The Antialiasing Problem in Ray Tracing

Don P. Mitchell

AT&T Bell Laboratories
Murray Hill, NJ 07974

1. Aliasing

Ray tracing is one of the most general and powerful techniques for synthesizing realistic images. Unfortunately, this method suffers from a notoriously difficult problem with aliasing. This survey will consider why the problem is hard, several classes of solutions, and some of the issues that require further study. Before discussing ray tracing in particular, some general concepts of images and signals will be reviewed.

An image is an example of a two-dimensional signal, and can be in one of several different representations. For example, in a television camera, a lens system focuses an optical image onto a target. That image is a two-dimensional region of varying light intensity and color. When the target is scanned, an electrical signal is produced which is "analogous" to the optical image. In this paper, both the optical and the electrical representations will be called *analog images*, because they both consist of continuously variable physical quantities.

A second possible representation is that of a *symbolic image.* For example, a "zone plate" pattern can be expresses as a two-dimensional function, $I(x, y) = sin(x^2 + y^2)$. Symbolic images may be continuously variable, but they consist of mathematical expressions or computer programs rather than tangible physical quantities.

A *digital image* is a two-dimensional lattice of uniformly spaced samples of color, called pixels. Unlike the analog image, a digital image consists of discrete numerical quantities giving the value of an image at a finite number of sample points. These numerical quantities also have finite precision (typically eight-bit integers for each color channel), and this introduces *quantization* effects. Eight bits seems to be adequate, but for display systems with only one or two bits of precision (e.g., bitmap terminals and printers), quantization is a dominating effect.

Many image computations involve transformations from one of these representations to another. In a typical computer-graphics scenario, a user specifies an image symbolically, a renderer (e.g., a ray tracer) converts that symbolic representation to a digital image, and a display system (e.g., frame buffer and CRT) converts the digital representation to an analog image. An image-processing scenario might consist of first converting an analog image to digital, performing a computation on the digital image, and then converting back to an analog image in a display.

Visible defects in images can be introduced if there are errors in changing representation. Of particular interest are errors introduced when an image is digitized (or redigitized) by sampling. Conditions under which a signal can be sampled without loss or distortion of information are described by Shannon's Sampling Theorem [Shannon49]. It states that a signal whose spectrum contains no energy above some frequency W can be recovered without loss from samples taken at a rate of $2W$ or more.

For a two-dimensional image, the sampling theorem can be generalized to apply when the image spectrum contains no energy outside of some bounded area in frequency space [Petersen62]. Digitizing the image requires sampling it periodically, and the spectrum of a sampled image consists of periodically spaced replicas of the spectrum of the original image (see: Figure 1). The replicas are sometimes called "sidebands", and the original spectrum at the center is called the "baseband".

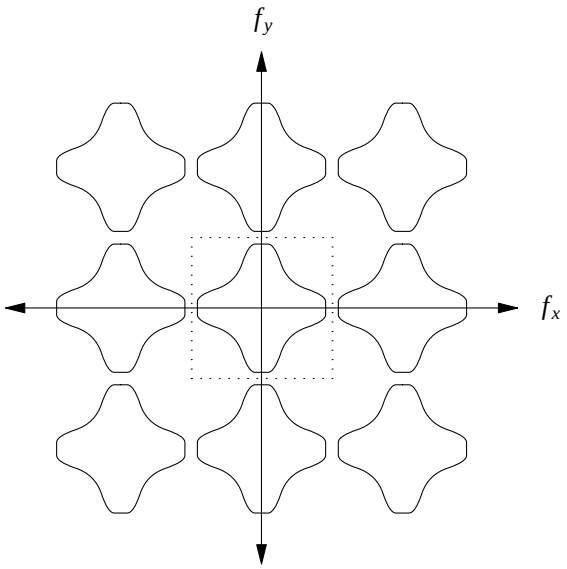


Figure 1. Spectrum of Sampled Image

If the samples are more closely spaced, the replicas of the spectrum will be wider apart. If the samples are not spaced closely enough, the spectral replicas will crowd closely together and overlap. In that case, the conditions of the two-dimensional sampling theorem are violated, and the resulting unrecoverable distortion of the image is called *aliasing.*

Converting a digital image back into a continuous image (either analog or symbolic) is called *reconstruction,* which can be accomplished by a low-pass *postfilter.* The dotted box in Figure 1 indicates a region of the spectrum which a reconstruction filter should pass, while blocking out the replicas. In the spatial domain, reconstruction can be thought of as a smooth interpolation of the samples. Some problems can occur if the reconstruction filter allows some of the replicas to leak thru. This is called "postaliasing" or "imaging". An example of postaliasing would be visible scan lines in an overfocused CRT.

The sampling theorem suggests two ways to prevent aliasing: the image can be sampled densely enough to prevent spectral overlap, or the image can be *prefiltered* to clip the spectrum within a smaller bounding area. Most often, prefiltering is the preferred solution, with the sampling rate fixed by the display or limitations in transmission. How this is accomplished depends very much on the representation of the source image; analog, digital, or symbolic.

Aliasing of images was first encountered when people began to scan and digitize analog images for television and facsimile transmission [Mertz34]. In this case, the antialiasing problem has been essentially solved. Analog images can be prefiltered by physical manipulations of the signal. For example, high frequencies can be removed by defocusing lenses, by the finite beam aperture in a vidicon camera, by the integration of light over the finite area of a CCD senser, or by analog electronic filtering after the photodetection stage of a camera.

A second type of antialiasing problem occurs when a digital image must be resampled at a different rate. In this case, proper prefiltering is accomplished by designing digital filters in which output pixels are weighted sums of input pixels. This is a fairly mature science now [Crochiere75]. Filter design for images also involves a variety of subjective image-quality considerations in addition to antialiasing considerations [Schreiber85, Mitchell88].

Finally, there is the case of converting a symbolic image to digital form. Graphical rendering programs, such as ray tracers, face this problem, which appears to be more difficult and wide-ranging than analog or

digital antialiasing [Crow77].

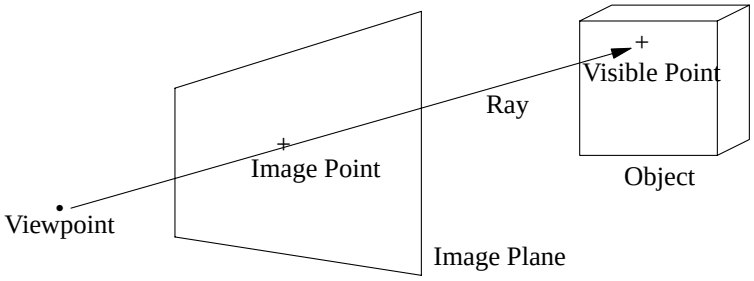


Figure 2. Basic Ray Tracing

Figure 2 illustrates a simple view of the ray-tracing procedure. A point (x, y) can be selected on the image plane, a ray cast from the viewpoint thru that point, intersection and shading calculations follow, and some color value results. For purposes of this discussion, it is useful to define the composition of these processes as the symbolic *image function* $I(x, y)$. The image function can be evaluated at continuously variable points (x, y) and returns a color value and perhaps some additional information from the object space. The end product of a ray tracer is a digital image, which must be produced from the symbolic image function—without aliasing.

2. Symbolic Prefiltering Methods

In the cases of analog and digital antialiasing, prefiltering is the standard solution; however, symbolic prefiltering is a difficult course of action. Catmull's antialiased scan-conversion algorithm is a good example [Catmull78], but for ray tracing, most practical antialiasing algorithms are based on resampling. Some approaches to antialiased ray tracing can be categorized as attempts to prefilter in the symbolic domain. These methods are interesting, and (even if not entirely successful) they help us to understand and appreciate the difficulty of the problem.

The desired low-pass prefilter kernel will be denoted by $k(x, y)$ and the sampling process will be represented by modulation with Dirac δ-functions $\delta(x, y)$. Then sampling a prefiltered image function can be expressed as a convolution of the filter with the image function, followed by multiplication by the δ-function. It is useful to notice that these operations can be combined in two possible orders:

$$pixel_{xy} = \int \delta(x, y) \cdot (k * I) \tag{1}$$

$$= \int (\delta(x, y) * k) \cdot I$$

Put in the most intuitive terms, this suggests dual approaches to prefiltered ray tracing, in which either the objects are filtered or the rays are filtered.

2.1. Cone Tracing

Cone tracing is an example of the ray-filtering approach [Amanatides84]. As indicated in Figure 3, the object space is probed with a finite-width cone instead of a ray. The intent is to prefilter by computing the integral of the image function within a circle on the image plane.

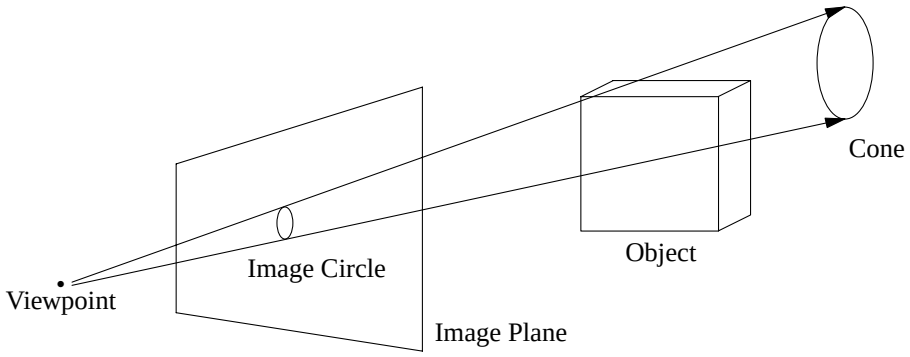


Figure 3. Cone Tracing

Exact symbolic integration of the image function would be practically impossible, but cone tracing programs approximate this result. If the cone comes in contact with an object, the projected silhouette area within the image circle is computed as shown in Figure 4. A limitation of the cone tracing algorithm is that this calculation is not easily performed except for simple objects such as spheres and polygons.

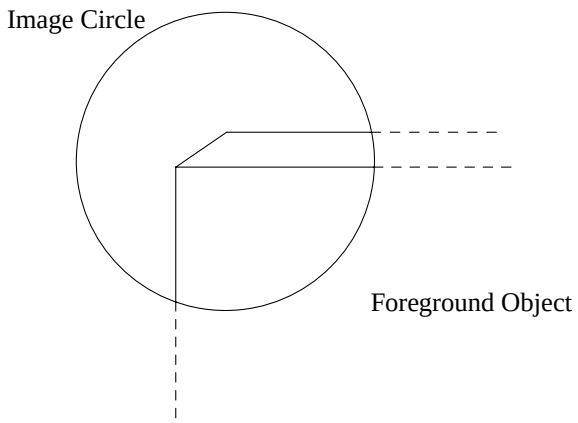


Figure 4. Projected Silhouette In Cone Intersection

Some practical approximations are made in evaluating an antialiased pixel value from a cone sample. First, the cone is not subdivided in cases like the one in Figure 4—a clipped cone is not cast into the background. Instead, the background scene is sampled with the whole cone, and then the silhouette area of the foreground object is used to proportionally blend the background color with the foreground color.

Evaluating the color of a foreground object involves some additional approximations. Exact integration of the shading model over the visible portion of the object surface is also impractical. If the object is texture mapped, the texture can be prefiltered using information about the silhouette area. If the object is reflective, a secondary cone can be cast from some representative point on the object, using a cone angle computed form the local surface curvature and shininess. In other cases, the color of the foreground object may be determined by shading a representative point on the surface.

Cone tracing has produced some striking images, such as Plate 1. It is one of the most efficient methods yet demonstrated for producing the diffused-reflection effect (seen in the smaller sphere in Plate 1) and soft shadows. Kirk has extended the method to handle bump mapped surfaces [Kirk87]. Cone tracing is still limited by the difficulty of cone intersections, and the possible artifacts caused by the various integration

approximations have not been fully studied.

2.2. Beam Tracing

Heckbert and Hanrahan invented a rendering algorithm called *beam tracing*, which is a hybrid of polygon scan conversion and ray tracing [Heckbert84]. If a scene is made up of polyhedral objects, one can use an object-space visible surface algorithm to divide the screen into non-overlaping visible polygonal regions [Weiler77]. Some of these scan-conversion algorithms perform antialiasing by symbolic prefiltering [Catmull78, Duff89].

Heckbert and Hanrahan show that if some of the polyhedral objects are reflective, it is possible to render recursively. When a reflective visible polygon is encountered, the scene is transformed and rendered from the virtual viewpoint and clipped to the polygon. A variant of this approach is described by Garcia [Garcia86].

Combining the beam-tracing technique with Duff's exact polygon convolution algorithm may give the most sophisticated rendering algorithm in which antialiasing can be performed by exact symbolic prefiltering.

2.3. Ray Tracing with Covers

Another approach to symbolic prefiltering is to filter the objects instead of the ray. One way of doing this is by adding *covers* to objects. Whitted first suggested the idea of putting invisible covers around small objects to make sure they were not missed by course sampling [Whitted80]. This idea is extended by Thomas, Netravali and Fox to handle other forms of aliasing [Thomas89]. In their scheme, (portrayed in Figure 5) a covered object has invisible inner and outer shells, which are not rendered but provide information to the ray-intersection routine which can be used for antialiasing.

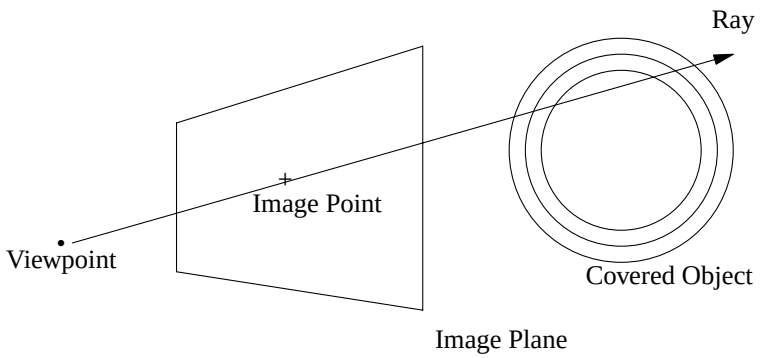


Figure 5. Ray Tracing with Covers

The inner and outer covers are used to *detect* situations where special computation is needed to antialias edges. Three situations are considered: the silhouette of an object, the curve of intersection between overlapping objects, and the edges of shadows.

Antialiasing silhouette edges is the simplest case. If a ray passes thru the outer cover without passing thru the inner cover, then the antialiasing calculation is done; otherwise a standard unfiltered ray-casting calculation is done. The covers are adjusted so that this detection zone, when projected onto the image plane, is about twice the width of the pixel spacing.

Once edge proximity is detected, the antialiasing calculation starts by finding the minimum distance between the image point and the projected silhouette edge. Based on that distance, filtering is done by linear interpolation between the background color and the color of the intersected object.

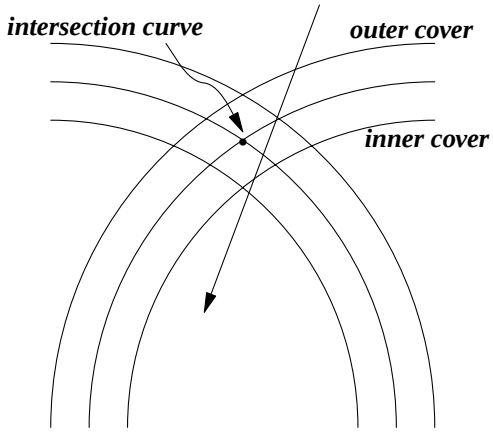


Figure 6. Intersection Between Overlapping Covered Objects

The second case is the antialiasing of the intersection curve when two objects overlap. Figure 6 shows how edge proximity can be detected when a ray passes thru the outer cover of both objects before passing thru the inner cover of either. Filtering is again a linear blend between the colors of the two objects, depending on the distance between the image point and the projected curve. The intersection curve is approximated by local tangent planes. The third case deals with shadow edges, and it is too complex to describe in detail here. It is somewhat problematic and involves the covers of the intersected object as well as the covers of the object casting the shadow.

Like cone tracing, ray tracing with covers requires only one sample per pixel to generate an antialiased digital image. Therefore, it is quite fast, but so far (as with cone tracing) it has been limited to models made up of very simple primitives. Unlike cone tracing, it does not seem feasible to simulate effects of diffuse reflection and area light sources easily.

2.4. Level of Detail

Another way to symbolically filter out unwanted high frequencies is to limit the level of detail at the modeling level. This might mean having a hierarchy of geometrical models for a given object, with simpler version of the object used when viewed at a distance [Clark76]. This technique has been used to increase the speed of real-time systems, but not (to my knowledge) in antialiasing ray tracers.

A more common use of variable level of detail is in *procedural models of texture.* Procedural texture is often defined as a series of increasingly higher frequency components [Perlin85]. For example, $1/f$ noise can be approximated as a series, using a function $B(\bar{X})$ which has a noisy spectrum limited to a narrow band of frequencies:

$$F(\bar{X}) = \sum_{i}^{\infty} \frac{B(2^i \bar{X})}{2^i} \tag{2}$$

This series can be truncated when the level of detail approaches subpixel frequencies. Even more exact prefiltering can be done if the procedural texture is an explicit Fourier series [Norton82]. Norton's technique, called *clamping*, took careful account of the perspective projection onto the image screen.

Specular highlights are often modeled by a $\cos^n(\alpha)$ function [Phong75], and these highlights can be a troublesome source of aliasing. Higher powers n result in a sharper highlight, but n can be lowered to affect antialiasing [Amanatides89].

3. Antialiasing by Resampling

Symbolic-prefiltering approaches to antialiased ray tracing have tended to be quite specialized and limited. In the current state of the art, the image function $I(x, y)$ of a ray tracer can be highly complex. Antialiasing must deal with many different mechanisms that contribute to unwanted high frequencies: object silhouettes, shadow edges, specular highlights, procedural texture, mapped digital texture, reflected and refracted details in an image, thin objects, etc. To be completely successful, symbolic methods must deal with all of these mechanisms, often requiring a special strategy for each one.

If the image function cannot be prefiltered, then the logical alternative is to increase the sampling rate. Recall from Figure 1, that the spectrum of a sample image consists of an array of replicas. When the samples are denser, the replicas are spaced farther apart. Even if the spectra continue to overlap, the amount of overlap will be reduced then. This is the case in ray tracing, where the image function has infinitely high frequencies, but the degree of aliasing can be reduced to arbitrarily low levels by higher sampling rates.

There are three important issues connected with boosting the sampling rate. First, since each sample requires a ray-casting calculation, increasing the sampling density will proportionally increase the computational expense. Secondly, there is the question of how to choose an appropriate sampling density. Finally, a digital *reconstruction filter* must be designed to convert the samples from the ray-casting rate down to the pixel rate. This approach to antialiasing is very general. In effect, it approximates the symbolic filtering problem with a digital filtering problem.

3.1. Adaptive Sampling

Clearly, there is a tradeoff between computational expense and sampling density. The simplest form of this antialiasing strategy would be to *supersample* the entire image at a high uniform rate. Choosing a sufficient supersampling rate is a matter of estimating or predicting properties of the image's frequency spectrum. By the sampling theorem, if W is the practical upper limit of frequencies in some dimension (i.e. the bandwidth), then a sampling rate of $2W$ is required to avoid aliasing.

Supersampling the entire image function seems wasteful since experience shows that aliasing usually occurs only in certain regions of an image (e.g, edges). This suggests that it would be much more efficient to make the sampling rate *adaptive.* Adaptive sampling is probably the best response to the issue of computational expense, but it increases the complexity of the issues of choosing a sampling rate and of reconstructing pixel values. This is because the sampling is no longer happening at a constant uniform rate.

The sampling rate is now based on some estimate of *local bandwidth.* Intuitively, if the image has a bandwidth of W in some dimension over a region of length X, then we expect that $2WX$ samples will be needed. This is an abuse of the sampling theorem, but it is a reasonable engineering rule of thumb which can be supported by careful theoretical analysis [Landau62].

A deep theoretical discussion of local bandwidth is not really necessary to understand adaptive sampling, but a brief digression into the interesting subject of localized frequency analysis won't do any harm. One common way to define the frequency spectrum of a signal in the neighborhood of a point is the *sliding-windowed Fourier transform.* For example in one dimension, a signal $g(x)$ is first multipled by a window function $w(x)$ which is zero everywhere except a small region centered at zero. The window might be tapered so that $g(x)w(x)$ goes to zero gradually at the edges. Then the strength of the signal around x_0 at frequency f_0 is found by sliding the window to that position and doing a Fourier transform:

$$G(x_0, f_0) = \int_{-\infty}^{\infty} g(x)w(x - x_0)e^{-2\pi i f_0 x}\,dx \tag{3}$$

This type of localized frequency analysis is implied when we discuss the bandwidth of an image in the

neighborhood of a pixel. Typically the window is simply a square region one pixel wide.

Another view of localized frequency analysis is given by *filter banks.* The signal is separated into a set of signals, each containing an exclusive range or "band" of frequencies. Then we can refer to the strength of the signal at location x_0 in a band $[f_i, f_{i+1}]$. This view is useful when discussion antialiased texture maps, where the texture images may be represented by a set of digital pictures at different resolutions.

It is not likely that such sophisticated views of local bandwidth will ever be needed to perform adaptive sampling in ray tracing, but it is important to understand that saying "local" and "bandwidth" in the same context implies something beyond standard Fourier analysis. The use of advanced estimation methods for local bandwidth is still a problem requiring more study.

The classic algorithm for adaptive sampling is described in the foundational paper on ray tracing by Whitted [Whitted80]. It provides a practical example of how to estimate local bandwidth and how to reconstruct the pixel values from the adaptively placed ray-casting samples. This algorithm estimates the integral of the image function over a square pixel-size region. Note that "area averaging" is mathematically equivalent to prefiltering with a box filter and then sampling.

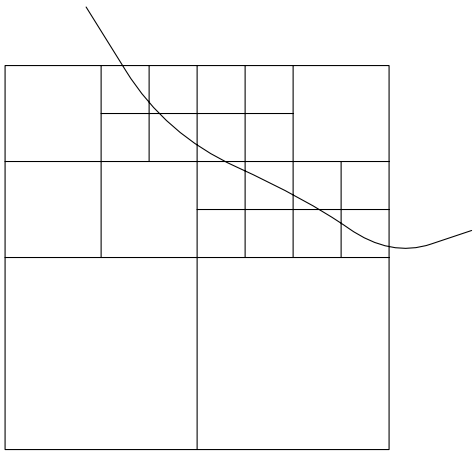


Figure 7. Subdivided Pixel Area in Whitted's Algorithm

Whitted's algorithm estimates the area integral by adaptive subdivision. Figure 7 illustrates a possible scenario, where an irregular object impinges on the upper right corner of the pixel area. We begin with ray-casting samples made at the pixel rate and located at the corners of the square pixel area. If there is no *significant* differences between the four corner samples, then the pixel value will be the average of the four corners. If there is a significant difference, then the square is subdivided into four subsquares, and the algorithm proceeds recursively to some limiting subdivision level. The contribution that each subsquare makes to the pixel value is the average of its four corner samples weighted by its proportional area.

Testing the differences between samples is a discrete estimate of the magnitude of the local gradient of the image function. This gives a rough measure of local bandwidth because the first derivative value of a signal $g(x)$ has the frequency-weighted spectrum $-i\omega G(\omega)$. Energy at high frequencies therefore makes a larger contribution to the gradient magnitude.

One difficulty with this approach is that very small objects can slip between the initial pixel-rate samples. In an animation, a small object like that might pop in and out of the scene, an effect called *scintillation.* Whitted proposed that sufficiently large invisible covers surround small objects so that the adaptive sampling procedure can be alerted to their presense (i.e., when a ray strikes a cover).

What is meant by "significant difference" when comparing adjacent samples? In practice, a threshold is chosen by trial and error. More sophisticated measurement of significance could be based on *visual psychophysics*, how the eye is expected to respond to the difference. Two properties of the visual system are particularly relevant. First, the eye's response to local differences in luminance (ΔL) is nonlinear and seems to be approximately proportional to $\Delta L/L$ rather than just the difference ΔL. In other words, a given difference in samples will be more noticeable in a dark region of an image than if it were in a bright region. This suggests one should test for a significant ratio rather than a significant difference.

A second interesting property of the eye is that sensitivity to high-frequency spatial noise (e.g., aliasing) is color dependent. Because green-sensitive receptors are much more densely populated in the retina, spatial variations in green intensity are more significant. The poorest sensitivity is to blue. Notice that we are talking about sensitivity to contrast and high-frequency spatial noise, not about perception of errors in hue. Thus, it would be incorrect to measure significance of difference as a "color distance", using CIE colorimetric coordinate systems.

The reconstruction scheme in Whitted's algorithm is interesting and intuitively pleasing. However, there are some minor problems. It is important to keep in mind that the end result of this reconstruction is the production of a pixel value as a weighted average of the ray-casting samples. Figure 8 shows the actual weights (in parts per 256) of the samples from the example in Figure 7:

4	5	2	2	2	5		4
	2	4	4	4	2		
8	13	2	7	4	7	2	5
			2	4	4	4	2
20	8		37	2	2	2	17
16			32				16

Figure 8. Sample Weights from Whitted's Algorithm

There are some peculiarities in these sample weights. Some samples contribute little, while nearby neighbors are weighted heavily. If area average is the goal (and area average is not the best prefilter), then a more satisfying scheme might be to weight each sample by the area of its surrounding Voronoi cell. The Voronoi cell of a sample is the area which is closer to that sample than to any other:

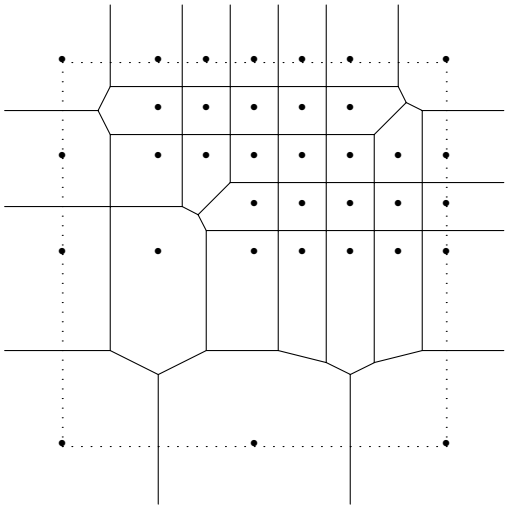


Figure 9 Voronoi Cells of Adaptive-Sample Locations

Unfortunately, computing the Voronoi diagram is quite expensive. This would not be a practical solution, but as shown in Figure 9, it gives some qualitative indication of how we would like to weight the ray-casting samples. An approximation of Figure 9 with rectangular cells is probably quite easy to compute.

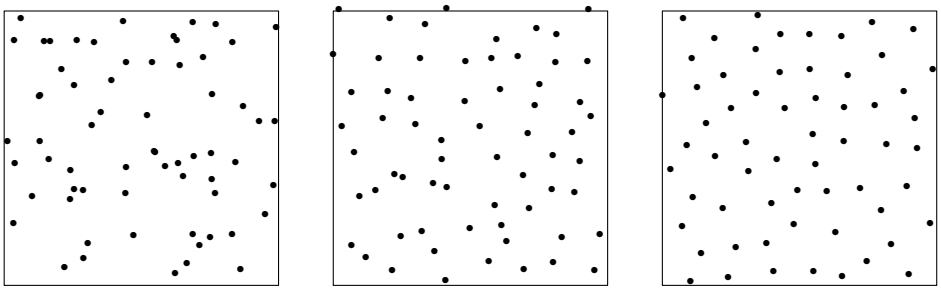
4. Nonuniform Sampling

The familiar aliasing artifacts of "jaggies" and Moiré patterns are particularly conspicuous to the human eye, because they are often extensive and orderly in structure. This results from the interaction of the image with the regular uniform spacing of the samples. For example, a Moiré pattern is the result of a periodic structure in the image *beating* with the periodic sampling pattern. Could this effect be eliminated by breaking up the regularity of the sampling pattern? At Bell Labs, we experimented briefly with this idea, randomizing the sampling positions. The resulting images were extremely noisy, and the approach was abandoned.

Our experiments with randomized sampling failed because we were missing the crucial theory of how nonuniform sampling designs affect subjective image quality. The standard view of aliasing breaks down when sampling is not uniform, and there is no simple notion of what the best nonuniform sampling pattern should be. An interesting solution to this problem was revealed by studying the pattern of photoreceptors in the retina [Yellott83], and this led to the successful application of nonuniform sampling to ray tracing by researchers at Lucasfilm and elsewhere [Cook84, Dippé85, Cook86, Mitchell87].

4.1. Stochastic Point Processes

Understanding Yellott's discovery and its impact on the antialiasing problem requires some knowledge about properties of nonuniform patterns and a few more facts about visual psychophysics. The nonuniform sampling patterns of greatest interest are examples of *stochastic point processes.* Figure 10 shows samples of three important point processes:



Poisson Uniform Jitter Poisson-disk

Figure 10. Examples of Stochastic Point Processes

The *Poisson* process is one of the simplest, where the probability of finding a point is uniformly random and independent of the location of any other point. Another important process is *uniform jitter,* where the points of a uniform lattice are perturbed by a uniformly distributed random amount. Notice that the points in a jitter process are not quite as "clumpy" as the points in a Poisson process; the variance of sample density is less.

The third process shown above is the *Poisson-disk* process, in which a minimum distance constraint is added to the Poisson process. The points are centers of randomly-placed nonoverlapping disks. There are actually several different point processes which are called Poisson-disk [Ripley77], but the one illustrated above is a *dart-throwing* process. A candidate point is generated at random and added to the pattern only if it is not too close to any other points already in the pattern. This proceeds until there is no place left to fit a new point.

Stochastic point processes can be characterized by statistics such as the mean density of points per unit area λ. An important second-order statistic is the *autocorrelation function.* Given a point at (x, y), the autocorrelation indicates the average likelihood of finding another point at $(x+u, y+v)$:

$$A(u,v) = E\left\langle \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} s(x,y)\, s(x+u, y+v)\, dx dy \right\rangle \tag{4}$$

Here $s(x, y)$ is the sampling function, a sum of δ-functions located at each sample point. The expectation value is an average over the ensemble of all possible instances of the stochastic point process. It is assumed that the point process is well-behaved in the sense that λ and $A(u, v)$ are not dependent on position (x, y). The process is then said to be *wide-sense stationary.*

Figure 1 illustrated the effect of uniform sampling on the spectrum of an image—periodic replicas of the original unsampled spectrum. It is particularly easy to view aliasing as the foldover of these replicas. By a similar analysis, we would like to know the spectral consequences of nonuniform sampling, and the first step is to look at the spectrum of the sampling function $s(x, y)$.

Some bending of the rules was required to state that the Fourier transform of a uniform lattice of δ-functions is also a uniform lattice of δ-functions. In the nonuniform case, this is not possible, and the Fourier transform of $s(x, y)$ is infinite or undefined. However, for stationary point processes, it is possible to define the *spectral density,* which is the energy per frequency *per unit area.* This happens to be the Fourier transform of the autocorrelation function.

The spectral density can be estimated by forming a collection of point patterns (random instances of the point process), windowing them to avoid edge effects, and averaging their Fourier transforms. The

following two figures show examples of these spectra, the radial average of the power spectra of the Poisson process and a Poisson-disk process:

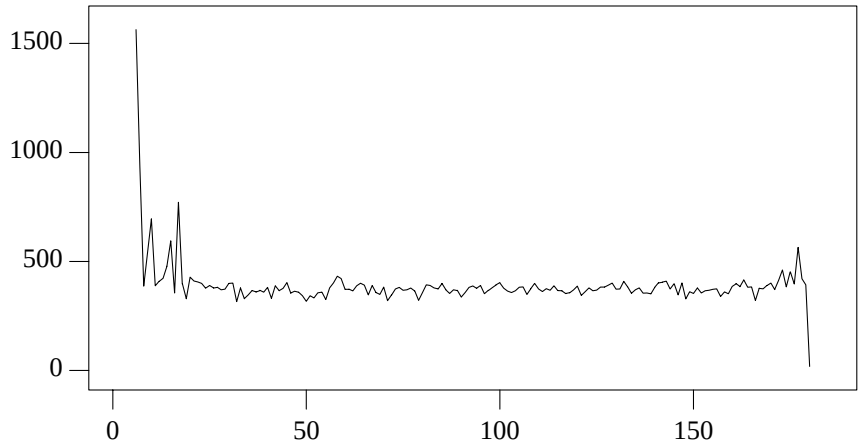


Figure 11. Radial Spectral Density of Poisson Process

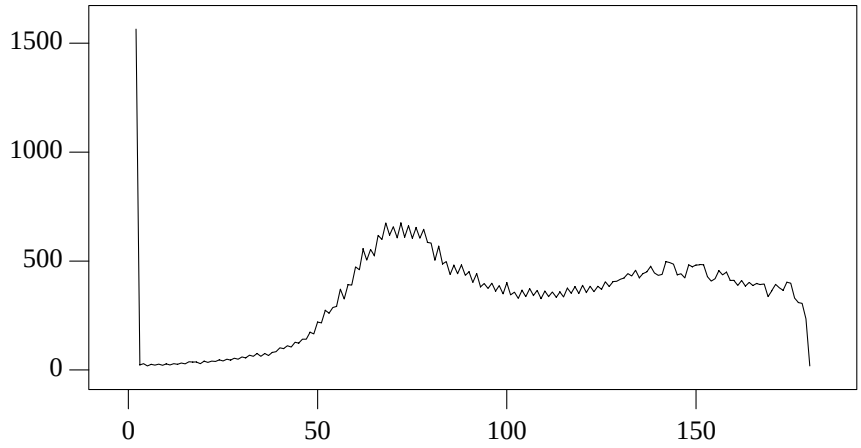


Figure 12. Radial Spectral Density of Poisson-disk Process

Unlike the regular lattice structure of the uniform-sampling spectrum, stochastic sampling structures like these consist of a single δ-function at the origin and some pattern of noise elsewhere. When sampled, the spectrum of an image is convolved with this, resulting in a copy of the image spectrum (from convolution with the δ-function) and some noise (from convolution with the noisy part of the sampling spectrum). This is illustrated in Figure 14. While uniform sampling can result in spectral overlap and periodic structured aliasing, nonuniform sampling can result in noisy unstructured aliasing.

In the case of the Poisson process, the noisy part of the spectrum is broadband white noise. However, the noisy part of the Poisson-disk spectrum is concentrated at high frequencies. We could call this type of spectrum *blue noise.*

Yellott noticed that the photoreceptors in the retina of rhesus monkeys are arranged in a Poisson-disk

pattern [Yellott83]. The spectral consequences of this type of sampling are that the noisy aliasing will be higher in frequency than if simple Poisson sampling was used. This is good because the eye is less sensitive to the high-frequency noise, and general lack of order in the aliasing (as opposed to a Moiré pattern) makes it less conspicuous.

The jitter process can also be used for nonuniform sampling. Its spectrum contains more low-frequency noise than the Poisson-disk, but it is much easier to generate this type of sampling pattern. Important existing systems use jitter sampling [Cook84, Dippé85, Cook86]. The Poisson-disk pattern can be precomputed and stored by simulating the dart-throwing process on a small patch (with periodic boundaries), and then replicating to cover the image plane. A good approximation to the Poisson-disk pattern can be generated on the fly by a point-diffusion algorithm similar to the Floyd-Steinberg dithering technique [Mitchell87].

4.2. Adaptive Nonuniform Sampling

Nonuniform sampling does not eliminate aliasing. Rather, it changes the nature of the aliasing noise to make it less conspicuous. Edges and other high-frequency components in the image function may still require relatively higher sampling density to get a true reduction of aliasing noise. This leads to a desire to make nonuniform sampling adaptive. As in the uniform-sampling case, there are the problems of choosing a local sampling rate and reconstructing pixel values, but nonuniformity exacerbates these problems. In addition, nonuniformity makes it slightly more difficult to generate variable-rate sampling patterns.

Straightforward adaptive-sampling schemes like Whitted's do not have a simple analog in the case of nonuniform sampling. Nonuniform subdivision would require more complex data structures, and it would be difficult to maintain the optimal blue-noise spectrum at the same time. An impractical subdivision algorithm would be to approximately double the sample density by forming the Voronoi diagram of a region of points and then adding a new sample at each vertex.

Dippé and Wold suggest creating a variable-density sampling pattern by a Poisson-disk process with a variable minimum-distance constraint [Dippé85]. The dart-throwing algorithm could then pack points more closely together in regions where the image has higher frequencies. The problem with this approach is that the sampling pattern cannot be precomputed and stored, and dart-throwing is too expensive to do on the fly. The point-diffusion method could be easily modified to produce variable density, altho as the density becomes higher, the process no longer yields a good blue-noise spectrum.

A simpler approach is to quantize the sampling density into a discrete number of levels—possibly just two levels. This is the approach taken by Cook and Mitchell [Cook86, Mitchell87]. In Cook's application, 16 samples per pixel were taken (using a jitter process), and where necessary, some pixels were sampled 64 times. These are rather high sampling rates, necessary for the "distributed ray tracing" algorithm.

The "antialiasing module" used by Mitchell was based on a stored periodic Poisson-disk pattern of 1024 points which was scaled to provide two densities of sampling. The entire image was sampled at about the pixel rate (an average of 1.2 samples per pixel), and where needed, additional samples were made at the rate of 9 samples per pixel. At these fairly low sampling rates, the Poisson-disk pattern was noticeably superior to jitter sampling.

Dippé and Wold have suggested an adaptive jitter-sampling process. Sample density can be progressively increased, as illustrated in Figure 13, by selecting a rectangular cell containing one point, bisecting in one dimension, and generating a new random point in the empty half. Which dimension is bisected could be chosen randomly, it could alternate with the level of subdivision like a K-D tree, or it could be based on local estimation of gradient direction [Kajiya86].

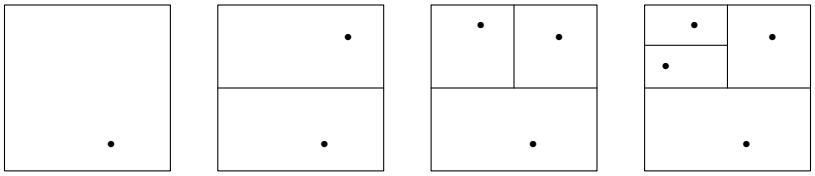


Figure 13. Adaptive Jitter Pattern

Sampling density is adapted to local bandwidth in the image. As in Whitted's adaptive-sampling scheme, local bandwidth is usually estimated by looking at the differences in the values of adjacent samples, but other cues may also be used (e.g., covers around small objects). For example, in Mitchell's system, the variation of sample values in a region determines whether additional high-density sampling is performed. More specifically, conditional high-density sampling is done in square blocks two pixels wide, and the decision to supersample is based on inspecting the low-density samples in a surrounding square four samples wide. Some visual psychophysics is considered in assigning significance to the sample variation [Mitchell87].

It should be noted that predicting the presense of local high frequency based on some initial low-density samples is not always possible in the most rigorous sense. The observation of a large difference between adjacent samples could be caused by a variation of the image that is below the Nyquist frequency. When we choose to supersample that region, we are implicitly accepting the alternative hypothesis that the sample values are actually an alias of a much higher frequency. This hypothesis is justified by the structure of typical images. A sharp edge is much more likely than a half-cycle-per-pixel texture. Furthermore, it is not a serious error to supersample in a few places where it was not really needed.

Once local supersampling has taken place, the alias hypothesis can be verified *a posteriori*. The high-density supersamples give better information about higher frequencies in the image. One could take advantage of *coherence* in the local bandwidth by continuing to supersample until reaching a region where the high-density samples lead to a confident rejection of the alias hypothesis. One might call this "hysteresis adaptive sampling" because it makes supersampling easier to turn on than to turn off. Some recent work on edge-following adaptive sampling could be considered an example of this concept [Thomas88].

4.3. Nonuniform Reconstruction

Deciding where to make ray-casting samples and how densely to sample is only half the problem. These samples must then be used to calculate pixel values, and their nonuniformity makes the problem especially difficult. This calculation can be viewed as consisting of two steps: reconstructing a continuous image from the samples, and then resampling this continuous image to get the pixel values. In practice, the continuous representation will probably not be explicitly constructed, or perhaps an approximation to it may be constructed.

The adaptive jitter sampling, shown in Figure 13, has an obvious reconstruction algorithm associated with it—to weight each sample in proportion to the area of the cell it is contained in. This is an approximation of the Voronoi-area-weighting scheme illustrated in Figure 9. The situation is somewhat simpler than with Whitted's subdivision algorithm, because the samples are distributed about the center of the cells instead of being located at the corners. Of course, area averaging (i.e., box filtering) is implied by this reconstruction scheme.

In the general case, we may be faced with an arbitrary arrangement of samples and with no algorithm for computing the pixel values that is obvious *a priori*. In the case of uniform sampling, reconstruction is accomplished by low-pass filtering, as indicated by the dotted box in Figure 1. However, the consequences of nonuniform sampling create a more complex situation as show in Figure 14:

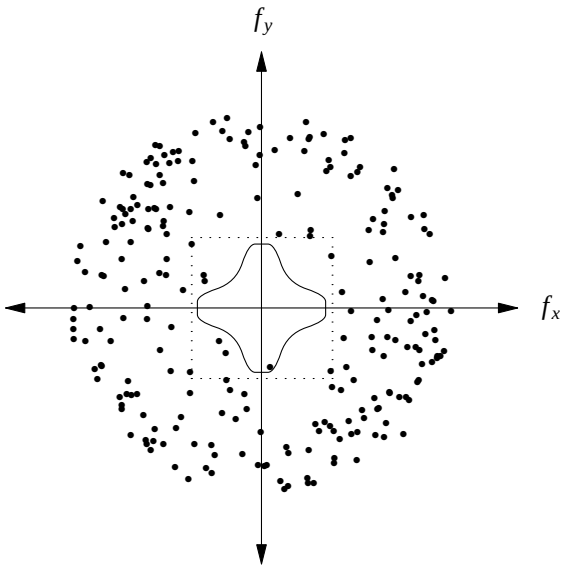


Figure 14. Spectrum of Nonuniformly Sampled Image

The scattered dots in this figure are meant to represent noise, which now replaces the neatly isolated spectral replicas of Figure 1. The low-pass filter, represented by the dotted box, passes some of this noise and can cause the reconstructed image to look grainy. This is an example of postaliasing, which is now a more perverse problem than it was with uniform sampling. When sampling is adaptive, the nonuniformity can be extreme and the sampling pattern can no longer be thought of as stationary. Low-pass filtering is completely unacceptable as a reconstruction method in that case.

For nonuniform sampling, a common alternative to low-pass filtering is *normalized weighted-average interpolation* [Dippé85, Cook86]. In one dimension, suppose we have a collection of samples $g(x_n)$ and a filter kernel $k(x)$. Then the value of g at any point x can be interpolated by:

$$g(x) = \frac{\sum_{n=-\infty}^{\infty} k(x - x_n)\, g(x_n)}{\sum_{n=-\infty}^{\infty} k(x - x_n)} \tag{5}$$

One of the results of this is perfect reconstruction of a constant signal $g(x) = C$. In that case, the expression above will be equal to C everywhere, and it seems that this method produces good results when g is fairly low in frequency and when the sample positions are not too unevenly distributed. However, adaptive sampling can violate those conditions and cause this method to perform poorly.

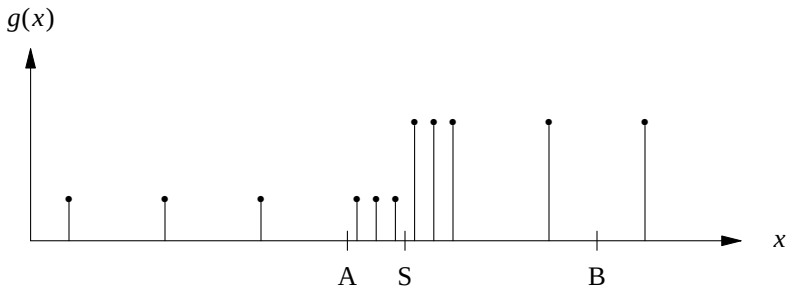


Figure 15. Problem Situation for Normalized Weighted Average

Figure 15 illustrates an example (in one dimension) of this problem. The signal being sampled has a sudden step at S, and its neighborhood has been adaptively sampled at higher density. Suppose we wish to reconstruct this function with a box filter, by estimating the average of the signal in the interval $[A, B]$. We would want a fairly small contribution to this interval from the low part of the signal to the left of the step (about 25%). But 3 of the 7 samples in this interval are to the left of the step. Normalized weighted average will give the low part of the function to the left of the step about 43% weight. Intuitively, we want the right-most sample in this interval to have more weight because it "represents" most of the right half of $[A, B]$.

Of course, it is easy to interpolate the samples in Figure 15 by connecting samples with piecewise linear sections. In general, any filter can be made to exactly interpolate these samples by first warping the kernel to match the uneven placement of the samples [Clark85]. Unfortunately, this does not solve the problem at hand, because the resulting interpolation will not be properly bandlimited. Aliasing can result when the reconstructed signal is resampled at the pixel rate. For example, linear interpolation between the samples in Figure 15 would result in a function with almost as sharp a step as the original signal.

What we really want to derive from the adaptive samples is an accurate estimate of the *prefiltered* signal, which we cannot compute directly in the ray tracer. This sort of reconstructed signal should look more like the curve in Figure 16:

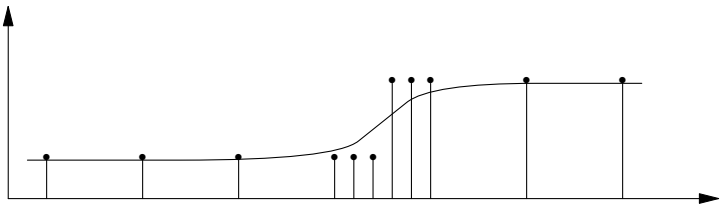


Figure 16. Bandlimited Reconstruction from Adaptive Samples

One possibility is to perform this bandlimited reconstruction in two stages, by first constructing a simple non-bandlimited interpolation (e.g., piecewise linear or piecewise constant), and then convolving that reconstruction with a proper filter [Mitchell87].

Painter and Sloan applied this technique to samples made by adaptive jitter [Painter89]. As the example in Figure 13 shows, adaptive jitter samples are each contained in a rectangular region, and in a piecewise-constant reconstruction, each of these regions would have a constant value (i.e., the sample's value). For each subpixel rectangular region R_i with constant value C_i, we can convolve with a filter kernel $k(x, y)$ by integration:

$$h_i(x, y) = C_i \int_{R_i} \int k(x - x', y - y')dx'dy' \tag{6}$$

The pixel value at (x, y) would be the sum of $h_i(x, y)$ values for all regions within the support (i.e., width) of $k(x, y)$. As Painter and Sloan point out, this integral may be done in closed form if the kernel is a polynomial, and for more general filters the integral values may be precomputed in a sum table [Crow84]. Naiman gives details of how to convolve rectangles with filters by table lookup [Naiman87].

Another possible approach to nonuniform reconstruction is to associate nonuniform samples with positions on a uniform subpixel lattice. Then samples generated by arbitrary methods can be handled somewhat like those of adaptive jitter. The multi-stage filter used in Mitchell's system begins with a subpixel lattice and then forms the final reconstruction by filtering and normalizing the image multiple times with filters of ever widening support [Mitchell87].

5. The Statistical Sampling Approach

Cook's remarkable paper on "distributed" ray tracing [Cook84] did not reveal the methods and theory of nonuniform sampling. Before this information was published [Cook86], a number of other sites began research in this area. Of these efforts, work done at the University of Tulsa was particularly interesting, because they assumed (incorrectly, but fortuitously) that the Lucasfilm results were based on Monte Carlo methods [Lee85].

Up to this point, sampling artifacts in graphics have been discussed solely in terms of the theory of signal processing and aliasing. An alternative viewpoint is provided by the theory of statistical sampling and estimation, and some work on antialiased ray tracing seems to emphasize this formalism [Lee85, Kajiya86, Painter89]. For pedagogical reasons, I am separating the discussion of that work from nonuniform sampling [Dippé85, Cook86, Mitchell87], but of course, they are interrelated. All of this work is often referred to as "nonuniform sampling" or "stochastic sampling".

5.1. Sequential and Block Designs

Only a short digression will be required to review the relevant statistical theory. A number of texts on sampling techniques and experimental design contain more details and many techniques which have not yet been applied to this particular problem [Davies56, Cochran77]. This methodology assumes that the parameter being estimated has a normal distribution, which is not really true. These techniques are often assumed to be robust when the parameter under study is at least unimodal, but even that is a doubtful assumption for typical image functions. Nevertheless, the techniques have been applied and seem successful. Keep in mind that the lack of a known bandlimit on the image function also causes the techniques based on signal-processing theory to be uncertain.

Computing a pixel value by area average is, as the name implies, an attempt to compute the mean value μ of the image function in the pixel area. To simplify the discussion of this problem, consider a one-dimensional signal $g(x)$ being studied (in some interval). Given a set of n samples, the estimate of the mean is simply:

$$\bar{g} = \frac{1}{n} \sum_{i=1}^{n} g(x_i) \tag{7}$$

The quality of this estimate is indicated by its variance:

$$\sigma_{\bar{g}}^2 = \frac{\sigma^2}{n} \tag{8}$$

Where σ^2 is the variance of $g(x)$ in the interval under study. A *confidence interval* can be computed giving a probability $1 - \alpha$ that the estimate $\bar{g}$ lies between tail-end values $\pm z_{\alpha/2}$ of the standard normal distribution:

$$\bar{g} \pm z_{\alpha/2} \frac{\sigma}{\sqrt{n}} \tag{9}$$

Unfortunately, we do not know the actual value of σ^2, but it can also be estimated from the samples by:

$$s^2 = \frac{\sum_{i=1}^{n} (g(x_i) - \bar{g})^2}{n - 1} \tag{10}$$

Using s instead of σ, the confidence interval is:

$$\bar{g} \pm t_{\alpha/2} \frac{s}{\sqrt{n}} \tag{11}$$

Where $t_{\alpha/2}$ is a tail-end value of the Student's t distribution with $n - 1$ degrees of freedom. One possible strategy would be to randomly sample the pixel area and compute the confidence interval after each new sample until it reaches an acceptable level. This is called a *sequential experiment.* An acceptable confidence interval might be determined by the quantization of the display (e.g., $\pm 1/256$ for typical 8-bit channels), or it could be based on considerations of visual psychophysics.

A second important statistical method is *block sampling* (also called stratified sampling). Instead of taking samples randomly within the pixel area, it could be subdivided into smaller blocks. This is advantageous if the image has a large variance within the pixel area, but is relatively homogeneous within each subpixel block. A precise pixel value can be computed by averaging the estimates of the mean in each low-variance block.

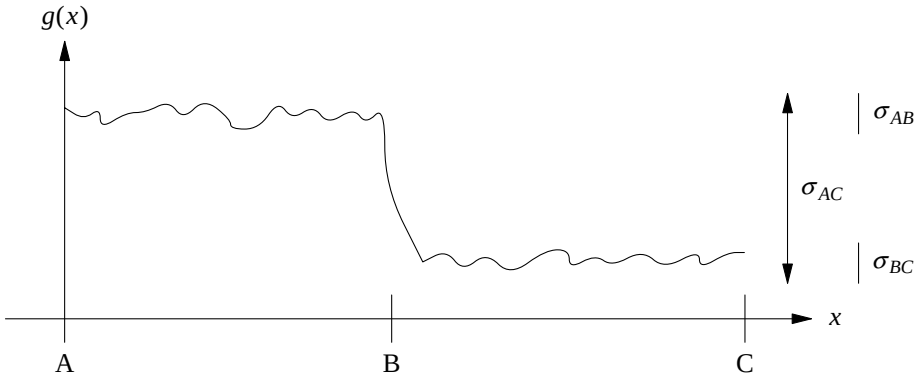


Figure 17. An Opportune Scenario for Block Sampling

Figure 17 shows a good example of where block sampling would be beneficial. The signal $g(x)$ has two distinct homogeneous regions of about equal size. The variance of the estimate of the mean will be much smaller if estimates from the two blocks are combined. In general, if the region being sampled is divided into L blocks with fractional areas W_i, and n_i samples are made in each block, then the intra-block estimates of the means and the variances of the means can be combined as:

$$\bar{g} = \sum_{i=1}^{L} W_i \bar{g}_i \tag{12a}$$

$$\sigma_{\bar{g}}^2 = \sum_{i=1}^{L} W_i^2 \left(\frac{s_i^2}{n_i} \right) \tag{12b}$$

Lee *et al* used both the concepts of sequential sampling and block sampling in their system [Lee85]. The pixel area was divided into 8 blocks, and the sampling was carried out until a satisfactory variance of the mean was achieved (between 8 and 96 samples per pixel). Kajiya discusses other possible sampling strategies, including the adaptive jitter method of Dippé and some techniques used in Monte Carlo integration [Kajiya86].

5.2. Multiparameter Sampling and Distributed Ray Tracing

Nonuniform/stochastic sampling is a powerful technique for suppressing visible spatial aliasing artifacts, but it is probably the benefits of "distributed ray tracing" which have provided the greatest motivation for dealing with the complications of these techniques. Distributed ray tracing is a method of rendering images with motion blur, depth of field effects, and illumination by area light sources [Cook84].

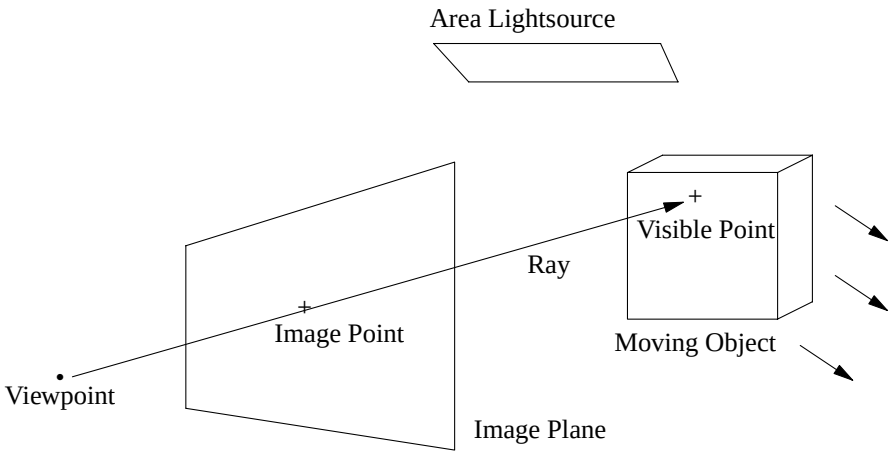


Figure 18. Distributed Ray Tracing

Figure 18 illustrates a simple example with an area light source and a moving object. The image function $I(x, y, t, u, v)$ now has more parameters to sample than a simple ray tracer. A single evaluation of this function gives a color value for some point (x, y) on the image plane, at some point in time t, illuminated by some point (u, v) on the area light source. This in itself does not require any techniques not found in the image function of a standard ray tracer. However, a fully antialiased pixel value represents an average over all these parameters; the area of a pixel, the interval of the time exposure, and the surface of the light source.

It is not practical to apply randomized block sampling in the obvious way. Dividing the five-dimensional parameter space into four levels in each dimension would yield $4^5 = 1024$ blocks—too many to compute a sample in each one. In the field of survey sampling and experimental design, there are numerous compromises which use sparser sampling [Davies56].

One such strategy is the *Latin square* design. Suppose that we wish to simulate motion blur by sampling a three-parameter image function $I(x, y, t)$. Let the temporal dimension be divided into four adjacent intervals labeled A, B, C, and D. The pixel area can be divided into 4 x 4 cells, and a temporal block associated

with each cell:

$$\begin{matrix} A & B & C & D \\ B & A & D & C \\ C & D & A & B \\ D & C & B & A \end{matrix} \tag{13}$$

Using this arrangement, 16 randomized samples are made, one in each subpixel cell and its associated temporal block. The advantage of this design is that it may avoid possible effects of correlation between subpixel location and object motion. Each row and each column contains a sample from each temporal block. Cook has described using a "magic square" design to accomplish the same effect [Cook85].

With additional parameters to sample, it is possible to construct additional Latin squares which are *orthogonal*. For example, labeling two additional parameter blocks with greek letters and numerals, we have:

$$\begin{matrix} A\alpha 1 & B\beta 2 & C\gamma 3 & D\delta 4 \\ B\gamma 4 & A\delta 3 & D\alpha 2 & C\beta 1 \\ C\delta 2 & D\gamma 1 & A\beta 4 & B\alpha 3 \\ D\beta 3 & C\alpha 4 & B\delta 1 & A\gamma 2 \end{matrix} \tag{14}$$

In this arrangement, each combination of pairs of blocks occurs just once. Now samples are being made in 16 of the possible 1024 blocks, and each pair of parameter levels (row, column, Latin letter, Greek letter, and numeral) occurs an equal number of times. The effect of these types of statistical sampling designs on ray-traced image quality has not been fully studied.

These techniques are concerned with intra-pixel sampling strategies, but one can also consider inter-pixel design. Suppose that in each 4 x 4 block of pixels, we sample time in 16 possible blocks as follows:

$$\begin{matrix} 0 & 12 & 3 & 15 \\ 8 & 4 & 11 & 7 \\ 2 & 14 & 1 & 13 \\ 10 & 6 & 9 & 5 \end{matrix} \tag{15}$$

This matrix happens to be the same as the ordered-dither matrix used in halftoning [Limb69]. In one simple experiment, the author ray traced a scene containing moving objects, using only one sample per pixel and the above inter-pixel temporal blocking. The image was, at first glance, remarkably good. The motion-blur, on closer inspection, had a finely stippled appearance. The temporal-aliasing noise had been forced into the form of a high-frequency pattern, just as ordered dither does to quantization noise. This phenomena probably merits further study.

6. Temporal Antialiasing

The effects of temporal sampling deserve special attention. At the present time, there are two major vehicles for displaying moving images; video and cine (i.e., motion pictures). Both of these mediums involve sampling in the time dimension, at 24 frames per second in cine or 30 in video (video is 25 frames per second in some countries). Like any instance of sampling, aliasing is a potential problem.

Anyone who has played with a strobe light is familiar with the odd effects of temporal aliasing. There is a peculiar jerky quality to motion, and a spinning wheel may appear to turn backwards. These effects are

called *strobing* or *motion judder.* Like a strobe light, the opening and closing of a movie-camera shutter can produce the same effects, and so can a computer-generated animation which is rendered as a sequence of still frames.

This aliasing can be removed by temporal prefiltering, which is often called *motion blur.* In a movie camera, this prefiltering is accomplished by simply exposing each frame for some interval of time (typically 1/48th of a second or less). In a ray tracer, this same procedure is simulated by stochastically sampling over an interval of time, as discussed in section 5.2.

However, motion blur is not a panacea. Motion blur looks realistic when viewers are not tracking a moving object with their eyes. But during visual pursuit, the image of the object is more or less fixed on the retina, and then motion blur results in an unnatural smeared appearance (and lack of motion blur may result in severe judder). The only way to eliminate this problem would be to greatly increase the frame rate of motion pictures and video—the effects of temporal aliasing are visible even at 120 frames per second [Hsu85]. Given the current standardized frame rates, judgements about the degree of motion blur versus judder are generally made by the cinematographer or videographer; it is more of an artistic issue than an engineering problem.

Aggressive low-pass filtering is almost never called for, since this would produce extreme motion blur. This is why sophisticated filter designs are not worth considering; just integrating over an interval of time is sufficient. Motion blur should be applied minimally, and only if strobing is judged to be a problem; otherwise the sharpest images are made by doing no temporal filtering at all. Generally, more motion blur is needed in cine than in video because of the lower temporal sampling rate.

In a single frame of an animation, a motion-blurred object seems to be spatially filtered along the direction of motion. Motion blur can be simulated by this principle; filtering an object spatially instead of actually sampling a time interval [Potmesil83]. Unfortunately, this algorithm does not always produce correct results (e.g., a moving mirror).

A ray tracer is not constrained to imitate the action of a physical camera. It is possible to apply different degrees of motion blur to different objects in the same scene. Perhaps this would be a useful feature to a cinematographer, a possible topic for further study.

Generating antialiased video presents some special problems not encountered with cine. A video signal represents the scanning of a moving 2D image at the rate of 30 frames per second (25 in some countries). This scanning is *interlaced,* which means the odd-numbered lines are scanned in the first 1/60th second, and then the even lines are scanned. Figure 19 illustrates this scanning process in a slice perpendicular to the horizontal scan lines. This view in (t, y) coordinates illustrates the discrete sampling nature of the scanning process.

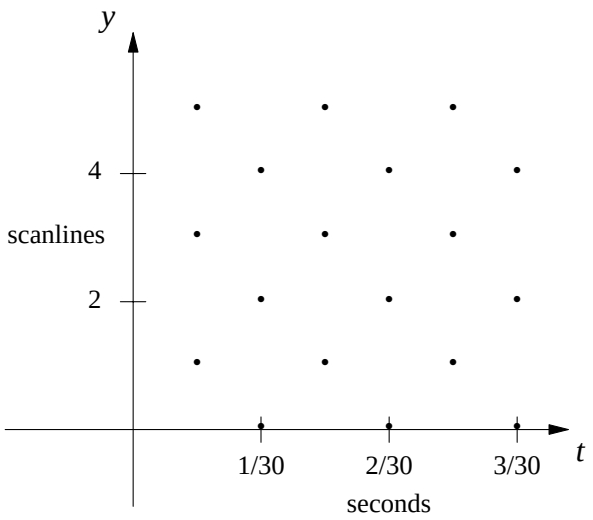


Figure 19. (t, y) Slice Through Interlaced Scans

When a signal is sampled, the resulting interlaced video spectrum consists of replicas of the spectrum of the original signal. Figure 20 shows the spectral consequences of interlaced sampling.

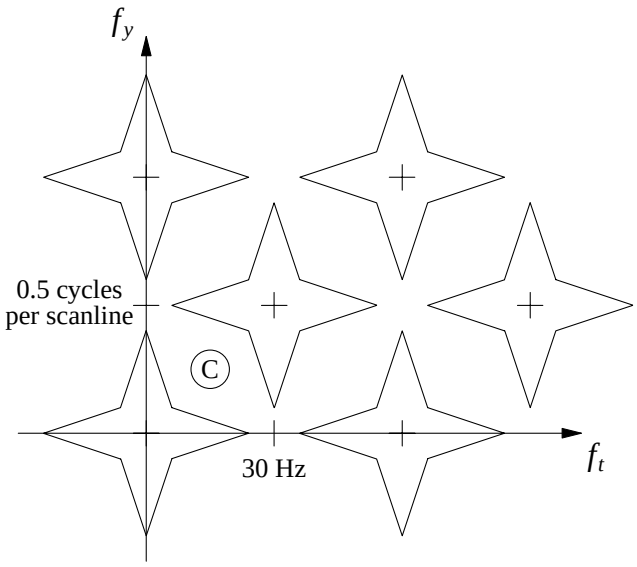


Figure 20. (t, y) Spectrum of Interlaced Digital Video

An annoying aliasing artifact of interlaced video is interline flicker. This results from the sideband centered at (30 Hz, 0.5 CPS) in the figure above. In particular, any signal near (0 Hz, 0.5 CPS) will be duplicated by this sideband at (30 Hz, 0 CPS). An obvious example would be a pattern which is brighter at every other scan line, which will cause the whole display to flicker at 30 Hz. A more typical manifestation of flicker is crawling jaggies ("line crawl") or fluctuating Moiré patterns.

The flicker problem is largely caused by the display (i.e., it is postaliasing), and could be eliminated by properly designed "de-interlacing" monitors. However, with standard interlaced monitors, the best that can be done is to carefully limit vertical frequencies near 0.5 cycles per scanline. Television cameras must also do this, typically integrating over a path two lines high as they scan.

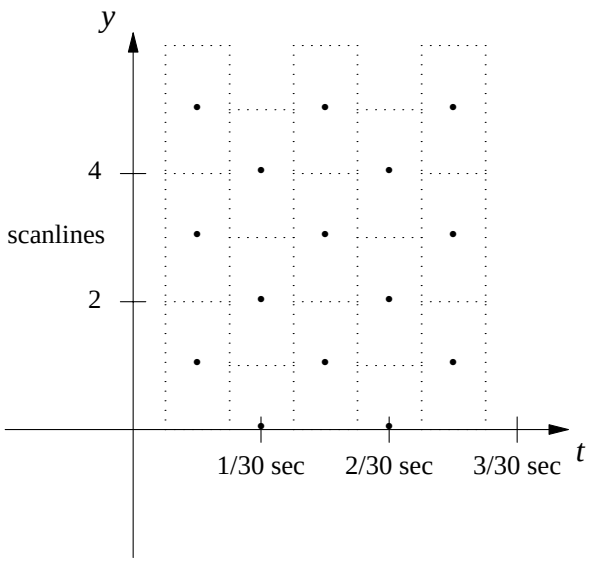


Figure 21. Interlaced Spatiotemporal Box Filter

Figure 21 shows a possible filter for interlaced video. Each pixel is an average over two lines high, one pixel wide (perpendicular to this view) and 1/60th of a second in duration. That is probably far to much motion blur, and in practice the time duration should be adjustable. Regardless of the amount of temporal filtering (if any), it is vital that the scanlines be interlaced in time.

In broadcast video (NTSC) there can also be problems if detail in the scene causes aliasing with the color subcarrier. This and other problems of generating antialiased video have been studied in some detail [Chuang85, Amanatides90].

7. Digital Filter Design

Antialiasing by resampling usually uses a digital filter to derive pixel values from the ray-casting samples. It is often expedient to use a very simple filter—averaging over a square pixel area is equivalent to convolving the image function with a square box filter and then sampling at the pixel locations. A box filter is not particularly good, as we will see, but it seems to be adequate for suppressing the most gross jagged-edge artifacts. In nonuniform-reconstruction schemes like Painter and Sloan's, almost any filter can be used, and so the issue of filter design becomes of practical importance.

One of the main objectives of prefiltering (whether it is symbolic filtering or digital filtering of supersamples) is to prevent aliasing during pixel-rate sampling. As Figure 1 shows, the spectrum of the image function must be confined to a bounded region (the dotted box) to prevent overlap with sidebands.

From that diagram, it is tempting to think that the optimum solution would be a prefilter that passes everything inside the dotted box unchanged and completely stops the signal outside the box—like a cookie cutter. In other words, the sampled spectrum is to be multiplied by a function $K(f_x, f_y)$ which is equal to 1 inside the dotted box and equal to zero everywhere else. This is the two-dimensional *ideal low-pass filter,* and its kernel is:

$$k(x, y) = sinc(x)sinc(y) = \frac{sin(\pi x)}{\pi x}\,\frac{sin(\pi y)}{\pi y} \tag{16}$$

In practice, this ideal behavior is impossible. Actual filters cannot be infinite in width, and in fact there is a computational advantage to keeping the width of the reconstruction kernel as small as possible (e.g., the

integral in equation (6) must be performed over the width of the kernel). There are two consequences of using a non-ideal filter; the baseband (inside the dotted box in Figure 1) might be attenuated in some way, and some of the alias-causing sidebands might leak thru.

If a lot of sideband leakage occurs, then aliasing will result. If the baseband is strongly attenuated, then the image may be badly blurred. Unfortunately, even if a nearly ideal low-pass filter is used, severe *ringing* can occur around sharp edges in the image. Figure 22 illustrates this phenomenon in one dimension:

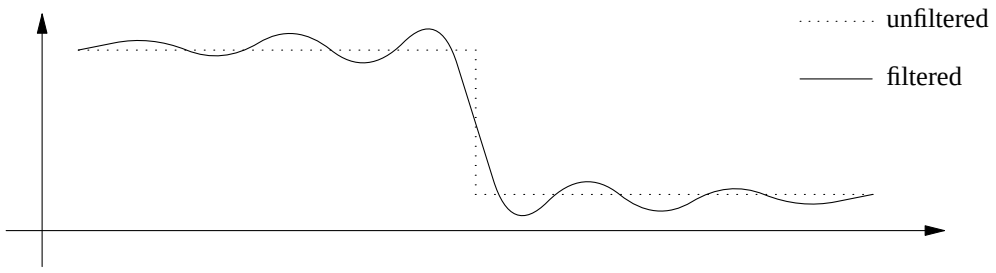


Figure 22. Ringing Caused by Ideal Low-Pass Filter

A tradeoff exists between ringing, blurring, and aliasing; and no filter will eliminate all three artifacts. The commonly-used box filter is poor because of the high degree of aliasing. Gaussian filters tend to cause problems with blurring or aliasing.

Classical digital filter design tends to emphasize close approximation of ideal low-pass behavior [Hamming83], but for images this causes too much ringing. It is also a problem that filters with sharp frequency cutoff are necessarily very wide, and this increases the computational expense of performing the convolution with $k(x, y)$. Some degree of ringing can be beneficial and gives a crisp appearance to an image, but this is somewhat risky since it may cause parts of the image to be clipped in intensity (if ringing dips below zero intensity or overshoots the maximum digital value).

Schreiber and Troxel consider these issues and recommend a sharpened Gaussian filter [Schreiber85]. Mitchell and Netravali recommend members of a two-parameter family of piecewise cubic filters [Mitchell88].

$$k(x) = \frac{1}{6}\begin{cases} (12-9B-6C)|x|^3 + & \text{if } |x|<1 \\ \quad (-18+12B+6C)|x|^2+(6-2B) & \\ (-B-6C)|x|^3+(6B+30C)|x|^2+ & \text{if } 1\le|x|<2 \\ \quad (-12B-48C)|x|+(8B+24C) & \\ 0 & \textit{otherwise} \end{cases} \tag{17}$$

In two dimensions, the separable kernel $k(x)k(y)$ is used. The (1/3, 1/3) filter seems to represent a good tradeoff between the various image-quality properties, suppressing ringing, blurring, and aliasing fairly well.

A second important application of digital filter design is in two-dimensional mapped texture. In a view of an object which greatly magnifies the texture, the subjective quality of the interpolation is very important. Both the sharpened Gaussian and the (1/3, 1/3) bicubic filters are designed to perform well under those circumstances, with a low degree of anisotropy and low visibility of sample-frequency structure.

When a view of a textured surface happens to greatly reduce a complex texture, this can create rather high

frequencies which place a demand on the ray tracer's antialiasing capabilities. An interesting question is whether or not texture-prefilting methods [Heckbert86], used in scan conversion, should be applied to ray tracing. Some of these algorithms provide the necessary capability of sampling the texture at an arbitrary location with an arbitrary amount of two-dimensional prefiltering (e.g., the sum-table method [Crow84]).

There are still open problems with regard to prefiltering texture in ray tracers. The interaction of the texture filter, and the ray tracer's antialiasing algorithm should be considered. Ray traced images often contain complex curved surfaces, and some prefiltering algorithms may not perform perfectly well (e.g., filters may extend into hidden portions of the surface). Perhaps the most difficult problem is prefiltering a texture for secondary rays. If a textured object is seen in the reflection of a curved surface, what is the proper amount of prefiltering?

8. References

[Amanatides84] Amanatides, John, "Ray Tracing with Cones", *Computer Graphics*, 18(3), July 1984, pp. 129-135.

[Amanatides89] Amanatides, John, "A Solid Angle Approach to Ray Tracing in Computer Graphics", PhD Thesis, University of Toronto, 1989.

[Amanatides90] Amanatides, John and Don P. Mitchell, "Antialiasing of Interlaced Video Animation", *Computer Graphics*, 24, August 1989.

[Catmull78] Catmull, Edwin, "A Hidden-Surface Algorithm with Anti-Aliasing", *Computer Graphics*, 12(3), July 1978, pp 6-10.

[Chuang85] Chuang, Richard, "Rendering for Television", SIGGRAPH Course Notes, 1985.

[Clark76] Clark, James H., "Hierarchical Geometric Models for Visible Surface Algorithms", *Comm. ACM*, 19(10), October 1976, pp 547-554.

[Clark85] Clark, James J., Matthew R. Palmer, and Peter D. Lawrence, "A Transformational Method for the Reconstruction of Functions from Nonuniformly Spaced Samples", *IEEE Trans. on Acoustics, Speech, and Signal Processing*, ASSP-33(4), October 1985, pp 1151-1165.

[Cochran77] Cochran, William G., *Sampling Techniques*, John Wiley & Sons, New York (1977).

[Cook84] Cook, Robert L, Thomas Porter, Loren Carpenter, "Distributed Ray Tracing", *Computer Graphics*, 18(3), July 1984, pp. 137-145.

[Cook85] Cook, Robert L, "Stochastic Sampling in Computer Graphics", SIGGRAPH 85 course notes.

[Cook86] Cook, Robert L, "Stochastic Sampling in Computer Graphics", *ACM Trans. Graphics*, 5(1), January 1986.

[Crochiere75] Crochiere, R. E., and L. R. Rabiner, "Optimum FIR Filter Implementations for Decimation, Interpolation, and Narrow-Band Filtering", *IEEE Trans. Acoustics, Speech, and Signal Processing*, October 1975, pp 444-456.

[Crow77] Crow, Franklin C., "The Aliasing Problem in Computer-Generated Shaded Images", *Comm. ACM*, 20(11), November 1977, pp 799-805.

[Crow84] Crow, Franklin C., "Summed-Area Tables for Texture Mapping", *Computer Graphics*, 18(3), July 1984, pp 207-212.

[Davies56] Davies, Owen L., *Design and Analysis of Industrial Experiments*, Hafner Publishing Company, New York, 1956.

[Dippé85] Dippe, Mark A. Z. and Erling Henry Wold, "Antialiasing Through Stochastic Sampling", *Computer Graphics*, 19(3), July 1985, pp. 69-78.

[Duff89] Duff, Tom, "Polygon Scan Conversion by Exact Convolution", *Proc. Raster Imaging and Digital Topography*, Academic Press, 1989.

[Garcia86] Garcia, A., "Efficient Rendering of Synthetic Images", PhD Thesis, Massachusetts Institute of Technology, February 1986.

[Hamming83] Hamming, R. W., *Digital Filters*, Prentice Hall, 1983.

[Heckbert84] Heckbert, Paul, Pat Hanrahan, "Beam Tracing Polygonal Objects", *Computer Graphics*, 18(3), July 1984, pp 119-127.

[Heckbert86] Heckbert, Paul, "Survey of Texture Mapping", *IEEE Computer Graphics and Applications*, 6(11), November 1986, pp 56-67.

[Hsu85] Hsu, Steve C., "Motion-Induced Degradations of Temporally Sampled Images", Master's thesis, MIT Department of Electrical Engineering, June 1985.

[Kajiya86] Kajiya, James T., "The Rendering Equation", *Computer Graphics*, 20(4), July 1986, pp. 143-150.

[Kirk87] Kirk, D.B., "The Simulation of Natural Features Using Cone Tracing", *Visual Computer*, 3(2), August 1987, pp 63-71

[Landau62] Landau, H. J. and J. O. Pollak, "Prolate Spheroidal Wave Functions, Fourier Analysis and Uncertainty—III: The Dimension of the Space of Essentially Time and Band-Limited Signals", *Bell System Tech. J.*, 41(4), pp 1295-1336, July 1962.

[Lee85] Lee, Mark, Richard A. Redner, Samuel P. Uselton, "Statistically Optimized Sampling for Distributed Ray Tracing", *Computer Graphics*, 19(3), July 1985, pp. 61-67.

[Limb69] Limb, J. O., "Design of Dither Waveforms for Quantized Visual Signals", *Bell System Tech. J.*, 48, pp. 2555-2582, 1969.

[Mertz34] Mertz, Pierre, and Frank Grey, "A Theory of Scanning and its Relation to the Characteristics of the Transmitted Signal in Telephotography and Television," *Bell System Tech. J.*, 13, pp. 464-515, July 1934.

[Mitchell87] Mitchell, Don P., "Generating Antialiased Images at Low Sampling Densities", *Computer Graphics*, 21(4), July 1987, pp. 65-72.

[Mitchell88] Mitchell, Don P., and Arun N. Netravali, "Reconstruction Filters in Computer Graphics", *Computer Graphics*, 22(4), August 1988, pp 221-228.

[Naiman87] Naiman, Avi and Alain Fournier, "Rectangular Convolution for Faster Filtering of Characters", *Computer Graphics*, 21(4), July 1987, pp 233-242.

[Norton82] Norton, A., A.P. Rockwood, P.T. Skolmoski, "Clamping: A Method of Antialiasing Textured Surfaces by Bandwidth Limiting in Object Space", *Computer Graphics*, 16(3), July 1982, pp 1-8.

[Painter89] Painter, James, and Kenneth Sloan, "Antialiased Ray Tracing by Adaptive Progressive Refinement", *Computer Graphics*, 23(3), July 1989, pp 281-288.

[Perlin85] Perlin, K., "An Image Synthesizer", *Computer Graphics*, 19(3), July 1985, pp 287-296.

[Petersen62] Petersen, Daniel P., David Middleton, "Sampling and Reconstruction of Wave-Number-Limited Functions in N-Dimensional Euclidean Spaces", *Information and Control*, 5, 1962, pp. 279-323.

[Phong75] Phong, Bui T., "Illumination for Computer Generated Pictures", *Comm. ACM*, 18(6), June 1975, pp 311-317.

[Potmesil83] Potmesil, M., I. Chakravarty, "Modeling Motion Blur in Computer-Generated Images", *Computer Graphics*, 17(3), July 1983, pp 389-399.

[Ripley77] Ripley, B. D., "Modeling Spatial Patterns", *J. Roy. Statist. Soc. B*, 39, 1977, pp. 172-212.

[Schreiber85] Schreiber, William F., Donald E. Troxel, "Transformation Between Continuous and Discrete Representations of Images: A Perceptual Approach", *IEEE Trans. Pattern Analysis and Machine Intelligence*, PAMI-7(2), March 1985, pp. 178-186.

[Shannon49] Shannon, C.E., "Communication in the presence of noise.", *Proc. IRE* 37, 1949, pp. 10-21.

[Thomas88] Thomas, David, *Personal Communications.*

[Thomas89] Thomas, David, Arun N. Netravali, D. S. Fox, "Antialiased Ray Tracing with Covers", *Computer Graphics Forum*, 8(4), December 1989, pp 325-336.

[Weiler77] Weiler, K., P. Atherton, "Hidden Surface Removal Using Polygon Area Sorting", *Computer Graphics*, 11(2), Summer 1977, pp 214-222.

[Whitted80] Whitted, Turner, "An Improved Illumination Model for Shaded Display", *Comm. ACM*, 23(6), June 1980, pp. 343-349.

[Yellott83] Yen, J. L., "On Nonuniform Sampling of Bandwidth-Limited Signals", *IRE Trans. Circuit Theory,* 3, Dec. 1 1956, pp. 251-257.