sciences,

Dúbravská cesta 9, 842 19 Bratislava,ČSFR

Volume visualization (VV), represents a wide plethora of methodologies aimed at processing of 3D scalar data, with a common goal to give a 2D presentation of desired data feature. It can be traced back to late seventies, when the first attempts to render 3D tomographic data on 2D screen came true. Since then, VV has widespread to such diverse branches as seismic measurements, meteorology, molecular structure analysis, confocal microscopy, CAD and astrophysical simulation are. However, due to wide exploration of various 3D medical imaging technologies (CT, MRI, PET ...), medicine still remains the VV basic domain.

Recently, an important task of quality and visual acceptability of rendered images is going to foreground. Since a distinguishing ability of the input data is usually limited by the scanning device, the only way how to e.g. display small details of an object of interrest in acceptable visual quality is to interpolate the input data.

In the contribution, a subvoxel precision volume visualization system based on ray tracing algorithm is presented. A trilinear interpolation scheme was chosen to find an exact surface–ray intersection points due to its computational simplicity and relatively low temporal demands. An effect of this approach is demonstrated by comparison of pairs of images, rendered either by standard and subvoxel precision approach.

1 Visualization and volume visualization

Visualization is a tool for display and understanding of large amounts of multidimensional scalar or vector data, produced either by computer simulations or by sophisticated measurement systems. It enables to investigate qualitative nature of the data and therefore represents a supplement to quantitative numerical analysis. Further, due to the large amount of data, it is very often not possible to analyze them quantitatively in the whole extent, and therefore visualization is the only tool being capable of such tasks.

Volume visualization (VV), a subbranch of visualization, represents a wide plethora of methodologies aimed at processing of 3D scalar data, with a common goal to give a 2D presentation of desired data feature. It can be traced back to late seventies [HL79], when the first attempts to render 3D tomographic data on 2D screen came true. Since then, VV has widespread to such diverse branches as seismic measurements, meteorology, molecular structure analysis, confocal microscopy, CAD and astrophysical simulation are. However, due to wide exploration of various 3D medical imaging technologies (CT, MRI, PET ...), medicine still remains the VV basic domain.

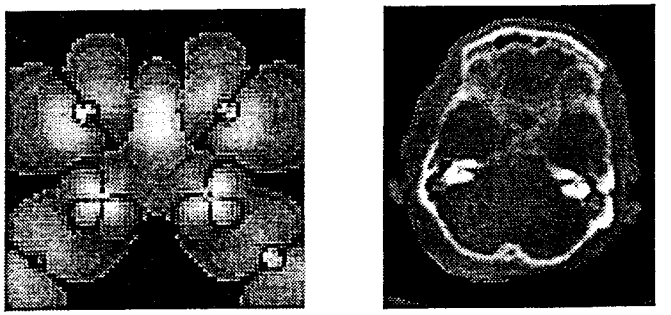
VV is in its nature a complicated process comprising of more consecutive steps, each of them having more alternatives. In the first part of our contribution, we are going to address the whole VV process with the emphasis on the most crucial steps.

In the second part of the contribution, a volume visualization system based either on binary and volume rendering techniques will be presented.

2 Volume visualization process

As it was already mentioned, volume visualization process comprises of more successive steps, which can be divided into following:

Data acquisition: The most typical form of either simulated or measured 3D data is a *stack of 2D matrices—slices*, simply because of the fact, that this representation enables their viewing by image viewers (Figure 1).



(a) (b)

Figure 1: (a) Electrone density map of High Potential Iron Protein molecule (b) one slice of a CT head sequence

Preprocessing: Usually, the data produced by measurement instruments or by simulation, are not in form, which is suitable for further processing aimed to 3D reconstruction. Therefore, e.g. conversion from vendor dependent data formats or 12 bit to 8 bit conversions in order to save space are necessary. Since a distinguishing ability of some scanners in direction perpendicular to slice plane is up to an order of magnitude lower than that in the slice plane, it is necessary to compute missing slice either by linear or nonlinear interpolation schemes.

Like any other measurement systems, also 3D scanners produce data distorted by noise. Therefore it is necessary to process them by noise reduction filters. Recently, an *anisotropic diffusion filtering* is becoming popular, because of efficient noise removing in homogeneous regions and good preserving or even sharpening of edges

[BvHM92] .

Segmentation: Specification of a region of interrest is the most complex preprocessing task. Although in some cases a simple thresholding or windowing is sufficient to define the region of interrest (e.g. bone tissue in CT tomography or blood vessels in magnetic resonance angiography [PBH92]), generally knowledge based approaches or more or less interactive systems have to be used to accomplish this task. Lately, some segmentation methodologies belonging to the second group were described [HH92] [HsvSB92], which were also based on thresholding, in this case supplemented by repetitive application of various kinds of morphological operators (mostly 3D erosion and dilatation) and connected component labeling to break false connections between disjunct objects.

Volume viewing: In comparison to 2D data sets, 3D sets have no direct visible manifestation. Therefore, some intermediate model, object or phenomenon, must be chosen that can be rendered to produce an image. Depending on this intermediate representation, most techniques for display of volumetric data fall into 3 categories:

- *Surface-based techniques* These techniques are historically oldest [Kep75] and represent application of the standard computer graphics methods to 3D volume data rendering. They work in two steps. First, a *geometric representation* of an object, a surface model, which is obtained by fitting geometric primitives to surface detected by segmentation is built. As a second step, either *standard* CG packages or *special rendering tools* can be used to create an image.
- *Binary voxel rendering techniques* The characteristic feature of these techniques is, that the intermediate surface model is omited, and that the surface is rendered directly by processing of the whole segmented data volume. Since no volume information is lost, it is possible to combine shaded surfaces with original gray scale cut planes.

Recently, the greatest popularity have gained techniques based on well known *ray tracing* algorithm [Lev88]. For each pixel of the resulting image, a ray is fired into the scene (Figure 2). It is is traced, until the first object voxel,

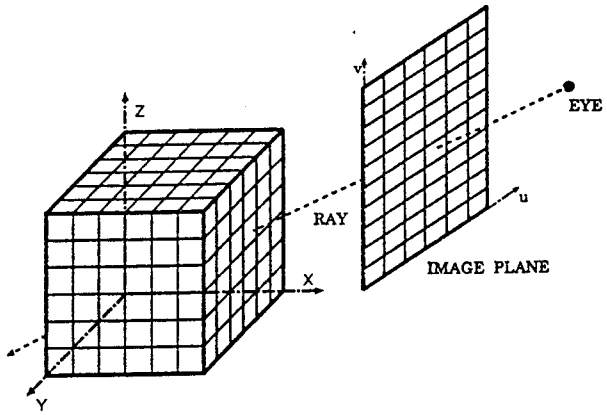
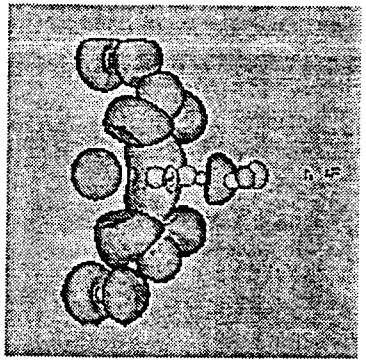


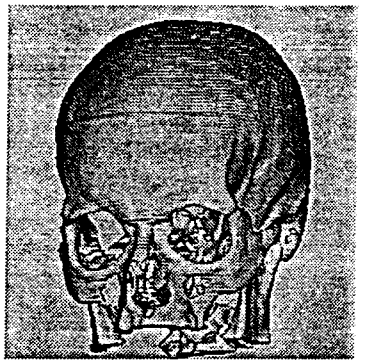
Figure 2: Ray tracing geometry

identified e.g. during the preprocessing phase, is found. Finally, a color which is computed in dependence on local object properties is assigned to the image pixel. This approach, due to its variability is an ideal way for interactive data exploration. Its main drawback is large amount of processed data, which results in relatively slow, although high quality, rendering (Figure 3).

- *Volume rendering techniques* [Lev88, DCH88] are a variation of the previously mentioned techniques based on ray tracing, with the difference, that instead of segmentation, a color and opacity are assigned to each voxel prior to rendering. The image pixel color is defined by blending together contributions of all voxels, which are hit by the ray on its route across the scene. This method shows good results for objects with surfaces, which are not well defined (various types of soft tissues in CT tomography, thin objects).



(a)



(b)

Figure 3: Ray tracing with gray level gradient shading: (a) Protein molecule (b) scull from CT slices

Shading: Once the surface voxel is found, it is necessary to compute a corresponding color which should be assigned to the pixel, where it is projected on. Human visual perception and the fundamental laws of optics should be taken into account in order to produce high quality, photography like images. One of such models is so called *Phong shading*, which takes into account ambient light influence as well as the diffuse and specular reflection(responsible for highlights):

$$I = I_a k_a + I_l(k_d \cos\theta + k_s \cos^n \alpha) \qquad (1)$$

where k_s is a specular reflection constant ($0 \leq k_s \leq 1$), and α is the angle between the direction of reflection (R) and the direction to the eye (E). Varying the power exponent n, more or less focused highlights can be generated. Term $k_d \cos\theta$ represents the contribution of diffuse reflection to pixel color, where θ is an angle between surface normal (N) and light source direction vector (L) and I_d is a diffuse reflection constant for given material. The additional constant term ($I_a k_a$) is used to simulate

the influence of ambient light (Figure 4). I_a represents the ambient light intensity, and k_a, the ambient diffuse reflection constant ($0 \leq k_a \leq 1$).

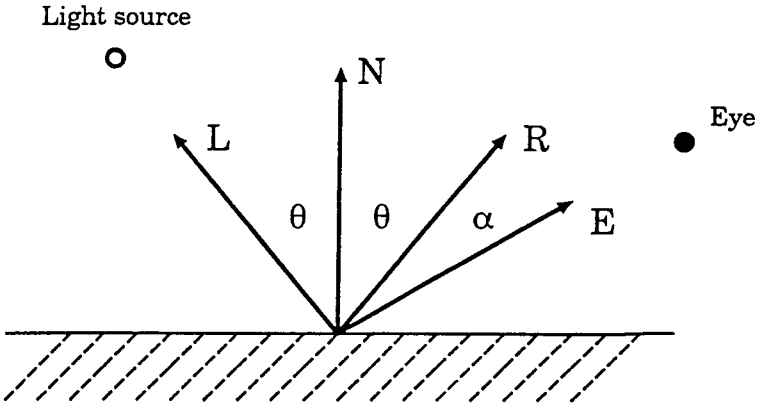


Figure 4: Shading

From equation 1 we see that both angles θ and α are dependent on a surface normal vector N. Therefore, the fidelity of the rendered image is tightly bounded with correspondence of the computed normal to real surface normal vector. The most widespread method of normal computation is *gray level gradient shading* [HB86], based on partial volume effect. When, say two types of tissues contribute to one voxel, the resulting value will be dependent on relative volume of both of them. Thus the gray level gradient can be considered as a measure of surface inclination. The gradients are calculated from the gray values of some of the neighborhood voxels. Given a voxel with coordinates (x, y, z) and gray value $g(x, y, z)$, the approximated surface normal—considering the 6-connected neighborhood—of the voxel is the vector

$$\mathbf{N} = \begin{pmatrix} g(x+1,y,z) - g(x-1,y,z) \\ g(x,y+1,z) - g(x,y-1,z) \\ g(x,y,z+1) - g(x,y,z-1) \end{pmatrix}. \quad (2)$$

3 Subvoxel precision ray tracer

Let's imagine, that we want to render image of an object, which is defined within a scene with resolution 64^3. If we want to obtain an image with size, say, 300x300 pixels, each object voxel will be inevitably projected on more image pixels resulting in blocky artifacts and 'jaggies' along edges of objects. Although these artifacts are similar to those, we know from ray tracing of analytically defined objects, they cannot be removed by a standard supersampling methods, since this distortion is caused by insufficient sampling rate of original data for given size of an image. Right column of figure 5 shows this case for 3 different magnifications. While the first image, where pixel size corresponds to voxel size, shows no apparent aliasing, artifacts on the third one are clearly visible.

In order to remove these artifacts, it is necessary to replace the discrete function defined by the scene by some kind of continuous one. The computationally as well as algorithmically easiest solution of this problem is provided by a linear interpolation, or in our 3D case, by a trilinear interpolation from 8 samples.

Approach presented in this contribution belongs to the class of binary voxel rendering algorithms. In the following, its two versions, resulting from two different ways of scene segmentation, are described.

The first, more precise version is based on simple threshold segmentation. We know, that contours of real scanned objects are never sharp, but they are more or less blurred due to the *partial volume effect (PVE)*. It was shown [HB86], that we can make use of this artifact for the surface normal evaluation resulting in realistic shading of objects (gray level gradient shading - GGS). In our algorithm it is also used for precise surface point

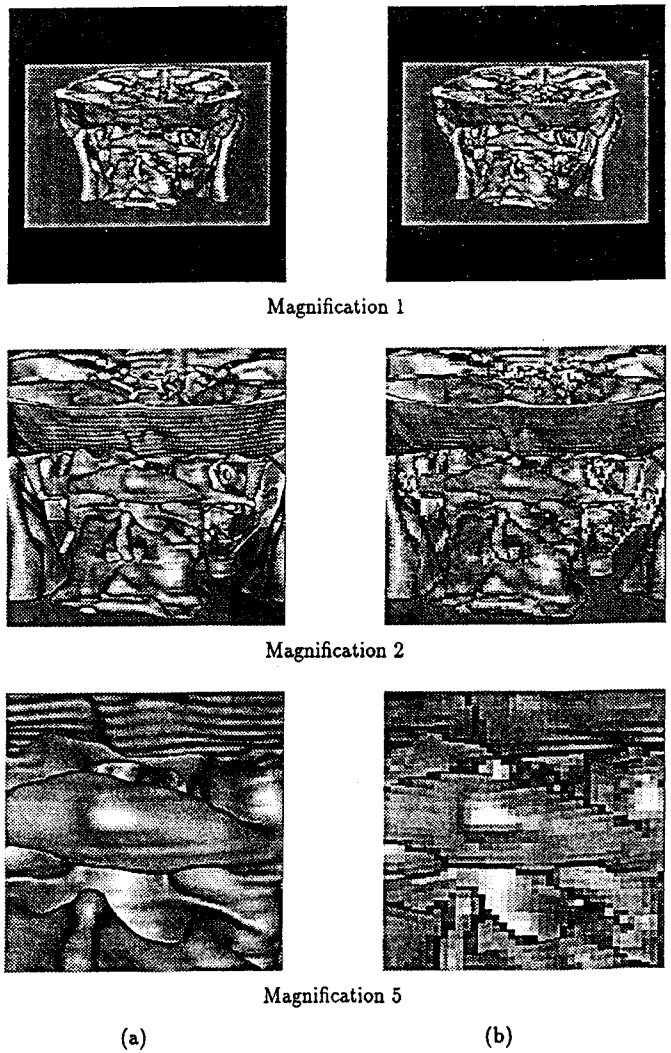


Figure 5: (a) Image rendered at 3 different magnifications (a) with subvoxel precision, (b) standard approach

computation. A discrete sample array is first replaced by a continuous function, a trilinear interpolation of values of neighboring voxels. An exact intersection point is then searched as the nearest ray point , where the interpolated function is equal to the threshold value. The surface normal corresponding to this exact surface point is computed in a similar way as the trilinear interpolation of normals in vertices of the actual voxel.

The second version is also for objects detected by thresholding, but in this case followed by some other operations (manual editing etc.). In this case, some voxels or even a whole layer of voxels may be removed or added, and therefore the previous scheme may fail. In order to overcome this, the binary scene is interpolated instead of the original one, so that object voxel is assigned value 1, the background voxel is assigned value 0 and the threshold value is set to 0.5. Since we assume, that we are still in the PVE transient region, normal is computed in the same way as before.

3.1 Algorithm description

There are two possible interpretations of the voxel as the basic unit of a 3D binary scene. It may be treated either as a sample of the measured quantity at dimensionless grid point or as a number which represents its mean value in the voxel volume (cuberille model). E. g. volume rendering algorithm proposed by Levoy [Lev88] is based on the first model. Equidistant samples along the ray are computed first by trilinear interpolation and these values are subsequently used to compute the pixel color.

The presented subvoxel precision ray tracing algorithm is based on combination of both approaches.

The first phase of tracing a single ray, the 0-voxel traversal, is based on *fast vozel traversal* algorithm [CW88] utilizing the cuberille representation of the scene. This algorithm approximates the digital line by making use of 3 decision variables (in 3 dimensions) which represent total distance traversed along each of coordinate axes. The next voxel along the ray is found by increment of decrement of that coordinate, the corresponding already

traversed distance is smallest. This algorithm is not only very fast, since it needs only few instructions for one step, but enables also to compute precise voxel entry and exit points, which is important for exact ray–object intersection computation.

The second phase, the precise ray–object intersection, is based in on the first, sample interpretation of the voxel. A density profile along the ray within the voxel is computed by trilinear interpolation from values corresponding to voxel vertices. Since each voxel vertex is shared by 8 adjacent voxels of different densities, the corresponding vertex value is computed as the mean of these values (figure 6). The nearest ray point, where the interpolated density exceeds the threshold value, is the searched surface point. If there is no such point, we continue with the next voxel until the surface point is found or the ray leaves the scene.

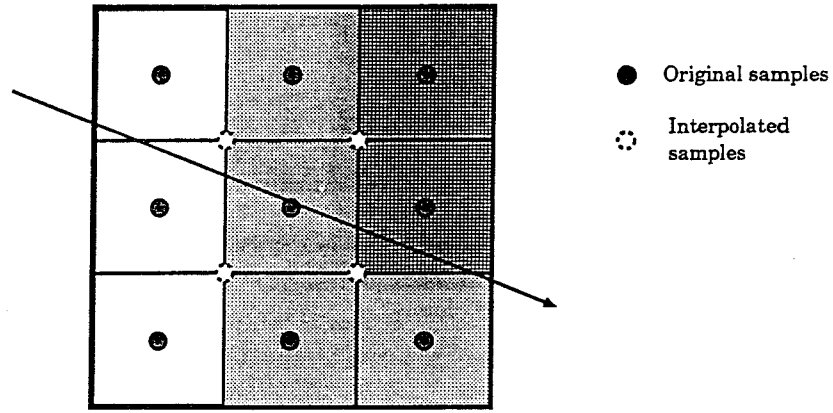


Figure 6: Vertex value interpolation

It may happen, that the interpolated density exceeds the threshold value along the whole ray segment within the voxel.

In this case, the intersection point should be searched for outside the object defined by the

1-voxel set. The ray should be in this case either traced back in reversed order, or, in the preprocessing phase, the object should be dilated with 3x3x3 structuring element. Object is thus covered by a one voxel thick layer, which ensures, that the exact intersection is always found properly (figure 7).

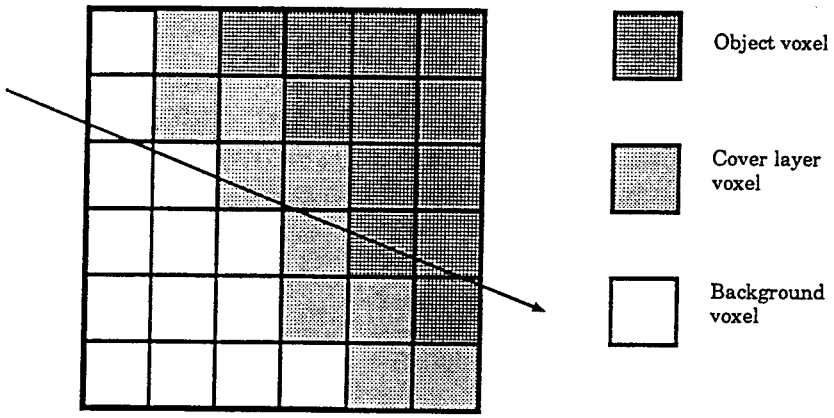


Figure 7: Dilated binary scene

Finally, a standard Phong's shading (equation 1) is used to compute a pixel color. First, normal for each voxel vertex is computed as a gradient from densities of 8 voxels and , second, the desired surface point normal is obtained as the trilinear interpolation from these values. Since function defined in this manner is continuous, resulting images show no artificial edges along the voxel boundaries.

4 Results

In this section we would like to answer following two questions:

- which is the accuracy of the proposed method, and

- up to which magnification can it be used.

The problem of accuracy of 3D render images of volumetric data has been up till now widely investigated ([MLD88], [THB+90], [YCK92]).

For our case we have chosen a method, similar to this used in [THB+90], which is based on simulated tomographic data. As the test object a sphere was chosen to generate the artificial tomographic slices, because medical objects usually have smooth surfaces, which can be approximated by a sphere of bigger or smaller diameter.

In order to simulate the partial volume effect, each voxel near the sphere surface was assigned a value proportional to its distance from this surface:

$$s = \begin{cases} M & \text{if } d < r - t \\ m & \text{if } d > r + t \\ m + \frac{(M-m)(d-(r-t))}{2t} & \text{otherwise} \end{cases} \quad (3)$$

where d is distance of the actual voxel from the sphere center; r is the sphere diameter; t is the surface transient region thickness and M and m are densities inside and outside of the sphere. Following values were used to generate the test data:

$$\begin{aligned} M &= 128 \\ m &= 0 \\ t &= 1.5 \end{aligned} \quad (4)$$

In order to find answers on our two questions, a following experiment was done: A set of 128x128x128 scenes each with the sphere situated at [64,64,64] with diameters 50, 25, 10, 5, 3.33 and 2 was generated. These objects were subsequently rendered with magnification giving an image of the same size (1, 2, 5, 10, 15 and 20). Total of three different surface detection algorithms were used:

- with grid point accuracy,
- with subvoxel accuracy and gray level interpolation and

- with subvoxel accuracy and binary interpolation.

The gray level gradient shading with diffuse reflection was used in all three cases, the latter two of them with normal interpolation.

Except of this three images an exactly rendered sphere surface image with the same lighting conditions was computed for each magnification. This precise model served us as the reference for evaluation and comparison of our renderings from the aspect of following criteria:

normal error: which is given as a mean of an angle between computed and precise normal for all image pixels,

area error: rendered sphere image diameter is computed as a mean of distances from the sphere center projection of those image pixels, which belong to sphere image but have a background pixel in their neighborhood and

shape error: given as a standard deviation of the distances from the previous item.

Numerical results of this experiment are depicted on figure 8, while figures 9 and 10 show some of the rendered surfaces and normal deviation distribution.

The graphs as well as images show, that in the case when magnification equals 1 (which means that image pixel and scene voxel have edges of the same size) all three types of renderings are approximately the same.

The situation is quite different for higher magnifications. While at magnification 5 are both gray level and binary interpolated images very good and at magnification 10 still acceptable, voxel precision renderings show blocky artifacts already at magnifications smaller than 5. This can be seen also at graphs (a) and (c) on figure 8, which show the normal and shape deviation to be of an order of magnitude worser for non interpolated images as for the interpolated.

What concerns the magnification above 10, the image quality decreases. Although the shading is still no so bad (see figure 9), the size of the object is smaller as it should be. It can be seen in graph (b) on figure 8 or as a black surrounding around the object on figure 10. However, the quality of renderings with interpolation is still much better in comparison with that of non interpolated images.

References

[BvHM92] I. Bajla, M. Šrámek, D. Hraško, and M. Marušiak. A comparison study of smoothing techniques for 3-d image synthesis from mri data. In *Proc. of 11-th international conference Biosignal '92*, pages 31–32, Brno, ČSFR, 1992.

[CW88] John C. Cleary and Geoff Wyvill. Analysis of an algorithm for fast ray tracing using uniform space subdivision. *The Visual Computer*, 4(2):65–83, July 1988.

[DCH88] Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *Computer Graphics*, 22(4):65–74, August 1988.

[HB86] Karl Heinz Höhne and Ralph Bernstein. Shading 3D-images from CT using gray-level gradients. *IEEE Transactions on Medical Imaging*, MI-5(1):45–47, March 1986.

[HH92] Karl Heinz Höhne and William A. Hanson. Interactive 3d segmentation of mri and ct volumes using morphological operations. *Journal of Computer Assisted Tomography*, 16(2):285–294, March/April 1992.

[HL79] Gabor T. Herman and Hsun Kao Liu. Three-dimensional display of human organs from computed tomograms. *Computer Graphics and Image Processing*, 9:1–21, 1979.

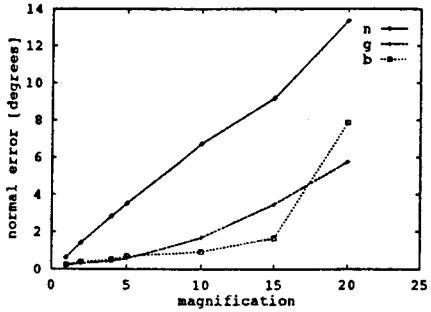
[HsvSB92] Igor Holländer, Miloš Šrámek, and Ivan Bajla. Problémy predspracovania a syntézy 3d obrazov v tomografii, časť 2: Segmentácia 2d a 3d obrazov. *Lékař a technika*, in press, 1992.

[Kep75] E. Keppel. Approximating complex surfaces by triangulation of contour lines. *IBM Journal of Research and Development*, 19(1):2–11, January 1975.

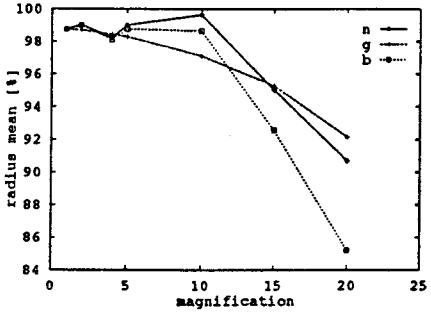
[Lev88] Marc Levoy. Display of surfaces from volume data. *Computer Graphics and Applications*, 8(3):29–37, May 1988.

[MLD88] Maria Magnusson, Reiner Lenz, and Per-Erik Danielsson. Evaluation of methods for shaded surface display of ct-volumes. In *Proceedings of the 9th International Conference on Pattern Recognition*, Rome, Italy, November 14-17, 1988.

(a)



(b)



(c)

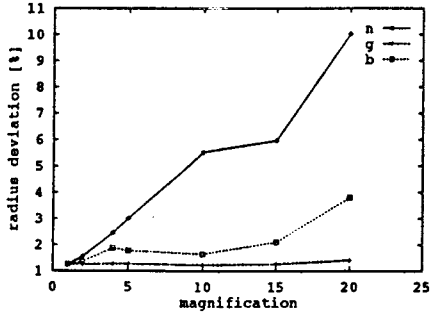
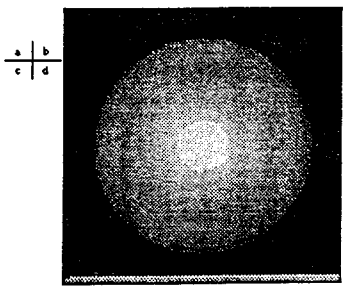
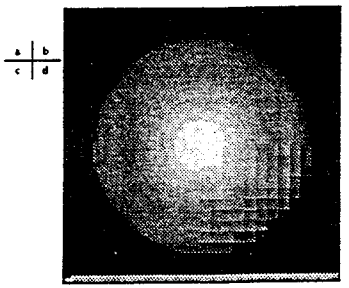


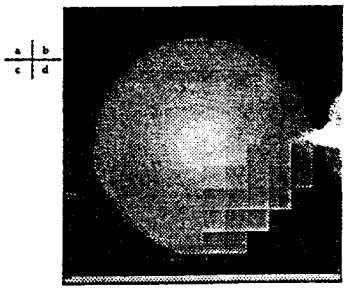
Figure 8: Dependence of (a) normal error, (b) diameter mean and (c) diameter standard deviation on magnification (g – gray level, b – binary, n – no interpolation).



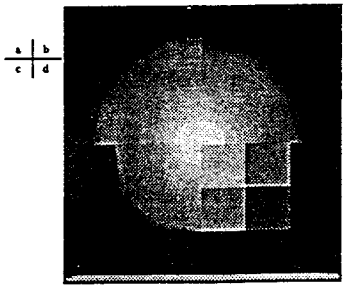
Magnification 1



Magnification 5



Magnification 10



Magnification 25

Figure 9: Comparison of rendered spheres at different magnifications: (a) precise, (b) gray level, (c) binary and (d) no interpolation.

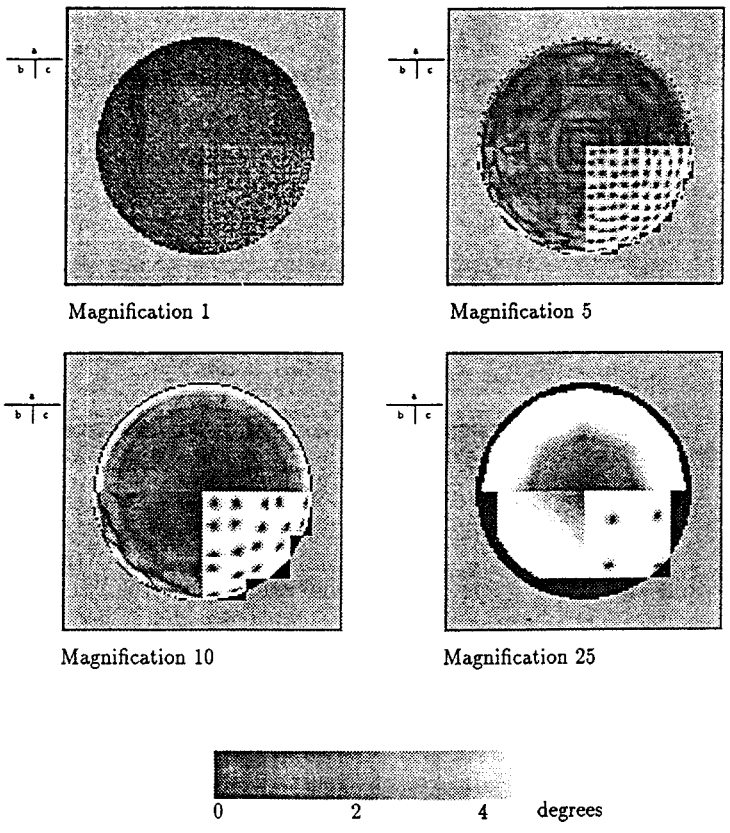


Figure 10: Normal error at different magnifications: (a) gray level, (b) binary and (c) no interpolation.

[PBH92] Andreas Pommert, Michael Bomans, and Karl Heinz Höhne. Volume visualization in magnetic resonance angiography. *Computer Graphics and Applications*, 12(5):12–13, September 1992.

[THB+90] U. Tiede, K. H. Höhne, M. Bomans, A. Pommert, M. Riemer, and G. Wiebecke. Investigation of medical 3D-rendering algorithms. *Computer Graphics and Applications*, 10(3):41–53, 1990.

[YCK92] Roni Yagel, Daniel Cohen, and Arie Kaufman. Normal estimation in 3d discrete space. *The Visual Computer*, 8(5-6):278–291, June 1992.