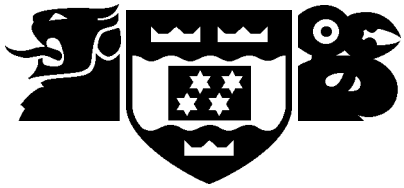
VICTORIA UNIVERSITY OF WELLINGTON



Department of Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 471 5328
Fax: +64 4 495 5232
Internet: Tech.Reports@comp.vuw.ac.nz

Implicit volume rendering of generalised cylinders

Peter Hall

Technical Report CS-TR-94/10
May 1994

Abstract

This paper addresses the issue of rendering the interior volume of generalised cylinders that are filled with a semi-translucent material. A point sampling method is proposed. Volumes that are defined by combining generalised cylinders via set theoretic operations may be rendered also.The original motivation for this work was to simulate an x-ray process that is capable of imaging networks of blood vessels. However, the rendering technique is not restricted to that domain and could be used in more general computer graphic applications. The principal characteristic of this rendering technique is that absorption is the only optical effect that is modelled. The method provides algorithmic simplicity, geometrical accuracy, and computational efficiency.

Publishing Information

A version of this report has been submitted to Computer Graphic Forum.

Author Information

Peter Hall is a lecturer at Victoria University. His principle research interests are scientific visualisation, and computer vision.

Implicit volume rendering of generalised cylinders

Peter Hall
Department of Computer Science,
P.O. Box 600,
Victoria University of Wellington,
Wellington,
New Zealand
Email: peter@comp.vuw.ac.nz

Abstract

This paper addresses the issue of rendering the interior volume of generalised cylinders that are filled with a semi-translucent material. A point sampling method is proposed. Volumes that are defined by combining generalised cylinders via set theoretic operations may be rendered also.The original motivation for this work was to simulate an x-ray process that is capable of imaging networks of blood vessels. However, the rendering technique is not restricted to that domain and could be used in more general computer graphic applications. The principal characteristic of this rendering technique is that absorption is the only optical effect that is modelled. The method provides algorithmic simplicity, geometrical accuracy, and computational efficiency.

1 Introduction

The motivation for the work presented here was to find a mechanism capable of simulating x-ray projections of blood vessels. To simulate such images accurately would require a very careful description of the physical processes involved. For example, when simulating x-rays for treatment planning, transport theory should be used (see Williamson, 1989). Fortunately for the motivating application, the important features of the x-ray images are produced by only one physical process; the absorption[1] of radiation. This allows the simulation of x-rays to be greatly simplified and has led to a mechanism that is sufficiently general to be of wider use in computer graphics.

The rendering method is very simple. It is detailed in section 3 but outlined here. Each generalised cylinder (see section 2) is assumed to be filled with material that partially absorbs radiation. During rendering, a generalised cylinder is approximated by a series of straight sections. Each section is rendered separately by point sampling. Each sample point represents a small but finite volume of the absorbing material. If a sample point lies within the volume of the current section then it *may* be projected onto an image plane. Resting in that plane is a pixel array. Each pixel accumulates the path length between it and the source of the radiation (see section 3.3). When all the generalised cylinders in a scene have been rendered, the fraction of radiation absorbed under each pixel can be estimated from its path length.

The rendering mechanism does not specify that the interior volume be defined in any particular way (this is why a point only *may* be projected should it be interior to a given section). The mechanism will operate satisfactorily on volumes defined by combining generalised cylinders via set theoretic operations. Care is taken to ensure that points samples are distributed at regular

1. ‘Absorption’ as used here is a synonym for attenuation for generally (Mitchell 1994).

intervals within a single frame of reference. This helps ensure that objects that abut or overlap are rendered seamlessly. Because the point samples are all taken from a single frame of reference the method is very similar to volume rendering (see section 2). Indeed, because the method here is so similar to volume rendering I have chosen to call it ‘implicit volume rendering’; implicit because no voxel model is ever held in memory explicitly.

The structure of this paper is as follows. Background literature is outlined in section 2. The rendering method that is the subject of the paper is presented in section 3, and images resulting from it are shown in section 4. The conclusion, section 5, summarises the paper.

2 Background

Generalised cylinders are modelling primitives that have been used in both computer graphic and computer vision contexts; refer to Foley et al. (1990), Ballard and Brown (1982), respectively. A generalised cylinder is a three dimensional object that is specified by sweeping a cross-section over a trajectory in three dimensions. The cross-section may vary in size, shape, and orientation during the sweep. Bronsvoort and Klok (1985) define a generalised cylinder, **G**, by its boundary. The expression they give is

$$\mathbf{G}(u,v) = \mathbf{t}(u) + c_x(v)\mathbf{e}_2(u) + c_y(v)\mathbf{e}_3(u).$$

in which **t** is the trajectory. Let $\mathbf{e}_1$ denote the unit vector that is tangent to **t**. Unit vector $\mathbf{e}_1$ lies in the plane of instantaneous curvature of **t**.Unit vector $\mathbf{e}_2$ is perpendicular to $\mathbf{e}_1$ and also rests in the plane of instantaneous curvature of **t**. The unit vector $\mathbf{e}_3$ is the cross product of $\mathbf{e}_1$ and $\mathbf{e}_2$. Together, the vectors $\mathbf{e}_1$, $\mathbf{e}_2$, and $\mathbf{e}_3$ form a Frenet frame; being the unit tangent, the principal normal, and the binormal respectively. Vectors comprising a Frenet frame are always mutually orthogonal. The functions c_x and c_y describe a contour in the plane defined by $\mathbf{e}_2$, and $\mathbf{e}_3$. This contour represents the cross section, and must be closed. The definition requires two parameters, u and v. The first of these specifies a particular position on the trajectory, the second specifies a particular position on the boundary.The representation is shown in figure 1.

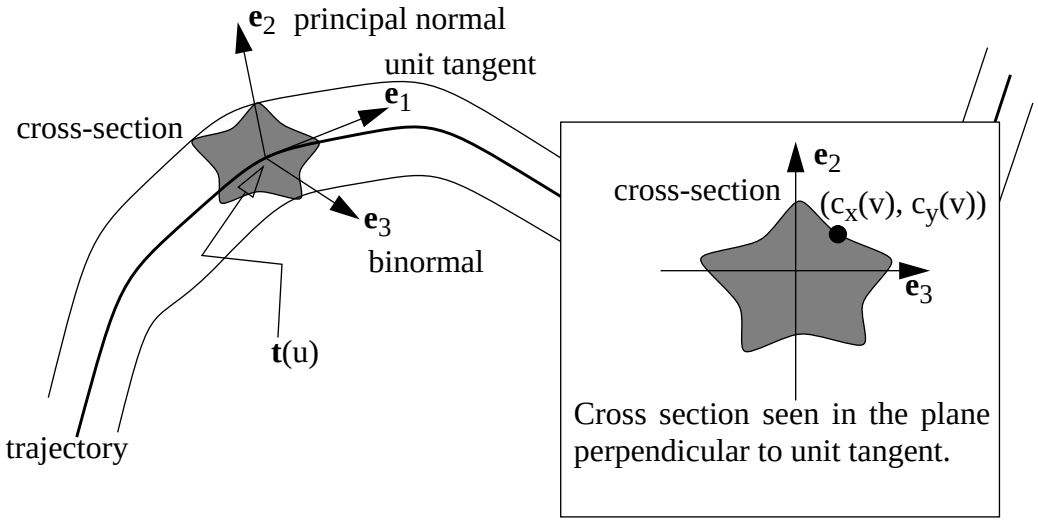


Figure 1: A generalised cylinder.

A generalised cylinder can be rendered by ray tracing, see Bronsvoort and Klok (1985). This option may be developed if an accurate simulation is required. However, ray tracing such objects is computationally expensive. This is because it is difficult to find the intersection of a ray with the surface. Often, analytic expressions that fix these intersections are unavailable, so numerical methods must be used instead. A more efficient method for directly rendering the surface of a generalised cylinder is by point sampling the surface, see Bronsvoort (1992). Surface scanning methods have also been used to render parametric surfaces, Chang et al. (1989). In surface scanning methods a surface is decomposed into a set of sample points and each point is rendered separately. Such algorithms must carefully chose the distance between sample points. The sample points must be close enough to create the impression of a continuous surface in the final image, but not so close to impair computational efficiency. In the specific case of rendering the surface of generalised cylinders this means choosing suitable changes in the parameters u and v.

The scanning method described in Bronsvoort (1992) is suitable for rendering surfaces but not suitable for rendering interior volumes. This is because rendering a volume implies that the process of absorption is being modelled. Consequently the amount of absorbing material that a ray passes through must be estimated. The surface scanning methods fail to produce such an estimate. As mentioned, ray tracing could be used to simulate absorption (amongst other phenomena), but the computational cost is high. This leads us to consider the possibility of volume rendering.

Originally, volume rendering was conceived for producing images of voxel models in the context of scientific visualisation (see Drebin et al., 1988, Levoy 1988). The technique of volume rendering considers each voxel as being filled with a semi-translucent material. Semi-translucency is measured by opacity, say. Typically, the opacity value in a particular voxel depends upon the value of some datum associated with the voxel. An image can be produced in one of two ways. One possibility is ray casting through the volume (Levoy 1988), in which case opacity is integrated along each ray. This answers the question “which voxels contribute to this pixel?”. Alternatively each voxel can be projected onto the image surface, opacity values are accumulated there (Drebin et al. 1988). This answers the question “which pixels does this voxel contribute to?”. In either case the fraction of radiation absorbed between some source and a pixel is recorded in the opacity value of the pixel. Also notice that because the volume of data is defined explicitly is inexpensive to compute field properties, such as gradient. Properties such as these are often used to simulate the presence of a radiation-scattering surface in the data volume; the gradient is normal to the affected surface.

If a generalised cylinder is scan converted into a voxel format then it can be rendered using standard volume rendering methods. Kaufman (1987) provides an efficient mechanism for three dimensional scan conversion. However, the resulting voxel model would loose a great deal of geometrical accuracy because the curved outer surface of a volume is approximated by a set of cubes. This effect may be offset by increasing the resolution of the sample space. However, a rise in geometrical resolution is accompanied by a cubic rise in both the size of the data and the time taken to render it.

The rendering method proposed is based on point sampling. It can be regarded as a volume rendering method. Concerns with geometrical accuracy, computational efficiency, and correct optical modelling of absorption all influenced its design. The method is described next.

3 Volume rendering generalised cylinders by point sampling

The rendering method proposed point samples in three dimensions. Those sample points that are interior to any object in the scene are projected onto the image plane. Each sample point is taken to be the centre of a small sphere - a droplet of absorbing material. The method can be regarded as a volume renderer because it answers the question “which pixels does this droplet (voxel) contribute to?”. Unlike conventional volume renderers the sampling space is never held in memory explicitly. In principle this means that sampling density is not constrained by computer memory; the memory complexity is no greater than is required to store the scene database. In practice, the major constraint is time complexity. Improving the time performance leads to an increase in memory cost, as will be discussed.

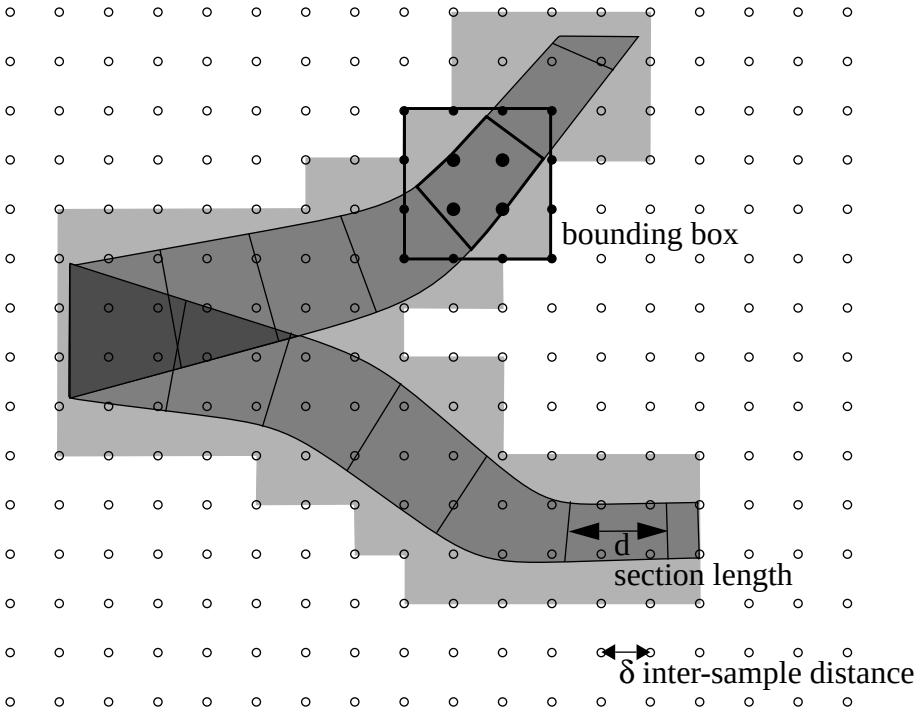


Figure 2: The overall sampling scheme. Generalised cylinders are sliced into sections each of length d (except ‘end sections’). Geometrical accuracy is determined by 1/d. Each section has a bounding box fitted, which is sampled at rate 1/δ. Sampling density is $1/\delta^3$. Only the shaded region is sampled.

Like conventional volume renderers, the time complexity of the method here is cubic with sampling density. Efficiency is improved by point sampling only in the neighbourhood of the generalised cylinders that comprise the scene. Neighbourhood sampling is made possible because each generalised cylinder is approximated by a series of small sections. A bounding box is fitted to each section, and each bounding box is point sampled. This procedure yields an improvement if the objects are sparse within the volume that surrounds all of them. For objects comprising generalised cylinders this is usually the case. The memory cost associated with this method arises because a list of the small sections must be maintained. The size of this list is linear with geometrical accuracy, hence memory complexity is linear with geometrical accuracy. Notice that

geometrical accuracy is distinct from sampling density. Geometrical accuracy is the number of sections per unit length (how finely the generalised cylinders are dissected), sampling density is the number of sample points per unit volume (how finely the sample space is disected).This distinction is seen in figure 2.

An overview of the rendering method has already been given in section 1. This section describes the three major components of the algorithm in detail. In section 3.1. a method for slicing a generalised cylinder into a series of small sections is given. This method reparameterises the trajectory in terms of arc length. The advantages of this are discussed in the same section. Determining whether a sample point is interior to any object in the scene is the subject of section 3.2. Set theoretic methods are used to combine generalised cylinders into new objects. To prevent any point from being rendered more than once the expression used to combine the cylinders is amended slightly. Once a sample point is determined to be interior to some object in the scene it needs to be projected onto an image plane, where path length is accumulated. The mechanisms for this, and for computing the final intensity at a pixel are found in section 3.3.

3.1 Decomposing a generalised cylinder into a series of small sections

To decompose a generalised cylinder into nearly-straight sections it is necessary to consider the trajectory, $\mathbf{t}(u)$, along which the cross section is swept. One way of deciding the straightness of any section is to examine the rate at which the unit tangent to this trajectory, $\mathbf{e}_1(u)$, changes direction. This rate is called the curvature of the trajectory. The value of curvature is zero where the trajectory is straight, and greater than zero where the trajectory is curved. A length of the trajectory can then be chosen as the medial axis for a small section based on the curvature, and some arbitrary straightness condition. This is not done here. Instead the trajectory is decomposed into small sections each with a medial axis sufficiently close to a unit length. Rendering efficiency is impaired but it is simpler algorithm to implement. Moreover, this decomposition is used to reparameterise the trajectory in terms of physical distance along the curve. This reparameterisation used to interpolate changes in the quantities that describe a particular cross section as it sweeps along a trajectory.

Two conditions must be satisfied to decompose the trajectory. The first is that two neigbouring points on the trajectory, $\mathbf{t}(u_i)$ and $\mathbf{t}(u_{i+1})$, are separated by a Euclidean distance that is tolerably close to some given unit distance, d, say. The second is that there should be no other point on the trajectory, $\mathbf{t}(v)$, between this pair of points that also satisfies this distance condition. So, the full condition that must be satisfied is

$$(\mathbf{t}(u_{i+1}) - \mathbf{t}(u_i)| \approx d) \text{ AND } (\text{NOT}(|\mathbf{t}(v) - \mathbf{t}(u_i)| \approx d))$$

where $u_i < v < u_{i+1}$. The decomposition starts at one end of the trajectory, $\mathbf{t}(u_0)$ say. A new point $\mathbf{t}(u_1)$ is found that satisfies that above condition. In general there is no analytic expression that yields a value for u_1. Instead numerical methods such as Newton Raphson iteration must be used (refer to Press et al. 1988 for details on such methods). Once a new point is determined the decomposition procedure uses it as a new starting point. Iteration continues until the trajectory is fully covered. The final section will usually be of length less than the unit distance. This does not effect subsequent processing.

Subdivision generates a sequence of N small sections that partition the generalised cylinder along its medial axis. If L is the arc length of the trajectory then $N = \lfloor L/d \rfloor$, where $\lfloor a \rfloor$ is the largest integer

that is no larger than a. The i^{th} small subsection is bound by a pair of points at $\mathbf{t}(u_i)$ and $\mathbf{t}(u_{i+1})$, and there are N+1 such points. To reparameterise the curve in terms of distance along the curve, values of distance along the trajectory, z_i, are tabulated against values of the parameter values u_i. These distances are accumulated during decomposition. Since the sequence of the u_i is given by

$$u_{i+1} = u_i + du_i \text{ such that } | \mathbf{t}(u_{i+1}) - \mathbf{t}(u_i) | \approx d$$

Each of the parameter values, u_i, has an associated distance value, z_i. These z_i are specified by the sequence

$$z_0(u_0) = 0$$
$$z_{i+1}(u_{i+1}) = z(u_i) + d$$

This pair of sequences provide a direct mapping between the parameter values u_i and distance values z_i. Hence we can write $u_i = u(z_i)$ at these points. An estimate of parameter values between the z_i may be found by interpolation.

As mentioned, one use for this reparameterisation is interpolating quantities that control the shape, or size, or orientation, of the cross section. If the numerical value of any of these quantities were to be determined using the parameter u then they could be treated as additional spatial quantities and incorporated into the equation of a generalised cylinder given above. The effect of this is that the position along the trajectory at which a particular value, of a particular quantity, appears depends on the specification of that trajectory. This is an undesirable effect that is demonstrated in figure 5 of section 4, using radius of cross-section as the quantity. By using physical distance along the trajectory to interpolate quantities that control the cross section this problem is overcome. Again, refer to figure 5 of section 4 for an example. The reparameterisation permits the contour to be deformed as it is swept along the trajectory by methods such as those described by Post and Klok (1986).

Because of the reparameterisation we can write the sequence of N+1 points that partition the trajectory as $\mathbf{t}(z_i)$ rather than $\mathbf{t}(u_i)$. At this point the unit vector $\mathbf{e}_1(z_i)$, is tangent to the trajectory, and the cross section specified by a tuple of parameters $C(z_i)$, say. Consistency with the Bronsvoort and Klok (1985) definition of a generalised cylinder (given in section 2) is maintained since we can write $C(z_i) = c_x(v)\mathbf{e}_2(x_2) + c_y(v)\mathbf{e}_3(x_2)$. A small section, S_i, of the generalised cylinder that is bound by points $\mathbf{p}_1 = \mathbf{t}(z_i)$ and $\mathbf{p}_2 = \mathbf{t}(z_{i+1})$ is defined by the tuple

$$S_i = < \mathbf{p}_1, \mathbf{n}_1, C_1, \mathbf{p}_2, \mathbf{n}_2, C_2 >$$

in which $\mathbf{n}_1 = \mathbf{e}_1(z_i)$, $C_1 = C(z_i)$, $\mathbf{n}_2 = \mathbf{e}_1(z_{i+1})$, $C_2 = C(z_{i+1})$. These tuples can all computed during decomposition and stored on a list associated with the generalised cylinder. Hence spatial complexity rises linearly with geometric resolution, 1/d. A section is depicted in figure 3.

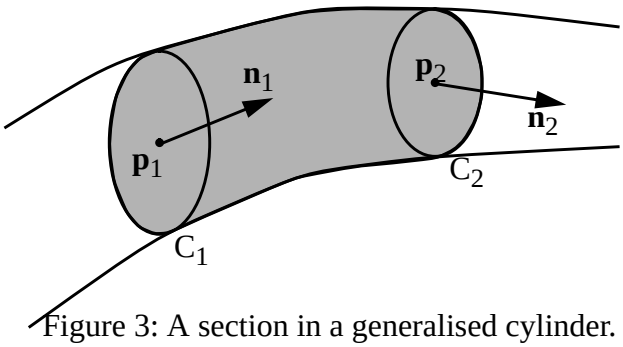


Figure 3: A section in a generalised cylinder.

After it has been sliced a particular generalised cylinder, **G**, is represented by a list, L. We can write

$$\mathbf{G} \cong L$$

to mean that G is approximated by a list L. The list is defined as

$$L = \cup_i S_i.$$

where $\cup_i$ denotes set union over all sections in a given list. This notation is used in the following discussion on determining whether a point is interior to an object.

3.2 Point sampling combinations of generalised cylinders

Point sampling the entire space that contains the generalised is prohibited on practical grounds; it would take far too long. Rather, a bounding box is fitted around each small section and each bounding box is point sampled. Now point sampling occurs only in the neighbourhood of each section, see figure 2 for an example. The bounding box of a small section is formed by merging the bounding boxes of the cross-sections at either end. In effect the bounding box of the bounding boxes is computed.

Finding the bounding box of a particular cross-section requires a centre point **p** = (u,v,w), a unit normal **n** = (a,b,c), a and radius, r, of a circle that bounds the cross-section, C. The cross-section is now considered to be a disc. The high bounds for an origin centred disc is given by the system of equations

$$u_{hi} = r\,((b^2 + c^2) / |\mathbf{n}|)^{1/2}$$
$$v_{hi} = r\,((c^2 + a^2) / |\mathbf{n}|)^{1/2}$$
$$w_{hi} = r\,((a^2 + b^2) / |\mathbf{n}|)^{1/2}$$

This is proven in section 7, which is appendix A. Setting $\mathbf{h} = (u_{hi}, v_{hi}, w_{hi})$ the bounding box ($\mathbf{b}_{lo}$, $\mathbf{b}_{hi}$) is given by $\mathbf{b}_{lo} = \mathbf{p} - \mathbf{h}$, and $\mathbf{b}_{hi} = \mathbf{p} + \mathbf{h}$. This bounding box is for a single disc at one end of the small section. The bounding box for the whole section is found by merging the bounding boxes. If $\mathbf{b}_{lo1} = (x_1, y_1, z_1)$ and $\mathbf{b}_{lo2} = (x_2, y_2, z_2)$ are the lower limits for individual discs then $\mathbf{b}_{lo} = (\min(x_1,x_2), \min(y_1,y_2), \min(z_1,z_2))$ is the lower limit for the whole box. Similarly, individual components of the $\mathbf{b}_{hi}$ are maximised to obtain the upper limits on the bounding box for the whole small section. To ensure that each sample point is a member of the same sample space the limits of the bounded box are expanded so that its edges line up with a virtual sample grid with inter-sample distance δ. If the un-expanded bounding box is $\mathbf{b}_{lo} = (x_{lo}, y_{lo}, z_{lo})$ and $\mathbf{b}_{hi} = (x_{hi}, y_{hi}, z_{hi})$ then the expanded bounding box is

$$\mathbf{b}_{lo} = \delta(\lfloor x_{lo}/\delta \rfloor, \delta\lfloor y_{lo}/\delta \rfloor, \lfloor z_{lo}/\delta \rfloor),$$
$$\mathbf{b}_{hi} = \delta(\lceil x_{hi}/\delta \rceil, \lceil y_{hi}/\delta \rceil, \lceil z_{hi}/\delta \rceil)$$

where $\lceil a \rceil$ is the smallest integer that is no smaller than a.

When a particular section is being rendered its bounding box is point sampled. Each sample point, **x**, within this box is checked against the small section being rendered. A sample point **x** is deemed to be inside the small section, S, if it lies between the planes that bound the section at $\mathbf{p}_1$ and $\mathbf{p}_2$, and is interior to the cross section at some distance z along the trajectory. That is, if

$$((\mathbf{x} - \mathbf{p}_1)\bullet\mathbf{t}_1 >= 0) \text{AND } ((\mathbf{x} - \mathbf{p}_2)\bullet\mathbf{t}_2 > 0) \text{ AND } (\mathbf{x} \text{ is interior to } C(z))$$

is satisfied, where $\bullet$ is the dot product of vectors, and

$$z = z_i + (\mathbf{x} - \mathbf{p}_1)\bullet(\mathbf{p}_2 - \mathbf{p}_1) / |\,\mathbf{p}_2 - \mathbf{p}_1\,|$$

This completes the interior test for a single section. The computation that combines small sections is now considered.

Set theoretic combinations of generalised cylinders are possible. Such a combination operates in the manner of constructive solid geometry (CSG). This is because each small section is a CSG primitive. As discussed above, a particular generalised cylinder **G** is equivalent the union of such primitives. A binary operation, (op), combines generalised cylinders to create a new object. Given a pair of generalised cylinders, $\mathbf{G}_1 == L_1$ and $\mathbf{G}_2 == L_2$, we would like to write

$\mathbf{G}_1$ (op) $\mathbf{G}_2 == L_1$ (op) L_2

There is a subtly in computation though. Potentially, points that are in the intersection of a pair of generalised cylinders will be rendered more than once - because each is rendered separately. The solution is to maintain a dynamic list, X, of small sections that have already been rendered. No point that belongs to this list should be rendered again. Hence the implementation of any binary operation, (op), is in fact

$\mathbf{G}_1$ (op) $\mathbf{G}_2 == (L_1$ (op) $L_2) / X$

where A/B is the difference between sets A and B. In general, an expression, $OP(L_1, ..., L_M)$, that combines M generalised cylinders to create a new object is amended to $OP(L_1, ..., L_M) / X$. Whenever a sample point is found to belong to a small section of one of the lists it is checked against the other lists in the expression, including the dynamic list X. If the sample point satisfies the amended expression then it is rendered. After a particular section has been completely rendered it is added to the exclusion list.The amended expression generates a new model with all internal points rendered exactly once. The effect of the exclusion list on an object comprising a pair of generalised cylinders is seen in figure 6 of section 4.

The efficiency of this sampling scheme rises with the cube of the sample rate, $1/\delta$, and linearly with the geometric resolution 1/d. Sampling bounding boxes reduces the effective volume sampled, as already discussed. Some points are sampled more than once which raises the effective volume - but such points are quickly dismissed because of the ordering of the interior tests. This is discussed further in section 8, which is appendix B. The linear effect can be mitigated if an object can be subdivided into fragments that are known to be mutually exclusive; usually this is not a problem. No exclusion list is required between such instances. To clarify pseudo-code for the process discussed so far is:

```
Decompose each generalised cylinder into a series of sections.
Set an exclusion list to empty
For each generalised cylinder
        For each section comprising the cylinder
                Compute the bounding box for the section
                For each sample point in the box
                        If the sample point is in the section AND
                           the sample point is not in the exclusion list AND
                           the sample point satisfies some set expression
                        Then
                                Render the point
                        Endif
                EndFor (sample point)
                Add the section to the exclusion list
        EndFor (section)
EndFor (cylinder)
```

Rendering a point is the principal omission from the text so far; it is discussed next.

3.3 Rendering a sample point and preparing a final image

Each sample point represents the centre of a small but finite droplet that contains radiation absorbing material at some predetermined level of concentration. For simplicity this droplet is considered to be a sphere. Its radius, ρ, is of the order of the distance between sample points, δ. The inter-sample distance is important; too high and aliasing artefacts set in, too low and rendering efficiency is severely compromised. A reasonable choice for δ is just a pixel length. To mitigate aliasing effects the radius of the droplet is given a suitable value. Experiment showed that $3^{1/2}\delta$, the length of a long diagonal in a sample cube was found give acceptable results. This value ensures that a droplet overlaps all of the 26 nearest droplets that are surround it on a Cartesian grid.

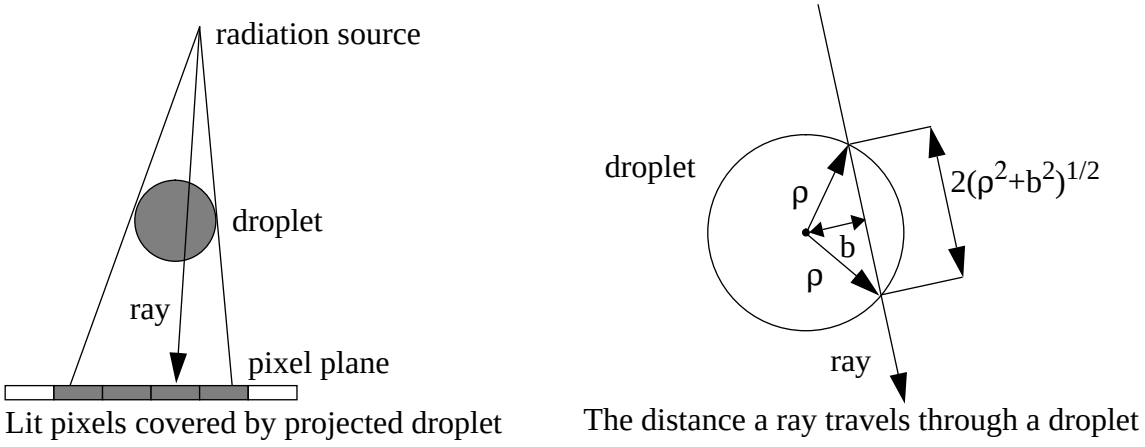
A droplet is projected onto a circle, with centre (u,v) and radius P. Any pixel at (i,j) is covered by the projection of the droplet if $|(u,v)-(i,j)| < P$. Each pixel that is covered by the droplet receives an additional contribution to the path length between it and the radiation source. For a single droplet, the path length of a ray is defined to be the product of the length of the ray through the droplet, s, and the absorption coefficient of the absorbing media in the droplet, μ. The total path length, p(i,j) of a pixel at (i,j) is the sum of such products;

$$p(i,j) = \Sigma_k(s_k\mu_k)$$

where k ranges over each droplet whose projection covers the given pixel. The distance a ray travels through a droplet can found by elementary geometry, $s = 2(\rho^2 + b^2)^{1/2}$, where b is the closest distance between the ray and the droplet centre in three dimensions. In practice this may be estimated at each pixel using either a simpler expression, or via a pre-computed look up table (refer to Westover, 1990 for a discussion of the latter method). Rendering a particular droplet is summarised in figure 4. Once all objects in a scene have been rendered an intensity level, I(i,j), can be computed for each pixel using

$$I(i,j) = I_0\exp(-p(i,j))$$

where I_0 is the initial intensity of the radiation.



Lit pixels covered by projected droplet

The distance a ray travels through a droplet

Figure 4: Rendering a particular droplet.

Notice that this method accounts for absorption only. Moreover, it accounts for absorption at a single wavelength of radiation and so yields monochrome images. A simulation of greater accuracy could be obtained by allowing μ to vary as a function of wavelength. A set of images would then be produced, one at each wavelength. For x-ray simulation a final monochrome image would be

reproduced by resampling these images. Alternatively, a coloured image could be produced, as in standard computer graphics.

Also notice that a point may be simultaneously rendered on more than one image plane with no extra search cost. This contrasts with ray tracing where each projection must be independently rendered. The above discussion focused on rendering a single point in a single image, rendering more than one image is a trivial extension.

4 Results

The results shown are for trajectories that are cubic space curves, thus

$$\mathbf{t}(u) = \mathbf{a} + \mathbf{b}u + \mathbf{c}u^2 + \mathbf{d}u^3$$

and all models are made by combining such sections. These trajectories were specified by defining end points and end tangents.

The importance of the reparameterisation is shown in the table below. Six generalised cylinders were generated. Each had length 80 units, and a circular cross section that varies from 8 to 2 units. The difference in the tubes was their end tangent specification and the variable used in interpolation. Three had radius values interpolated using the parameter u, and three with physical distance, z. The table shows how radius varies with physical distance for each generalised cylinder, values given to 2 decimal places. The head of each column specifies whether parameter, u, or distance, z, was used in interpolation. The ‘x component’ of the tangent is also shown; other components being held constant. These generalised cylinders are shown in figure 5. It is clear that changing the specification changes the curve profile when the parameter is interpolated.

distance, z	u: 100; 100	u: 0.001; 100	u: 100; 0.001	z: 100; 100	z: 0.001; 100	z: 100; 0.001
0	8.00	8.00	8.00	8.00	8/00	8.00
4	7.75	6.95	7.76	7.70	7.70	7.70
8	7.49	6.48	7.53	7.40	7.40	7.40
12	7.23	6.11	7.30	7.10	7.10	7.10
16	6.94	5.78	7.08	6.80	6.80	6.80
20	6.64	5.50	6.86	6.50	6.50	6.50
24	6.33	5.23	6.64	6.20	6.20	6.20
28	6.01	4.97	6.42	5.90	5.90	5.90
32	5.68	4.72	6.20	5.60	5.60	5.60
36	5.34	4.48	5.97	5.30	5.30	5.30
40	5.00	4.25	5.75	5.00	5.00	5.00
44	4.66	4.03	5.52	4.70	4.70	4.70
48	4.32	3.80	5.28	4.40	4.40	4.40
52	3.99	3.58	5.03	4.10	4.10	4.10
56	3.67	3.36	4.77	3.80	3.80	3.80
60	3.36	3.14	4.50	3.50	3.50	3.50
64	3.06	2.92	4.21	3.20	3.20	3.20
68	2.77	2.69	3.89	2.90	2.90	2.90
72	2.50	2.47	3.52	2.60	2.60	2.60
76	2.24	2.24	3.05	2.30	2.30	2.30
80	2.00	2.00	2.00	2.00	2.00	2.00

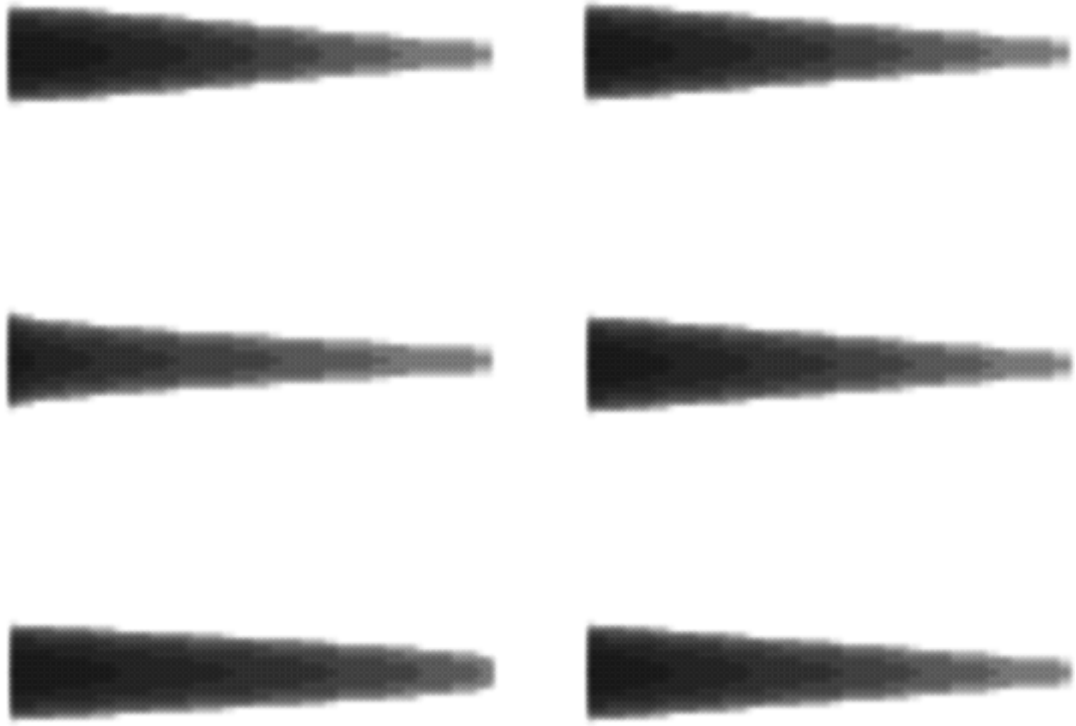


Figure 5: Different profiles are obtained if cross-sectional radius depend on the parameter u (left column). Equivalent profiles are obtained if the radius depends on physical distance, z (right column). The end points of each cylinder are at (-40,0,0) and (40,0,0). The end tangents differ. For each row from the top to bottom these are (100,0,0) and (100,0,0), (0.001,0,0) and (100,0,0), (100,0,0) and (0.001, 0, 0).

A use of the exclusion list is demonstrated in figure 6. There, generalised cylinders are combined using a set union operator, a ‘close up’ of a branch-like structure. The left image shows the effect of omitting the exclusion list from the expression. The right image shows the same operations with exclusion list amendment. It is clear that points in the common volume are rendered more than once unless the exclusion list is used.

A model that was made by arranging generalised cylinders into a graph structure is seen in figure 7. The nodes in the graph are geometrically represented by the union of generalised cylinders with a non-trivial volume of intersection. These are branches. A separate exclusion list was maintained for each branch point. The edges of the graph are geometrically represented by a sequence of generalised cylinders that abut end-to-end. These are limbs. Although edges are defined as the union of generalised cylinders the volume of intersection is zero. No exclusion list was maintained for any branch limb. Generalised cylinders in a limb were made to abut end-to-end with generalised cylinders in a branch. No exclusion list was needed at such points. Seams between limbs and limbs, and limbs and branches, cannot be seen. The union of cylinders that comprise a branch is correctly rendered. The image darkens where the generalised cylinders either cross or have a trajectory that approaches the normal to the image plane. These effects are expected by absorption of radiation

Figure 6: The exclusion list prevents points in the common volume being rendered twice.

Figure 7: A complete model ‘tubular flower’, seen from the top and an arbitrary view.

Recalling that the motivating application was the simulation of x-rays of blood vessels networks, a genuine x-ray is presented in figure 8.
.

Figure 8: An example of a real x-ray of blood vessels.

The simulated images seem to capture the salient features of the real x-ray. In particular, where vessels cross, or where vessels move perpendicular to the plane, or where the vessels are thicker, so the simulated and real images are darker. This is seen despite the simplicity of the rendering model used.

5 Conclusion

A method for rendering the interior volume of generalised cylinders has been presented. The basis of the method is point sampling. Its operation is similar to a volume renderer. The principal

difference is that the sample space is implicitly defined rather than explicitly stored in computer memory. This means that high geometric resolution is available but are that field properties, such as gradient, are expensive to compute. Such properties were not required in the motivating application because absorption of radiation was the was only phenomenon considered. Surface effects such as scattering were ignored, for which ray-tracing could be used. Advantages of the method over ray-tracing are algorithmic simplicity and reasonable time cost. In addition multiple views can be produced simultaneously at little additional cost. In the motivation application this was an important consideration. Typical future work might consider a more wavelength-dependent model for the absorption coefficient. Coherence properties between the small sections that approximate a generalised cylinder might also be considered as the basis for an investigation into improving the time performance of the renderer.

6 References

Ballard, D.H., and Brown, C.M. (1982) *Computer Vision.* Prentice-Hall, Englewood Cliffs, NJ, USA.

Bronsvoort, W., and Klok, F. (1985) Ray tracing generalised cylinders. *ACM Trans. Graph 4(4):* pp 291-303

Bronsvoort, W. (1992) A surface-scanning algorithm for displaying generalised cylinders. *The Visual Computer 8(3)*: pp 162-170

Chang, S., Shantz, M., and Rocchetti, R. (1989) Rendering cubic curves and surfaces with integer adaptive forward differencing. *Compt. Graph 23(3) (Proc. SIGGRAPH)*: pp 157-166

Drebin R.A. Carpenter L. Hanrahan P. (1988) Volume rendering. *Comput Graph 22(4) (Proc. SIGGRAPH)*: 65-74

Foley, J., van Dam, A., Feiner, S., and Hughes, J. (1990) *Computer graphics principles and practice, second edition.* Addison-Wesley, Reading, MA, USA.

Hall, P.M. (1994) *Implicit volume rendering of generalised cylinders.* Tech. Report CS-TR-94/10. Victoria University of Wellington, Wellington, New Zealand. Email Tech.Reports@comp.vuw.ac.nz

Kaufman A. (1987) Efficient algorithms for 3D scan-conversion of parametric curves, surfaces, and volumes. *Comput. Graph. 21(4) (Proc. SIGGRAPH)*: 171-179

Levoy, M. (1988) Display of surfaces from volume data. *IEEE Computer Graphics and Applications*, May ‘88: 29-37

Mitchell, A. (1994) Personal Communication.

Post, F.H., and Klok, F. (1986) Deformations of sweep objects in solid modelling. *Proc. Eurographics 1986*: 103-114

Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. (1988) *Numerical recipes in C, The art of scientific computing*. Cambridge University Press, Cambridge, England.

Westover, L. (1990) Footprint evaluation for volume rendering. *Comput. Graph. 24(4) (Proc. SIGGRAPH)*: 367-376

Williamson, J.F. (1989) Radiation transport calculations in treatment planning. *Comput. Med. Imag. Graph 13*: pp 251-28

7 Appendix A

To prove the an origin centred disc, radius r, resting in a plane with unit normal **n** = (a,b,c) has upper bounds given by

$u_{hi} = r\ (\ (b^2 + c^2) / |\mathbf{n}|\ )^{1/2}$,
$v_{hi} = r\ (\ (c^2 + a^2) / |\mathbf{n}|\ )^{1/2,}$
$w_{hi} = r\ (\ (a^2 + b^2) / |\mathbf{n}|\ )^{1/2}$.

Consider the disc to be the intersection of an origin-centred sphere, radius r, and a plane with normal **n**. The equation of the sphere is

$f(x,y,z) = x^2 + y^2 + z^2 - r^2$,

and the equation of the plane is

$g(x,y,z) = ax + by + cz$.

Now, **grad**(f) is everwhere perpendicular to the surface of the sphere, and **grad**(g) is normal to the plane. Hence the vector product

grad(f) * **grad**(g) = (bz - cy, cx - az, ay - bx)

is a vector that is tangent to the disc.

At extrema in x the x component of this tangent vanishes. Hence

$bz - cy = 0$.

Using this expression, and g() and f(), terms containing y and z can be eliminated leaving

$x = \pm r\ ((b^2 + c^2) / |\mathbf{n}|)^{1/2}$.

The high bound in x is the positive root, the low bound in x is the negative root.

Analogous processes are used at extrema in y, where cx - az = 0; and at extrema in z, where ay - bx = 0. The result in each case is the solution of a quadratic, and the positive root is the high bound along the relevant dimension. This completes the proof.

8 Appendix B

The time complexity of the algorithm is deduced. Each point sample is tested for volume membership in a time

$T_{test} = T_{section} + T_{xlist} + T_{expr}$.

In which $T_{section}$ is the time taken to test he sample point against the current section, T_{xlist} is the time taken to test against the current exclusion list, and T_{expr} is the time take to test against the current expression - which is a list of sections. The time taken to test for a section is taken to be the unit time interval here because that test is used in all other tests. For every section we have

$T_{sect} = 2(T_{subtract} + T_{dot\text{-}product}) + T_{interior\text{-}test}$

where $T_{subtract}$ is time taken for vector subtraction, $T_{dot\text{-}product}$ is the time taken for vector dot product, and $T_{interior\text{-}tes}$ is the 'time fot the interior to cross section' test.. For T_{xlist} we have in the worst case

$$T_{xlist} = \kappa_{in\text{-}sect}\, N_{xlist}\, T_{sect}$$

in which N_{xlis} is the number of sections on the current exclusion list, and κ_{in} is a term that allows for the possibility that the list is not checked, which happens if the 'in current section' test succeeds. So

$$\kappa_{in\text{-}sect} = \begin{cases} 1 \text{ if the current sample is in the current section,} \\ 0 \text{ otherwise.} \end{cases}$$

Similarly, a worst case expression for T_{expr} is given by

$$T_{expr} = \kappa_{in\text{-}xlist} N_{expt\text{-}list} T_{sect}$$

in which $N_{expt\text{-}list}$ is the number of sections in the expression, and $\kappa_{in\text{-}xlist}$ is a term that allows for the possibility that the list is not checked, which happens if the current sample point is not in the exclusion list. So,

$$\kappa_{in\text{-}xlist} = \begin{cases} 0 \text{ if the current sample is in the exclusion list,} \\ 1 \text{ otherwise.} \end{cases}$$

Substitution gives the worst case time complexity for a sample point:

$$T_{test} = T_{sect} (1 + \kappa_{in\text{-}sect}\, N_{xlist} + \kappa_{in\text{-}xlist} N_{expt\text{-}list})$$

the average case complexity will depend how far a search along the exclusion list and the expression list must proceed before a membership decision if made. The answer to this question is data-dependent, and is not developed here.

This test is carried out for each of the sample points in each bounding box. These boxes may overlap - so the same point may be sampled more than once. The number of points sampled, $N_{sampled}$, is just the sum of bounding box volumes divided by the unit sample resolution cubed:

$$N_{sampled} = (\Sigma_i\, V_i)/\, \delta^3$$

where V_i is the volume of the i^{th} bounding box. Thus the time complexity is given by

$$T = N_{sampled}\, T_{test}.$$

Clearly, the efficacy of the method depends on reducing the number of sampled points and ordering the tests so that membership is decided as early as possible.