Object Voxelization by Filtering

Miloš Šrámek* and Arie Kaufman†

Center for Visual Computing (CVC)‡ and Department of Computer Science
State University of New York at Stony Brook
Stony Brook, NY, 11794-4400

Abstract

We analyze different filters used for the voxelization of analytically described objects. We show that, when the voxel model is used for visualization, the filter design is related to the subsequent rendering phase, namely the gradient estimation technique. Our theoretical and experimental analyses show that, in order to avoid a systematic error in normal estimation, the density profile near the surface should be linearly proportional to the distance from the surface. Optimal thickness of this transient region has been estimated to be about 3 voxel units. Based on these results, we propose a technique for voxelization of arbitrary parametric surfaces, using a hierarchical subdivision of the 2D surface domain.

CR Categories I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling — Curve, surface, solid, and object representations; I.3.6 [Computer Graphics]: Methodology and Techniques — Graphics data structures and data types

Keywords Volume graphics, Volume rendering, Filter-based voxelization, Normal estimation, Error estimation, Parametric surfaces, Hierarchical subdivision.

1 Introduction

Volume graphics represents a set of techniques aimed at modeling, manipulation and rendering of geometric objects, which have proven to be, in many aspects, superior to traditional computer graphics approaches [1]. The basic idea resides in the representation of a geometric object by means of 3D raster of elementary volume primitives — voxels. This data structure is the same as that of scanned real objects and thus enables simultaneous handling and rendering of synthesized and real objects.

The main advantages of volume graphics are: (i) decoupling of voxelization from rendering, (ii) uniformity of representation, and (iii) support of Boolean, block and CSG operations (for more details see [1]). Two drawbacks of volume graphics techniques are their high memory and processing time demands. However, due to the progress in both computers and specialized volume rendering hardware [2, 3], these drawbacks are gradually losing their significance.

To be represented by the voxel raster, an object has to be subjected to a process called *voxelization* or *3D scan conversion*. This is essentially a sampling process, and therefore sampling theory rules should be taken into account.

The first voxelization algorithms were binary, assigning, say, 1 to occupied voxels and 0 to unoccupied [4, 5]. This approach totally ignored sampling theory, and consequently rendered pictures suffering from aliasing, which was predominantly related to the poor estimation of the surface normal vector. Post-voxelization techniques were proposed to improve the normal vector estimation by taking into account information from a larger neighborhood (e.g., contextual shading [6], context sensitive normal estimation [7], center-of-gravity shading [8]). Although the improvement was significant, none of the techniques yielded a normal vector precise enough for the simulation of such effects as reflection and refraction of light on an object surface. However, impressive results were obtained by a discrete ray tracing [9]. In this technique, in addition to estimating the normal from the discrete data, the normal was confirmed from the analytic object description, which is kept along with the voxel raster.

Höhne and Bernstein [10] pointed out that shading of scanned objects could be significantly improved, if one takes advantage of "inaccuracy" of the 3D scanning device. Due to physical limitations, a point spread function of the scanner is not the ideal dimensionless pulse, but rather it has a Gaussian like profile with a finite support. Therefore, it acts as a low-pass filter, suppressing high frequencies and blurring object edges. This is known as the partial volume effect (PVE). Using such data, realistic shading can be achieved when the normal is computed by means of a discrete gradient filter (e.g., by central differencing).

The first "smoothed" objects were generated for the sake of algorithm testing. A value, proportional to the distance from a center of the test sphere, was stored in voxels near its surface [11]. Other test objects were obtained by simulating the PVE by computation of relative occupancy of each voxel, shared both by the object and background [12].

Techniques for voxelization of smooth objects, primarily aimed at visualization, were proposed later [8, 13, 14, 15, 16, 17]. Their common feature was that some kind of filtering function had been used to suppress the aliasing. Although high quality images have been generated by these techniques, the following questions remain open and demand systematic investigation:

1. Which is the most suitable voxelization filter?

2. Is there any relationship between the choice of the filtering technique and a normal estimation method?

*milos@cs.sunysb.edu, currently on leave from Institute of Measurement Science, Slovak Academy of Sciences, Bratislava, Slovakia
†ari@cs.sunysb.edu
‡http://www.cvc.sunysb.edu/

The voxelization techniques implement digitization and quantization of a continuous signal, while the rendering can be described as resampling — reconstruction of a continuous signal with subsequent sampling and projection [5, 18, 19, 20, 21]. Although both the voxelization and rendering phases are relatively independent and have been treated as such so far, we show that the quality of the rendered image can be improved by mutual adjustment of both the smoothing filter of the voxelization and reconstruction filter in the rendering

The goal of this work is to get a deeper insight and to propose some simple rules, which should be followed in order to voxelize objects with high rendering quality (Sections 2–4). Further, as a case study, a technique for voxelization of arbitrary parametric surfaces is proposed (Section 5).

2 Voxelization Filter Design

In voxelizing a geometric object, the discontinuity at its surface results in an unbounded frequency spectrum. Therefore, in order to suppress undesired aliasing in the sampled data, the high frequencies above the Nyquist rate must be eliminated by low-pass filtering. The ideal low pass filter, the *box* filter in the frequency domain, results in an unlocalized and oscillating *sinc* filter in the spatial domain. In order to avoid these drawbacks, blurring filters that are nonnegative in the spatial domain are often used. The cone shaped Bartlett function

$$f_B(\mathbf{p},\mathbf{p}_f) = \begin{cases} k_B(1-\frac{\|\mathbf{p}-\mathbf{p}_f\|}{r_f}) & \text{if } \|\mathbf{p}-\mathbf{p}_f\| < r_f \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

has been used [13], where $\mathbf{p}_f$ is the position of the filter center, r_f is the diameter of its spherical support, $\mathbf{p}$ is an arbitrary point and k_B is a normalization constant.

Motivation to use the Gaussian filter

$$f_G(\mathbf{p},\mathbf{p}_f) = k_G \exp\left[-\frac{\|\mathbf{p}-\mathbf{p}_f\|^2}{2\sigma^2}\right], \quad (2)$$

where σ defines its width, is twofold [8]. First, it is well known from scale-space theory [22] that filtering by a Gaussian simplifies the data, not introducing any new, artificial structures. Second, the Gaussian can be used to approximate the shape of a point spread function of a real scanning device, which has finite dimensions due to physical limitations and random errors. The voxelization itself consists of convolving the object $O(\mathbf{p})$ by the filter $f(\mathbf{p})$,

$$O_f(\mathbf{p}) = f(\mathbf{p}) \otimes O(\mathbf{p}) \quad (3)$$

and sampling the result at discrete locations $\mathbf{p}_{ijk} = (a\,i, a\,j, a\,k)$, where a is sampling step and (i,j,k) are integer coordinates of the sample in the volume data set.

The convolution must usually be evaluated numerically. However, its computational complexity is very high, $\mathcal{O}(m^3n^3)$, where m represents the resolution of the voxelized data grid and n is the resolution of the filter. Hence, different speedup techniques have been proposed. Lookup tables were precomputed for different objects and distance to the object surface was used as its index [13]. In another approach, object shape was not taken into account and convolution of a half-space object with the Gaussian kernel, Eq. 2, was computed [8]. In this case, as it was shown, the convolution can be in the surface vicinity approximated by a function, linearly proportional to the distance from the surface:

$$D(x,y,z) = \begin{cases} 1 & \text{if } x-x_0 < -r_f \\ 0 & \text{if } x-x_0 > r_f \\ (1-\frac{x-x_0}{r_f})/2 & \text{otherwise} \end{cases}, \quad (4)$$

where $x-x_0$ is the distance of the point (x,y,z) from the object surface $x = x_0$. The linear profile is a result of convolution of an object with a box filter. In our case, the box filter is only one dimensional and is applied along a direction, perpendicular to the object surface. Therefore, we call it an *oriented box filter*.

There are thus two competing approaches. One, where the shape of the object is taken into account and for each object a different lookup table is computed, and a second one, where the shape of the object is neglected and all objects are treated uniformly. Apparently, the second approach is only an approximation. However, in this case, both the blurring and normal estimation filters are of the first degree. This can result in more precise reconstruction of the underlying derivative and surface normal direction, than the first case with the higher order blurring filter [23]. Therefore, we set up a computer experiment, based on rendering shapes with different surface curvatures in order to evaluate properties of both voxelization approaches and to evaluate performance of various types of gradient filters (see Section 4).

3 Normal Estimation in Sampled Data

The most common method for the computation of a surface normal vector in sampled data is by gradient estimation using the *central differences*, and its subsequent normalization. Several authors have pointed out that this filter has a narrow bandwidth and smoothes out details of an image. An adaptive technique was proposed [12], which took into account the maximum of the forward and backward differences:

$$g^x_{ijk} = \max(d_{i+1,j,k} - d_{i,j,k}, d_{i,j,k} - d_{i-1,j,k}). \quad (5)$$

An adjustable filter was proposed by Goss [24], based on truncation of the ideal gradient filter $\frac{\cos \pi x}{x}$ by means of the Kaiser window. Response of the filter to higher frequencies can be adjusted by the parameter α of the Kaiser window, modifying its width.

Typically, properties of gradient estimation filters are investigated for arbitrary volumetric data without any specific assumptions about their properties (except for their band limitedness). However, in the case of voxelization of geometric objects, we can take advantage of the fact that we have full control over the properties of the volume data, and thus we can adjust the voxelization filter so that the gradient estimation filter gives the best possible results. In our case we prefer the central differences filter due to simplicity of its implementation and its low computational demands. In the following we show that gradient estimated by the central differences filter has the direction of the true surface normal only if the density profile in the surface vicinity depends linearly on the distance from the surface.

Let $h(d)$ be the density of the voxelized object depending only on the distance d from its surface. Let the true surface normal (2D case) have direction vector $(\cos\theta, \sin\theta)$ (Figure 1). The surface normal vector, estimated by central differences for point V, is then

$$g_x = h(d_B) - h(d_A) \quad (6)$$
$$g_y = h(d_D) - h(d_C) \quad (7)$$

where

$$d_A = d_V - d_x \quad (8)$$
$$d_B = d_V + d_x \quad (9)$$
$$d_C = d_V - d_y \quad (10)$$
$$d_D = d_V + d_y, \quad (11)$$

d_A, d_B, d_C, d_D and d_V are distances of the corresponding points from the surface and

$$d_x = a\cos\theta$$
$$d_y = a\sin\theta. \quad (12)$$

Assuming differentiability of $h(\cdot)$ in $(-\infty, \infty)$, it can be expressed by a Taylor series in the vicinity of d_V:

$$\begin{aligned} h(d_A) &= h(d_V) + \sum_{i=1}^{\infty} k_i \, (-d_x)^i \\ h(d_B) &= h(d_V) + \sum_{i=1}^{\infty} k_i \, d_x^i \\ h(d_C) &= h(d_V) + \sum_{i=1}^{\infty} k_i \, (-d_y)^i \\ h(d_D) &= h(d_V) + \sum_{i=1}^{\infty} k_i \, d_y^i. \end{aligned} \tag{13}$$

Then

$$\begin{aligned} g_x &= 2 \sum_{i=1}^{\infty} k_{2i-1} d_x^{2i-1} \\ g_y &= 2 \sum_{i=1}^{\infty} k_{2i-1} d_y^{2i-1} \end{aligned} \tag{14}$$

since the symmetric terms with $i = 2, 4, \ldots$ cancel out. Taking Eq. 12 into account, we get

$$\begin{aligned} g_x &= 2a \left(k_1 \cos\theta + \sum_{i=2}^{\infty} k_{2i-1} (\cos\theta)^{2i-1} \right) \\ g_y &= 2a \left(k_1 \sin\theta + \sum_{i=2}^{\infty} k_{2i-1} (\sin\theta)^{2i-1} \right). \end{aligned} \tag{15}$$

We can see that the estimated normal vector (g_x, g_y) is parallel to the true normal vector $(\cos\theta, \sin\theta)$ only if coefficients k_{2i-1}, $i \geq 2$ in the Taylor expansion are equal to zero. However, in order to blend the high value inside the object with the low value outside, $h(d)$ must be antisymmetric, and thus all its symmetric coefficients k_i, $i = 2, 4, \ldots$, in Eq. 13 must also be equal to zero.

Consequently, the only possibility we have is that $h(d)$ be linear with d, which means that any other profile introduces an error in normal estimation, with magnitude depending on the surface orientation. This is illustrated in Figure 2. The angle between the true and estimated normals for a Gaussian filter with $\sigma = 1$ was computed by a procedure similar to that proposed earlier, for $a = 1$. We can see that the error is minimal for surface orientations 0, 45 and 90 degrees, i.e., in the cases when either one of the normal components approaches 0 or both are approximately equal. Maximal amplitude of the error is about 4.5 degrees (0.07 radians), which significantly exceeds the empirically estimated visually indistinguishable angular difference (0.01 radians) between two surface normals [25]. Figure 3a shows distribution of this error on a sphere surface. Due to its smooth distribution over the surface, this deviation in surface normal estimation is nearly invisible in static images. However, in animated sequences it results in unpleasant and disturbing artifacts.

4 Experimental Evaluation

A set of computer experiments has been set up in order to compare our voxelization and normal estimation techniques, as well as verify the theoretical conclusion about the precision of surface normal estimation. For several reasons, spheres have often been used to compare performances of different algorithms [7, 11, 12, 13]: (i) a sphere is a simple object and, given its center and radius, surface position and normal can be computed easily, (ii) the sphere surface has constant curvature, and thus enables quantitative comparison of different techniques with respect to local surface properties, and (iii) any viewer knows how a sphere should look, which enables reliable visual confirmation.

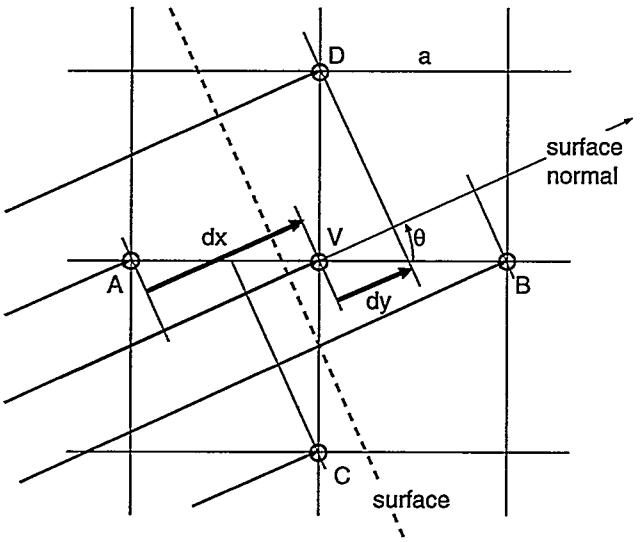


Figure 1: *Computation of the surface normal.*

Typically, a sphere is projected by means of ray casting and error measures are then mapped onto the projected image. These error measures usually reflect errors in surface position and surface normal. Surface position is detected first, and subsequently, a normal error is estimated for this surface point. However, due to a sharp angle between the surface and the ray, error in position estimation is higher near the silhouette of the sphere image than near its center. This error is transferred to the estimation of the normal error and deteriorates its value. In order to minimize mutual influences between precision of the surface and normal errors, a different configuration of the experiment has been used. Rays are now shot uniformly from the center of the sphere in all directions. Thus, the surface estimation error is minimized and its impact on the normal error estimation can be neglected.

For each ray, a precise intersection point $\mathbf{p}_i$ with a continuous surface, defined by the trilinear interpolation of 8 surrounding samples and thresholding at density level 0.5, is found. Then, for each of the 8 samples a normal vector is estimated and the obtained normals are interpolated for the point $\mathbf{p}_i$ by trilinear interpolation again. Four different gradient estimators have been tested:

`central` : the standard central differences technique,

`adapt` : the adaptive technique (Eq. 5),

`goss4` : adjustable Goss filter [24], $\alpha = 4.0$, and

`goss16` : adjustable Goss filter, $\alpha = 16.0$.

We decided to use the trilinear filter in spite of its poor frequency properties, which manifest themselves predominantly in cases when the sampling frequency is near the Nyquist rate. In this case higher order filters have proven to be more suitable [21]. Yet, our primary intent is to study the algorithm performance for "big" objects, i.e., that are significantly larger than the characteristic dimension of the sampling filter r_f. In this case, the error introduced by the trilinear filter is minimal and we can take advantage of its significantly lower computational demands.

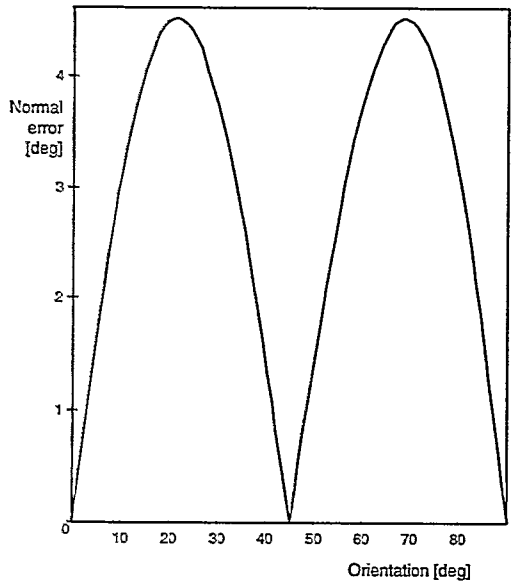


Figure 2: *Theoretical error estimation of the surface normal for a Gaussian filter with* $\sigma = 1$.

Experiment 1: Our first goal has been to investigate the influence of size and type of the filter kernel on the precision of the surface point estimation. Three filter types have been compared:

`cone` : the cone filter (Eq. 1),

`gauss` : the Gaussian filter (Eq. 2) with planar approximation of the surface, and

`box` : the oriented box filter (Eq. 4).

Several theoretical density profiles of objects convolved by different filters are depicted in Figure 4. The exact position of the surface point can be obtained by thresholding at density level 0.5. We can see that there is no shift in surface position for the `gauss` and `box` filters. However, the situation is different for the `cone` filter. Profiles obtained by convolving spheres with the cone filter are nearly independent on the sphere radius, if the sphere radius is 3 or more times larger than the filter radius and very near to the profile obtained by the Gaussian with $\sigma = 0.8$. However, for smaller sphere diameters the differences are significant and the estimated surface point is shifted into the object. This error, which increases with the increasing filter size, is caused by averaging the object within the volume of the filter support. Since there is no such averaging in the case of the Gaussian and box filters, there is also no error.

Experimental results showing dependency of the mean distance error on the surface curvature (defined by means of the test sphere radius) are depicted in Figure 5. For sphere radii larger than the filter size, the figure confirms our theoretical observation, since for both the Gaussian and box filters the surface position error is significantly lower than in the case of the cone filter. Nonetheless, there is some error, albeit minimal. This can be explained by blurring, which is introduced by sampling and subsequent reconstruction by the trilinear filter. The error is below 1/100 of a voxel unit, supporting our assumption that it is unnecessary to use higher order filters for reconstruction.

Behavior of all three curves dramatically changes for sphere radii below the filter size. The abrupt increase of the error to positive values signals that object details with characteristic radii, approximately equal to the filter radius, are smoothed and therefore cannot

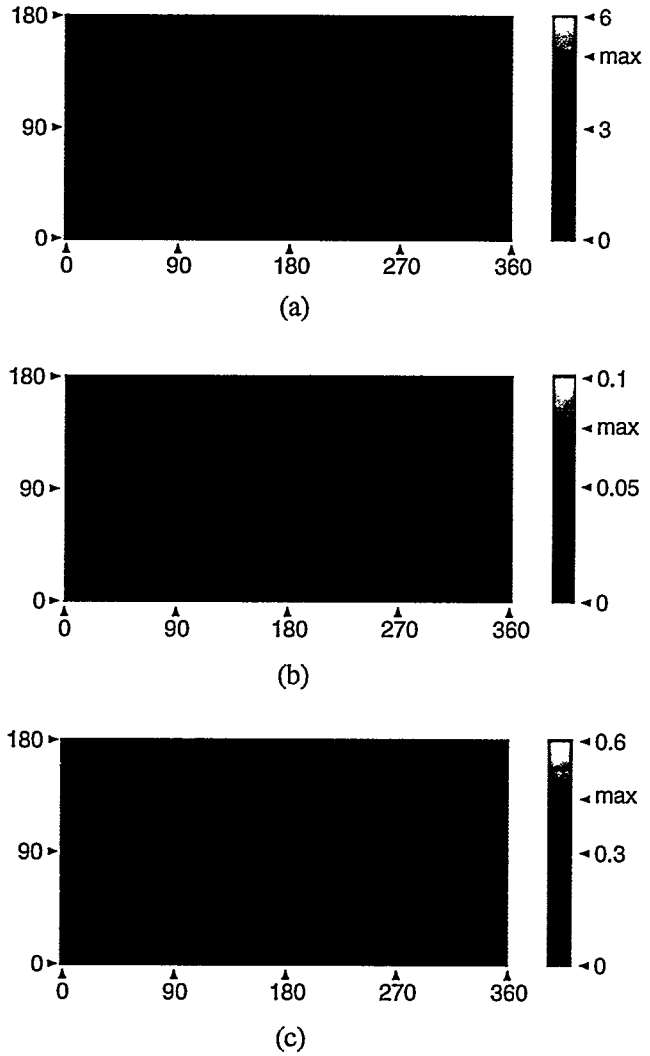


Figure 3: *Experimental error estimation of the surface normal (in degrees) as a function of a sphere parameterization for (a) a Gaussian filter with* $\sigma = 1$*; (b) oriented box filter* ($r_f = 1.8$) *with* `float` *precision; and (c) same as (b) with* `unsigned char` *precision (see also Color plates).*

be rendered correctly. The minimal size of these details can be to some extent decreased by narrowing the filter support. However, as we show later, shrinking the filter results in increasing the surface normal error. Therefore, improved precision of details can be achieved only by increasing the sampling rate, i.e., by increasing the volume of the voxelized data set (Color plates, Figure 10).

Experiment 2: The goal of the second experiment was to verify the theoretical conclusions concerning the dependency between the normal direction and filter type (Section 3). Figure 3 shows the distribution of the normal error for a sphere with a radius of 51.2 voxel units, voxelized with both *float* precision (Figure 3a,b) and *unsigned char* precision (Figure 3c). A Gaussian filter with $\sigma = 1$ was used for voxelization in case (a), and the oriented box filter was used otherwise. The figures depict error distribution expressed as an angle between the ideal and estimated surface normals on a sphere surface. Surface point coordinates are expressed for a standard sphere parameterization on the domain $(0, 2\pi) \times (0, \pi)$. We

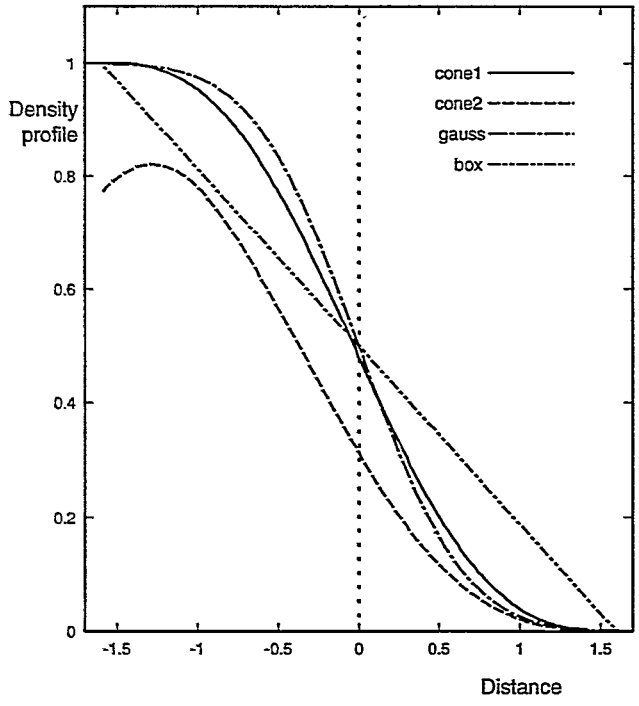


Figure 4: *Simulated theoretical density profiles of objects, convolved by different filters with support radius 1.6.* `cone1`*: Sphere, radius 12.8, cone filter;* `cone2`*: Sphere, radius 1.28, cone filter;* `gauss`*: a halfspace object, Gaussian filter with* $\sigma = 0.8$*;* `box`*: a halfspace object, oriented box filter.*

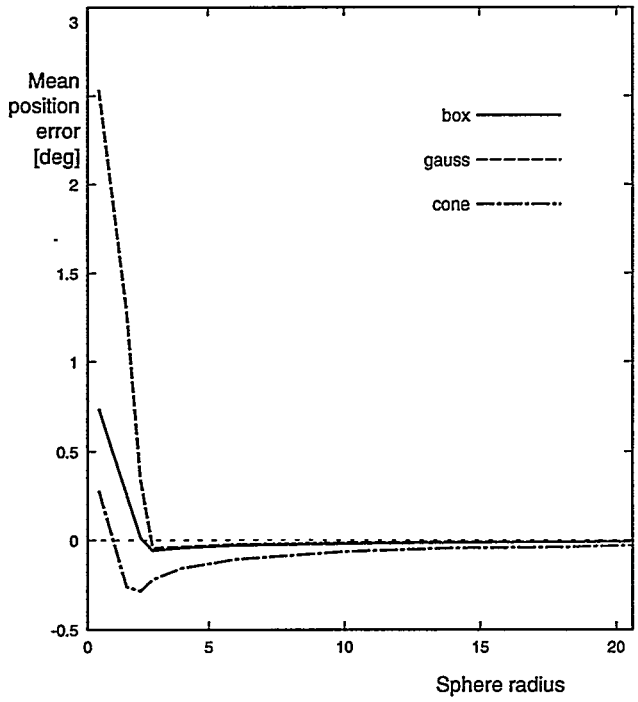


Figure 5: *Experimental results for the dependency of detected surface position on object curvature* ($r_f = 1.8$).

can see that the error is about 50 times higher for the Gaussian profile, with minima and maxima precisely corresponding to the theoretical results in Figure 2. Estimated maximal error (4.8 degree) is also very close to the theoretical estimation (4.5 degree). Furthermore, we can see that there is a difference between results obtained by storing the voxel densities with float and unsigned char precision. Yet, the result for the char precision is still more than 10 times better than for the Gaussian profile in float precision.

Figure 6 shows dependency of the mean normal error with respect to the filter support radius, computed for the central differences normal estimator. For the box filter, the error falls to negligible values for $r_f > 1.8$ (which means that the thickness of the transient region between "inside" and "outside" is 3.6). In this case, the $3 \times 3 \times 3$ filter (applied to samples at the grid vertices and occupying $2 \times 2 \times 2$ volume units) fully fits into the transient region, since the length of its diagonal is $2\sqrt{3} = 3.46$. For smaller r_f, the vertex samples of the filter fall outside of the linear region, introducing an error similar to the case of nonlinear profiles.

In contrast, the error for the two nonlinear profiles is significantly larger even for larger filters. This error decreases, as the profile gets more linear with the growing size of the filter support.

Experiment 3: The last experiment was aimed at revealing the optimal filter support radius for different gradient estimators. Results summarized in Figure 7 show that the optimal filter size is smaller for filters with smaller kernel sizes (the `adapt` filter has the smallest and the `goss4` has the largest kernel). The error again depends on how well the filter kernel fits into the linear transient region. The `central` filter shows the best results for thick transient regions. However, for thin transient regions the frequency properties of the filter become important, and the best results are obtained with the `adapt` filter. Unfortunately, a systematic error, introduced

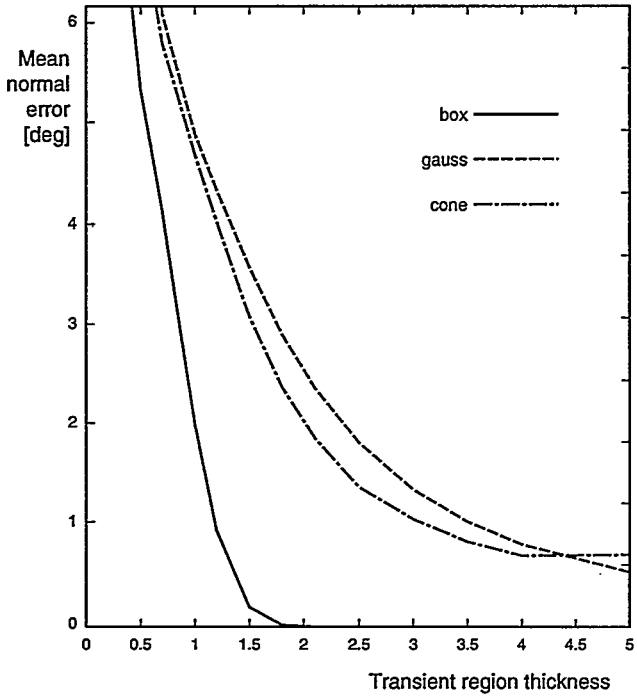


Figure 6: *Experimental results for the dependency of the mean normal error on filter support radius for different filter functions (sphere diameter 51.2).*

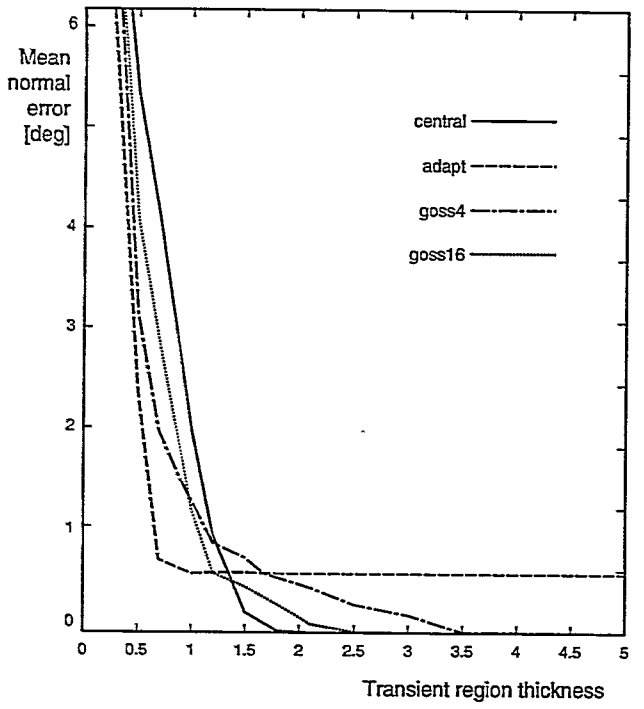


Figure 7: *Experimental results for the dependency of the normal error on filter support radius for different gradient estimation schemes (sphere diameter 51.2).*

by taking the maximum of forward and backward differences into account, deteriorates its performance for thick transient regions.

The aforementioned experiments have shown that the voxelization method of choice is filtering by the oriented box filter, provided that the the object details can be approximated by a sphere of a radius not smaller than 2 voxel units. The optimal radius of the filter is 1.8 and central differences should be used to compute the surface gradient.

This technique should be also preferred for objects with sharp details, in spite of the fact that the volume sampling technique using the cone filter gives better results for small details (i.e., sharper edges). The latter technique has a drawback, however. Due to the nonlinear density profile introduced by the cone filter, estimation of the surface normal is charged by an error, leading to more serious artifacts than slightly smoother edges.

The situation is different in the case of thin objects, such as curves. In this case, normal precision estimation is not so critical and therefore it is possible to use filters with smaller support, even if it results in less precise normal estimation. It can be shown that if the oriented box filter with radius 0.8 is used with the adaptive normal estimation, the surface position error for a sphere with radius 1.5 is still below 0.1 voxel unit, and the surface normal is estimated with error lower than 10 degrees.

5 Voxelization of Parametric Surfaces

For each sample point, the technique described in the previous section requires knowledge of its distance to the surface of the voxelized primitive. In some cases it can be computed easily (sphere, plane), and in others the object can be built by CSG operations between primitives (e.g., polyhedra by intersection of halfspaces, torus by union of many spheres). Distance from the surface for implicit solids $g(\mathrm{p}) = 0$ can be estimated directly from the function and its gradient:

$$d(\mathrm{p}) = \frac{g(\mathrm{p})}{||\nabla g(\mathrm{p})||}. \tag{16}$$

The situation is quite different for parametric surfaces. For each sample point of the data set, its distance to the primitive has to be computed. The computation often involves solving a system of nonlinear equations. This iterative procedure is often instable and, if there are several possible solutions, the nearest one is not always obtained.

Fortunately, the order of integration in Eq. 3 can be reversed. Let the surface be parameterized by equations $(x(u,v), y(u,v), z(u,v))$, where (u,v) are defined on a 2D interval, $(u,v) \in \mathcal{I}$. Then, the convolution Eq. 3 becomes

$$S_f(\mathrm{p}) = \iiint \delta(x - x(u,v), y - y(u,v), z - z(u,v)) f(p_x - x, p_y - y, p_z - z)dx\,dy\,dz, \tag{17}$$

where $\delta(\cdot)$ is Dirac's delta function. This equation simplifies to

$$S_f(\mathrm{p}) = \iint_{\mathcal{S}} f(p_x - x(u,v), p_y - y(u,v), p_z - z(u,v))du\,dv \tag{18}$$

From Eq. 18 we can see that the convolution can be accomplished by summing contributions from the whole surface for each grid sample point. This is computationally far easier, since the surface is only a 2D subset of the volume and the filter has only limited scope due to its finite support. A discrete approximation of this technique is termed 3D splatting [14, 26]. First, the object was point sampled into its binary representation and, second, the contribution of each nonzero discrete point was accumulated to sampling points within the reach of its support. In order to speed up the computation, an $m \times m \times m$ lookup table of the corresponding kernel weights was pre-computed, where m is the size of the discretized filter support in one direction. However, sampling the object with unit steps has proven to be insufficient. Therefore, a superbuffer technique was introduced, increasing resolution of the binary volume n times. Now, the lookup table has been precomputed for each supervoxel, resulting in an $n \times n \times n$ array of $m \times m \times m$ tables, which were used during the second phase to update the samples. Due to extreme memory demands of the algorithm (n^3 increase) only $n = 4$ times higher resolution for models with reasonable resolution was possible.

As we have already shown, in order to get correct images, the transient region near the object surface should have a linear profile. To fulfill this condition, we adopted a different approach, based on filtering by the Bartlett cone filter Eq. 18 and on registering the maximum value of contributions from different surface points instead of summing. To avoid the superbuffer, the algorithm samples the continuous representation of the object directly. However, computation of distances from each surface sample to all voxels in its neighborhood is very costly and therefore unacceptable. Instead, a similar $n \times n \times n$ array of $m \times m \times m$ lookup tables is used as in the superbuffer technique, but now with much higher resolution ($n = 20 \ldots 40, m = 7$). The index to the array is obtained by multiplying the fractional part of a surface point coordinate by n and subsequently truncating the fractional part.

The last question, which still remains open, is the optimal rate of sampling the interval $\mathcal{I}$. Uniform sampling is clearly not the method of choice. We know from differential geometry that unit change of u (v) coordinate of a parametric patch $(u,v) \rightarrow s(u,v)$, leads to change $||\mathrm{s}_u||$ ($||\mathrm{s}_v||$) on the surface, where s_u and s_v are partial velocity vectors, given by

$$\mathrm{s}_u = \left(\frac{\partial x}{\partial u}, \frac{\partial y}{\partial u}, \frac{\partial z}{\partial u}\right)$$

$$\mathbf{s}_v = \left(\frac{\partial x}{\partial v}, \frac{\partial y}{\partial v}, \frac{\partial z}{\partial v}\right). \quad (19)$$

In order to set the sampling rate properly, it is necessary to know the maximum value of both $||\mathbf{s}_u||$ and $||\mathbf{s}_v||$ in the interval $\mathcal{I}$. This value should be estimated numerically and may be connected with a complex search of a global maximum of a nonlinear function [27]. Once estimated, it defines a proper sampling rate only for a limited region of $\mathcal{I}$, while in the rest of $\mathcal{I}$ it leads to useless oversampling.

To avoid these shortcomings, we adopted an adaptive technique, based on a hierarchical subdivision of the interval $\mathcal{I}$. Originally, we tested a quadtree subdivision. The actual patch was divided into four smaller patches, until the patch size s_p reached some predefined value. Then, for each leaf of the quadtree, a single sample was taken in its center. The size s was estimated as

$$s_p = \max(s_u||\mathbf{s}_u||, s_v||\mathbf{s}_v||), \quad (20)$$

where s_u and s_v are dimensions of the actual 2D interval in $\mathcal{I}$. However, results we obtained were not satisfactory, due to large differences between the patch sizes (Figure 8a).

Better results were obtained by an alternative technique based on binary subdivision (Figure 8b). Here, each patch was subdivided into only two smaller ones. The same termination criterion was used as in the quadtree case and the relation between the $s_u||\mathbf{s}_u||$ and $s_v||\mathbf{s}_v||$ was used to decide in which direction to subdivide. Table 1 summarizes results obtained for different objects. We can see, that although the same termination criterion has been used, binary subdivision resulted in a considerably smaller number of patches.

Figure 9 depicts the results of an experiment, aimed at estimating the optimal size of the lookup table array and the sampling density. The graph, showing the dependency of the mean normal error on the number of samples, was obtained for a sphere with diameter 51.2 and a Bartlett filter with radius 1.6. The optimal number of samples was about $3.5 \cdot 10^5$, which corresponds to a step between samples of about 0.5 voxel units, since increasing the sampling density did not bring improvement in the image quality. We can further see that the surface quality improved with larger lookup table arrays. However, for arrays with the size above 40^3 this improvement was negligible.

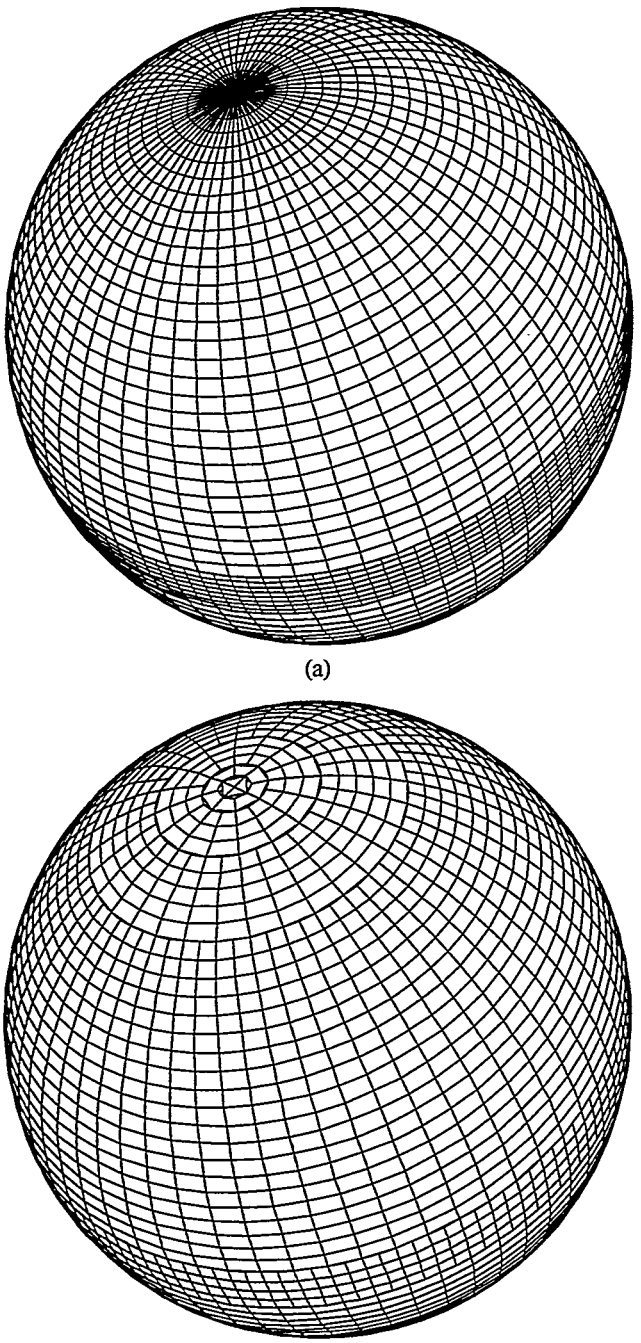
Table 1: *Number of samples for the quadtree and binary approaches for four objects shown in Figure 11 (Color plates).*

object	Figure	quadtree	binary
Monge patch	11a	937984	692224
Moebius strip	11b	1810124	638888
ellipsoid	11c	685404	552094
Bezier patches	11d	1573439	586785

6 Summary

In our theoretical and experimental analysis of filter-based voxelization techniques, we have shown that there is a close relationship between voxelization and normal estimation filters. In order to ensure correct normal estimation, the density profile should be linear within a surface transition area, the size of which depends on the size of the normal estimation filter. We have further shown, that if this condition is fulfilled, surface position and direction of surface normal vector can be estimated with very high precision.

These results have been used in the second part of our work for voxelization of parametric surfaces. An adaptive technique has been proposed, uniformly sampling the surface. In order to fulfill the condition of linearity, the Bartlett cone filter was used, but, instead of summing its contribution within its domain, only the maximum contribution of all samples was taken into account.



(a) (b)

Figure 8: *Sampling of a spherical surface: (a) quadtree subdivision, (b) binary subdivision. Samples are positioned at the centers of the quadrilaterals. (For the sake of clarity, the sampling density for these pictures was set to a significantly lower value than the estimated optimum).*

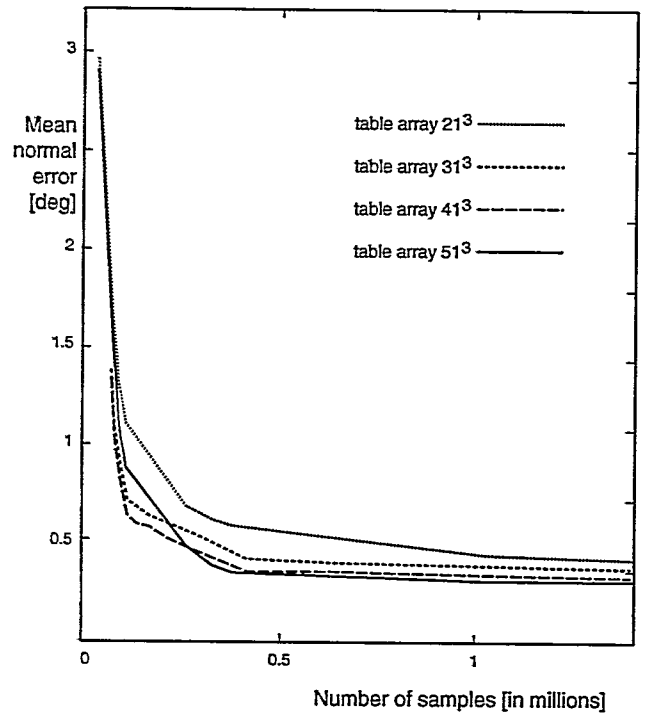


Figure 9: *Estimated normal error as a function of number of samples and size of the lookup table array.*

It has not been our goal to propose computationally optimal routines for filtered voxelization of special parametric surfaces. Rather, we focused on a universal technique, capable of processing any regular surface. However, our approach can lead to longer processing times. Since in practice, some kinds of parametric objects have proven to be more important than others, we are currently optimizing the technique for some special cases (Bezier patches, NURBS).

Acknowledgments This work has been supported by ONR grant N000149710402, NRL grant N00014961G015, and NSF grant MIP9527694.

References

[1] Arie Kaufman, Daniel Cohen, and Roni Yagel. Volume graphics. *IEEE Computer*, 26(7):51–64, July 1993.

[2] Hans-Peter Pfister and Arie Kaufman. Cube-4 — A scalable architecture for real-time volume rendering. In *ACM/IEEE Sympos. on Volume Visualization*, pages 47–54, 1996.

[3] Randy Osborne, Hanspeter Pfister, Hugh Lauer, Neil McKenzie, Sarah Gibson, Wally Hiatt, and Hide Ohkami. EM-Cube: An Architecture for Low-Cost Real-Time Volume Rendering. In *SIGGRAPH/Eurographics Workshop on Graphics Hardware 97*, pages 131–138, August 1997.

[4] Arie Kaufman. An algorithm for 3D scan-conversion of polygons. In *Eurographics '87*, pages 197–208. August 1987.

[5] A. Kaufman, editor. *Volume Visualization*. IEEE Computer Society Press, Los Alamitos, CA, 1991, chapter 5.

[6] Lih-Shyang Chen, Gabor T. Herman, R. Anthony Reynolds, and Jayaram K. Udupa. Surface shading in the cuberille environment. *IEEE Computer Graphics and Applications*, 5(12):33–43, December 1985.

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

[8] Miloš Šrámek. *Visualization of Volumetric Data by Ray Tracing*. Austrian Computer Society, Austria, 1998. ISBN: 3-85403-112-2.

[9] Roni Yagel, Daniel Cohen, and Arie Kaufman. Discrete ray tracing. *IEEE Computer Graphics and Applications*, 12(5):19–28, September 1992.

[10] 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.

[11] Maria Magnusson, Reiner Lenz, and Per-Erik Danielsson. Evaluation of methods for shaded surface display of CT-volumes. *Computerized Medical Imaging and Graphics*, 15(4):247–256, 1990.

[12] Ulf Tiede, Karl-Heinz Höhne, Michael Bomans, Andreas Pommert, Martin Riemer, and Gunnar Wiebecke. Investigation of medical 3D-rendering algorithms. *IEEE Computer Graphics and Applications*, 10(3):41–53, 1990.

[13] Sidney W. Wang and Arie Kaufman. Volume sampled voxelization of geometric primitives. In *Visualization '93*, pages 78–84, San Jose, CA, October 1993.

[14] Sidney W. Wang and Arie Kaufman. Volume-sampled 3D Modelling. *IEEE Computer Graphics and Applications*, 14(5):26–32, September 1994.

[15] Miloš Šrámek. Gray level voxelization: A tool for simultaneous rendering of scanned and analytical data. In *Proceedings of the 10th Spring School on Computer Graphics and its Applications*, pages 159–168, Bratislava, Slovak Republic, June 1994. Comenius University.

[16] Mark W. Jones. The production of volume data from triangular meshes using voxelisation. *Computer Graphics Forum*, 15(5):311–318, December 1996.

[17] Stijn Oomes, Peter Snoeren, and Tjeerd Dijkstra. Transforming polygons into voxels. In *Scale-Space Theory in Computer Vision, Lecture Notes in Computer Science. Vol. 1252*. Springer-Verlag, 1997.

[18] Robert A. Drebin, Loren Carpenter, and Pat Hanrahan. Volume rendering. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4):65–74, August 1988.

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

[20] Karl Heinz Höhne, Michael Bomans, Andreas Pommert, Martin Riemer, Carsten Schiers, Ulf Tiede, and Gunnar Wiebecke. 3D visualization of tomographic volume data using the generalized voxel-model. *The Visual Computer*, 6(1):28–36, February 1990.

[21] Stephen R. Marschner and Richard J. Lobb. An evaluation of reconstruction filters for volume rendering. In *Visualization'94*, pages 100–107, October, 1994.

[22] Jan J. Koenderink. The structure of images. *Biolg. Cybern.*, 50:364–370, 1988.

[23] George Wolberg. *Digital Image Warping*. IEEE Computer Society Press, 1990.

[24] Michael E. Goss. An adjustable gradient filter for volume visualization image enhancement. In *Proceedings of Graphics Interface '94*, pages 67–74, May 1994.

[25] Michael Deering. Geometry compression. *Computer Graphics*, 29(Annual Conference Series):13–20, August 1995.

[26] Sidney W. Wang. *Synthesis, Manipulation, and Modelling of Volume-Sampled Geometric Objects*. PhD thesis, SUNY at Stony Brook, New York, May 1995.

[27] Arie Kaufman. Efficient algorithms for 3D scan-conversion of parametric curves, surfaces, and volumes. *Computer Graphics (SIGGRAPH '87 Proceedings)*, 21(4):171–179, July 1987.