FUN WITH GAUSSIANS

Paul S. Heckbert

3-D Technical Memo 12
6 March 1985

Three Dimensional Animation Systems Group
Computer Graphics Lab
New York Institute of Technology

1. Introduction

We follow several uses for the gaussian bell curve on a quick romp through probability, statistics, combinatorics, signal processing, and computer graphics. Since it would be impossible in a paper this size to give introductions to all of these fields or to treat each of them rigorously, we will assume the reader has a certain level of familiarity with the subjects and attempt only to show some of the interdisciplinary ties. In particular, we discuss a simple piecewise polynomial approximation to the gaussian with applications in digital image filtering, interpolation, antialiasing, splines, random number generation, and blobby modeling. All this provides a convenient excuse to cram as many impressive-looking equations onto each page as possible.

2. Gaussians in Probability and Statistics

The ‘‘bell curve’’

$$e^{-x^2}$$

was first discussed in the context of probability by Abraham De Moivre in 1733 and developed independently later in the century by Pierre Laplace and Karl Friedrich Gauss [Schaaf64a] . The curve is usually attributed to Gauss, who discovered it in connection with his work in surveying and astronomy. Gauss noticed that precise quantitative measurements, such as the locations of stars in the sky or measurements of distances, vary according to a hump-shaped probability distribution†. More precisely, the measurements center around a *mean* value m, and the probability of a given value decreases rapidly as the value deviates above or below the mean. The uncertainty of the measurement (the width of the hump) is the *standard deviation* σ, which is often called the *root mean square error* in engineering.

Figure 1. A gaussian with mean m and standard devation σ.

Many empirically measured probability distributions approximate the function

$$g(m,\sigma,x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-m)^2}{2\sigma^2}}$$

which is called the *normal distribution of errors* or just *the gaussian*. Note that this function's integral, $\int_{-\infty}^{+\infty} g(m,\sigma,t)dt$, is 1, a requirement of all probability distributions. Most of the area under the curve is close to the mean, in fact 68.3% is within $\pm\sigma$ of the mean, 95.5% is within $\pm 2\sigma$, and 99.7% is within $\pm 3\sigma$. The inflection points of the gaussian are at $x \pm \sigma$. When normalized to have mean 0 and standard deviation 1, one gets the standard normal distribution:

$$g(0,1,x) = \mathrm{nor}(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$$

in terms of which the general gaussian probability distribution can be expressed:

$$g(m,\sigma,x) = \frac{1}{\sigma}\mathrm{nor}\left(\frac{x-m}{\sigma}\right)$$

Empirical measurements such as astronomical observations are subject to numerous sources of noise, for example the vibrations of the molecules in the measuring instrument. Why a multitude of small arbitrary errors results in the normal distribution of errors can be explained using the methods of random variables.

In statistics, any uncertain measurement (such as the position of a single atom or the azimuth of a telescope) is called a *random variable*. Random variables are variables which have a probability distribution rather than a single value. We will use boldface to set them apart. The probability that random variable **X** has value x is written:

---

† Clarification of terminology: Sometimes the term *probability distribution* is used for the integral of *probability density* but here we take the former to be synonymous with the latter.

$$\text{Prob}(\mathbf{X}=x) = f_{\mathbf{X}}(x)$$

Intuitively, $\mathbf{X}$ gives a different value x_i every time it is ''measured'', but as the number of samples grows, the histogram of measurements approaches the probability distribution $f_{\mathbf{X}}$. The mean or *expectation* of a random variable is defined like the centroid or first moment in physics:

$$m(\mathbf{X}) = E(\mathbf{X}) = \int_{-\infty}^{+\infty} x f_{\mathbf{X}}(x)dx$$

and the variance, which is the square of the standard deviation, is:

$$\sigma^2(\mathbf{X}) = E(\mathbf{X}^2) - E(\mathbf{X})^2$$

$$\text{where} \quad E(\mathbf{X}^2) = \int_{-\infty}^{+\infty} x^2 f_{\mathbf{X}}(x)dx$$

Using these definitions, it's easy to verify that the mean and standard deviation of $g(m,\sigma,x)$ are m and σ.

The continuous definitions of mean and variance above are very similar to the more familiar discrete ones:

$$m = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \sigma^2 = E\left((\mathbf{X} - E(\mathbf{X}))^2\right) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

which can be used to compute the approximate mean and variance when only a few sample values x_i for $i=1,2,...,n$ of a random variable are known.

A less well-known, but handier, formula for variance is:

$$\sigma^2 = E(\mathbf{X}^2) - E(\mathbf{X})^2 = \frac{\sum x_i^2}{n} - \left(\frac{\sum x_i}{n}\right)^2$$

which is useful for on-line calculation, since it entails only one pass over the data.

What will be the combined influence of many random variables, such as the vibrating atoms in Gauss' telescope? Let's try adding two of them together. When *independent* random variables $\mathbf{X}$ and $\mathbf{Y}$ are added, the probability that the random variable for their sum $\mathbf{Z}=\mathbf{X}+\mathbf{Y}$ has value z is the sum of the probabilities that $\mathbf{X}=x$ and $\mathbf{Y}=y$ for all values of x and y that add to z:

$$\text{Prob}(\mathbf{Z}=z) = f_{\mathbf{Z}}(z) = \int_{-\infty}^{+\infty} f_{\mathbf{X}}(x) f_{\mathbf{Y}}(z-x)dx$$

which is just convolution: $f_{\mathbf{Z}} = f_{\mathbf{X}} * f_{\mathbf{Y}}$ (the '*' denotes convolution). **So addition of independent random variables corresponds to convolution of their probability distributions**.

From here, it is easy to verify that when independent random variables are added, their means and variances add: $E(\mathbf{X}+\mathbf{Y})=E(\mathbf{X})+E(\mathbf{Y})$ and $\sigma^2(\mathbf{X}+\mathbf{Y})=\sigma^2(\mathbf{X})+\sigma^2(\mathbf{Y})$. Put another way, **the mean of the sum is the sum of the means** and **the variance of the sum is the sum of the variances**.

If a random variable is added to itself n times, the mean and variance grow linearly. But what happens to the shape of the distribution? In figure 3 we can see that self-convolution of box-shaped distributions become smoother, broader, and more gaussian shaped as the number of convolutions increases. The *Central Limit Theorem* tells us that this is true for almost any probability distribution† [Baclawski77a] :

> **The sum of n arbitrary independent random variables approaches a normal distribution as $n \to \infty$.**

Unfortunately, a proof of this theorem is one of the few things beyond the scope of this paper.

The Central Limit Theorem explains why the normal distribution is so commonly observed: it is a mathematical consequence of having numerous sources of noise, regardless of each one's character. In the words of Henri Poincare [Newman56a] :

> *We need know only one thing: that the errors are very numerous, that they are very slight, that each may be as well negative as positive. What is the curve of probability of each of them? We do not know; we only suppose that it is symmetric. We prove then that the resultant error will follow Gauss' law, and this resulting law is independent of the particular laws which we do not know. Here again the simplicity of the result is born of the very complexity of the data.*

The *entropy* of a continuous distribution can be defined as:

$$H(\mathbf{X}) = -\int_{-\infty}^{+\infty} f_{\mathbf{X}}(x)\log_2 f_{\mathbf{X}}(x)\,dx$$

thus yielding a quantitative measure of a random variable's information content or uncertainty. It turns out that, of all distributions having a given standard deviation, the gaussian has maximum entropy [Baclawski77a] .

3. Gaussians in Combinatorics

The simplest non-trivial random variable is an event which has only two possible outcomes, like flipping a coin. We can represent heads as **X**=1 and tails as **X**=0. This random variable's distribution function will be zero except for two spikes at 0 and 1. Let Prob(*heads*)=p and Prob(*tails*)=q where $p+q=1$. If $p \neq 1/2$ then the coin is said to be *biased*. If we flip this coin n times and add the random variables, the resulting random variable is called the *Bernoulli process*. The probability of getting k heads after n flips has the *binomial distribution*:

$$\text{Prob}(\mathbf{X}=k) = \binom{n}{k} p^k q^{n-k}$$

Which is a term from the binomial expansion:

† The Central Limit Theorem only applies when the distribution has finite mean and variance.

$$(p+q)^n = \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k}$$

The binomial coefficients $\binom{n}{k}$ are better known as Pascal's triangle. Since the Bernoulli process is the sum of many random variables, by the central limit theorem, the rows of Pascal's triangle must approach a gaussian as $n \to \infty$ [Feynman63a] .

Can we find a formula relating the two? We will try to find A, m, and σ such that $Ag(m,\sigma,k) \approx \binom{n}{k}$. Taking a fair coin $p=q=1/2$, it's clear that the average number of heads after n tosses will be $m=n/2$. Next, we match integrals:

$$\int_{-\infty}^{+\infty} Ag(m,\sigma,k)dk = A = \sum_{k=0}^{n} \binom{n}{k} = 2^n$$

So $A=2^n$. Finally, to match peak values, we use Stirling's formula, $n! \approx \left(\frac{n}{e}\right)^n \sqrt{2\pi n}$, to approximate the binomial coefficients. We find that

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \approx \frac{n^n}{k^k (n-k)^{n-k}} \sqrt{\frac{n}{2\pi k(n-k)}}$$

Equating peak values:

$$2^n g(m,\sigma,m) = \frac{2^n}{\sigma\sqrt{2\pi}} = \binom{n}{\frac{n}{2}} \approx \frac{2^{n+1}}{\sqrt{2\pi n}}$$

and the standard deviation is $\sigma = \frac{\sqrt{n}}{2}$.

Therefore,

$$\binom{n}{k} \approx 2^n g\left(\frac{n}{2}, \frac{\sqrt{n}}{2}, k\right) = \frac{2^{n+1}}{\sqrt{2\pi n}} e^{-\frac{2(k-\frac{n}{2})^2}{n}}$$

Expansion of the binomial series $(p+q)^n$ is analogous to convolution of sequences, since multiplication of power series involves convolution of their coefficients

$$\left(\sum_{i=0}^{\infty} a_i x^i\right)\left(\sum_{j=0}^{\infty} b_j x^j\right) = \sum_{k=0}^{\infty} x^k \sum_{i=0}^{k} a_i b_{k-i}$$

The sum on the far right is the discrete convolution of a_i and b_j. As a notation for convolution of sequences, we can write $[1,\ 1]^{*2} = [1,\ 1]*[1,\ 1] = [1,\ 2,\ 1]$ and $[1,\ 1]^{*3} = [1,\ 3,\ 3,\ 1]$, etc, where f^{*n} means $f*f*f*...*f$ (n times). Convolution of sequences is equivalent to multiplication of polynomials.

Using the impulse function $\delta(t)$, which is roughly defined as:

$$\delta(t) = \begin{cases} \infty & t=0 \\ 0 & t\neq 0 \end{cases} \qquad \text{where} \int_{-\infty}^{+\infty} \delta(t)dt = 1$$

to relate the continuous and the discrete, we can write the probability distribution for n coin tosses as

$$(p\,\delta(x)+q\,\delta(x-1))^{*n}$$

In this notation,

$$(p\,\delta(x)+q\,\delta(x-1))^{*n} = \sum_{k=0}^{n} \binom{n}{k} p^k q^{n-k}\,\delta(x-k)$$

Or, in sequence notation,

$$[p,\ q]^{*n} = [\ \binom{n}{k} p^k q^{n-k} \text{ for } k=0,1,...,n\ ]$$

We can relate this distribution to the gaussian another way by making use of the additive properties of mean and variance. Using the discrete definitions of mean and variance, we find $m([p,\ q])=q$ and $\sigma^2([p,\ q])=pq$ for one toss, so $m([p,\ q]^{*n})=nq$ and $\sigma^2([p,\ q]^{*n})=npq$ for n tosses. Therefore, the gaussian approximation to a biased binomial distribution is:

$$\binom{n}{k} p^k q^{n-k} = g(nq, \sqrt{npq}, k) = \frac{1}{\sqrt{2\pi npq}} e^{-\frac{(k-nq)^2}{2npq}}$$

which verifies the approximation to $\binom{n}{k}$ derived earlier [Baclawski77a] .

Many other stochastic processes such as the random walk or the Brownian motion of molecules in a gas obey binomial or normal distributions as discussed in [Feynman63a] .

4. Gaussians in Signal Processing

The Fourier transform represents an arbitrary function using sums of sine waves of different frequencies [Bracewell78a] . It transforms functions in the spatial or time domain $f(t)$ into the frequency domain $F(\omega)$ and vice versa. If $f(t)$ and $F(\omega)$ are such a *transform pair*, we write $f(t) \longleftrightarrow F(\omega)$. Before proceeding any farther, some synonyms:

spectrum = frequency response
filter (the verb) = convolve
filter (the noun) = kernel = window
= impulse response = point spread function

(It's not surprising that there are so many names for each concept considering how many branches of science and engineering have embraced the Fourier transform and the theory of linear systems.)

One of the Fourier transform's most interesting properties is that convolution in one domain corresponds to multiplication in the other: $f*g(t) \longleftrightarrow FG(\omega)$ and

$fg(t) \longleftrightarrow F^{*}G(\omega)$.

The Fourier transform also tells us that a finite spatial function (a kernel which is zero outside some finite domain) has an infinite spectrum (contains arbitrarily high frequencies). For example, the Fourier transform of the common box window (Fourier window)

$$box(t) = \begin{cases} 1 & |t| < 1/2 \\ 0 & |t| \geq 1/2 \end{cases}$$

is the sinc function: $sinc(\omega) = \frac{sin(\pi\omega)}{\pi\omega}$. This and several other transform pairs are pictured below.

The envelope of sinc is like $1/\omega$, which decays very slowly. Slow decay implies that the box filter lets through a lot of high frequencies. Conversely, since the inverse Fourier transform is almost the same as the forward transform, a box in the frequency domain (the ideal low pass filter), transforms to sinc in the spatial domain.

The ideal low pass filter is very desirable when reconstructing a sampled signal, but unfortunately its infinitely wide kernel makes it impractical. In practice, we must use finite impulse response (FIR) filters, whose frequency responses are necessarily less than ideal [Whitted81a] . When doing convolution one must trade off between the slowness of a wide kernel and the poor quality (excessive high frequencies) of a narrow one. At one extreme is the low cost, low quality box and at the other extreme is the high cost, high quality sinc. The gaussian provides a happy compromise between the two. In fact, **the Fourier transform of a gaussian is a gaussian**, so the kernel and its spectrum have similar shape. The gaussian decays quickly so it can be truncated with less error than a sinc. To truncate $g(m,\sigma,x)$ at threshold T, we equate g and T and solve for x, yielding: $x = m \pm \sigma\sqrt{-2\log(T\sqrt{2\pi\sigma})}$.

A gaussian is a less-than-ideal low pass filter, but it comes close while retaining a practically finite kernel. Many of true FIR filters discussed in the literature (such as the Hanning, Hamming, Blackman, and Kaiser windows [Oppenheim75a]) have better frequency responses than the gaussian, but they lack its simplicity.

Figure 3. The box^{*n} kernels for n=1-4.

5. Polynomial Approximations to the Gaussian

We can generate piecewise polynomial approximations to the gaussian by repeatedly convolving box windows. We denote the resulting function box^{*n}. A single box (often called a ''sample and hold'' in electrical engineering) has discontinuities at $x=\pm 1/2$. Two boxes convolved make a triangular window (Bartlett window), which has two linear intervals with continuity of position. Three boxes convolved make a parabolic window, which has three quadratic intervals and continuity of first derivative. Four boxes convolved are called a Parzen window, which has four cubic intervals and continuity of the second derivative. When n boxes are convolved, the result will have n intervals each a degree $n-1$ polynomial, a *support* (width) of n at the base, and continuity of its $n-2$ nd derivative. Above $n=3$, the shape is almost indistinguishable from a gaussian. As n increases, both the kernel and its Fourier transform converge on the gaussian. The formulas are given below, and graphs are shown in figure 3.

$n=1$ BOX $box(x) = \begin{cases} 1 & -\frac{1}{2} < x < \frac{1}{2} \\ 0 & |x| \geq \frac{1}{2} \end{cases}$

$n=2$ BARTLETT $box^{*2}(x) = \begin{cases} 1+x & -1 \leq x \leq 0 \\ 1-x & 0 \leq x \leq 1 \\ 0 & |x| \geq 1 \end{cases}$

$n=3$ PARABOLIC $box^{*3}(x) = \begin{cases} \frac{1}{2}(x+\frac{3}{2})^2 & -\frac{3}{2} \leq x \leq -\frac{1}{2} \\ \frac{3}{4}-x^2 & -\frac{1}{2} \leq x \leq \frac{1}{2} \\ \frac{1}{2}(x-\frac{3}{2})^2 & \frac{1}{2} \leq x \leq \frac{3}{2} \\ 0 & |x| \geq \frac{3}{2} \end{cases}$

$n=4$ PARZEN $box^{*4}(x) = \begin{cases} \frac{(2+x)^3}{6} & -2 \leq x \leq -1 \\ \frac{4-6x^2-3x^3}{6} & -1 \leq x \leq 0 \\ \frac{4-6x^2+3x^3}{6} & 0 \leq x \leq 1 \\ \frac{(2-x)^3}{6} & 1 \leq x \leq 2 \\ 0 & |x| \geq 2 \end{cases}$

There does not seem to be a precise relationship between these functions and Pascal's triangle. However, the box^{*n} functions relate quite nicely to splines; as we shall see, they are identical to the uniform B-spline basis functions.

6. Gaussians in Image Processing

The gaussian has a number of properties which make it especially attractive for image processing: it is a positive kernel, in two or more dimensions it is separable and circularly symmetric, and the polynomial approximations to it admit a very efficient gaussian low pass filter (blur).

Images are represented by two-dimensional intensity functions which are always non-negative. Therefore, convolution with negative-lobed kernels such as sinc can generate unrealizable negative intensities. This problem never occurs with gaussian filters, however, since the gaussian is a positive function. This suggests several interesting questions:

(a) Is the gaussian the ''best'' low pass filter with a positive kernel?

(b) Probability distributions and non-negative filter kernels are both non-negative functions which integrate to 1. Is there a fundamental similarity between them?

Perhaps its most attractive feature for image processing is that the gaussian is both *separable* and circularly symmetric. A 2-dimensional function $f(x,y)$ is said to be separable when it can be written as $g(x)h(y)$. The circularly symmetric gaussian is proven separable by the following:

$$e^{-r^2} = e^{-x^2-y^2} = e^{-x^2}e^{-y^2}$$

Separable kernels are very desirable because they allow 2-dimensional convolution to be decomposed into two 1-dimensional convolutions. Straightforward convolution of a $W \times W$ kernel with an $M \times N$ image requires W^2MN multiply-adds, but exploiting separability, the cost is reduced to $W^2(M+N)$. This can make the difference between practical and impractical computations. For example, on a VAX 11/780, a multiply-add takes about 15 μsec, so convolution of a 500x500 image with a 35x35 kernel would take over an hour with 2-D convolution, but only 4 minutes with two 1-D convolutions. While straightforward 2-D convolution is fastest for small kernels (say 3×3), and separable convolution is good for medium-size kernels, a third alternative which is best for large, non-separable kernels is *Fourier convolution*: transformation of image and kernel into the frequency domain by FFT, multiplication of the two spectra, followed by an inverse FFT.

A variation on the gaussian filter, the *sharpened gaussian*, has been proposed for image magnification [Schreiber85a] . Its negative lobes give it better sharpness than the gaussian without degrading the circular symmetry substantially.

Two ideas previously described, the separability of gaussians and their approximation by box^{*n}, can be combined to make an efficient low pass image filter algorithm, but further groundwork is needed before it can be explained.

Recalling that addition of independent random variables is equivalent to convolution of their probability distributions, and that filter kernels are like probability distributions, we see that for two kernels f and g **the mean of the convolution equals the sum of the means**:

$$m_{f*g} = m_f + m_g$$

and **the variance of the convolution equals the sum of the variances**:

$$\sigma^2_{f*g} = \sigma^2_f + \sigma^2_g$$

Also, the Central Limit Theorem tells us that the convolution of n arbitrary kernels approaches a gaussian as $n \to \infty$, whose mean grows proportional to n and standard deviation grows proportional to $\sqrt{n}$. The standard deviation of the continuous function box^{*n} is $\sqrt{n/12}$. These powerful laws help us compare box and gaussian low pass filters.

The low pass filters box^{*n} can be applied by either explicitly convolving the signal with a box n times or by pre-computing box^{*n} and then convolving it with the signal once. An example of the first method is the repeated use of the blur program on an image. The command blur m convolves the image with a wide box kernel $m \times m$ pixels in size. A discrete box with m terms, $[\frac{1}{m}, \frac{1}{m}, \cdots, \frac{1}{m}]^{*n}$, which is the kernel resulting from n applications of blur m, has standard deviation $\sqrt{\frac{n(m^2-1)}{12}}$. This relation allows us to say, for example, that three passes of blur m have nearly the same effect as a gaussian blur with standard deviation $m/2$ such as gauss m.

Normally, convolution with an $m \times m$ kernel has cost proportional to m^2. The constant of proportionality can be made quite small by simplifying the inner loop [Greene85a] but straightforward 2-D convolution is still slow for large m. The blur program takes advantage of the box window to make its cost independent of window size. It exploits the fact that the box window is an unweighted average which can be computed incrementally. The *summed area table* [Crow84a] uses a similar technique, performing box window filtering on texture maps with constant cost by pre-integrating the table. Perlin has generalized this quite elegantly for the box^{*n} filters [Perlin85a] . We can understand his method given two more facts about Fourier transforms.

Differentiation in the spatial domain corresponds to multiplication by $i\omega$ in the frequency domain.

Integration in the spatial domain corresponds to division by $i\omega$ in the frequency domain.

We are now ready to understand Perlin's identity† (which should be read in clockwise order):

$$f(t) * \text{box}^{*n} \quad \longleftrightarrow \quad F(\omega)\text{sinc}^n(\omega)$$

$$\left[\int^n f(t)dt^n\right] * (\text{box})^{*n} \quad \longleftrightarrow \quad \frac{F(\omega)}{(i\,\omega)^n}[i\,\omega\ \text{sinc}(\omega)]^n$$

This identity says that another way to convolve with box^{*n} is to integrate the signal n times as a pre-process, and then convolve with $(box)'^{*n}$. But box' is the difference of two impulse functions (figure 4) – a trivial convolution filter, which generates $n+1$ sample points when convolved with itself n times. (Note that the Fourier transforms here are used only for proof, not in the implementation. Also note that in practice the signal f and the box window are discrete, not continuous.)

Figure 4. The derivative of *box* is two impulse functions: $\delta(x+.5)-\delta(x-.5)$.

Perlin's filter, then, is a technique for convolving a 1-D signal with an approximately gaussian kernel with cost proportional to the order of the approximation, but independent of the size of the kernel. For image (2-D) filtering, there are $(n+1)^2$ sample points, so the cost rises like the square of the order of approximation. Perlin's method is equivalent to the summed area table if $n=1$, but for $n=2$ it gives Bartlett filtering and for $n=3$ it gives parabolic filtering, which is very close to gaussian. Perlin calls it a *selective image filter* because it is useful for spatially variant filters. (If it were only good for space invariant filters, as blur and gauss are, we'd do better for large kernels by using an FFT.) Unfortunately, Perlin's technique does not allow filtering of arbitrarily oriented ellipses, which is required for good texture filtering [Greene85a] .

Other disadvantages of the method are the storage and numerical problems resulting from the large dynamic range. For a $2^r \times 2^r$ monochrome image with b bits per pixel, 2-D integration n times can result in numbers as large as 2^{2nr+b}. For example, an 8-bit 512x512 image has $r=9$ and $b=8$, so 26 bits are required for $n=1$ (summed area tables), 44 bits are needed for $n=2$, and 62 bits for $n=3$.

† see also fig. 7.2 of [Bracewell78a] .

7. Gaussians in Signal Reconstruction and Spline Interpolation

A very popular technique for smooth interpolation of data points is the *spline*: a continuous, piecewise polynomial function. Generally, the higher degree polynomials allow greater continuity at the *knots* between intervals but often introduce undesirable ripples. They're also computationally expensive. Consequently, the most popular splines have low degree, often three.

When the data points are equally spaced, interpolation is analogous to reconstruction of a sampled signal as in signal processing. The Sampling Theorem [Bracewell78a] says that a function can be reconstructed exactly if the original signal had no frequencies higher than half the rate at which it was sampled. Given a signal $f(x)$ sampled at integer values of x, convolution with a continuous kernel h results in the reconstructed signal f':

$$f'(x+t) = \sum_{i=-\infty}^{+\infty} f(x+i)h(t-i)$$

where t is a fractional offset between 0 and 1. Such a filter, which is analogous to a weighted average of the sample values, is often needed in 2-D image processing when an image is being scaled up or shifted by a fraction of a pixel [Greene85a] , [Smith83a] .

The ideal reconstruction kernel, the sinc, is not an FIR filter, as discussed earlier, so the resulting infinite sum makes it computationally intractable. Reconstruction using an FIR filter results in a less-than-ideal frequency response which can cause reconstruction errors called *rastering*, but this is unavoidable because the impulse response and frequency response cannot both be finite.

For digital reconstruction of signals we desire kernels with small, finite *support* which are both computationally inexpensive and ''good'' low pass filters. We've already seen how to generate piecewise polynomial approximations to a fairly good low pass filter: the gaussian. In the spline field these box^{*n} kernels are very well known [Gordon74a] , in fact:

$$box^{*n} = \text{uniform B-spline}$$

The box^{*n} filters have different meaning here than they did for the image blurring application. Whereas for blurring the *box* was a wide, discrete signal, in the present context it is continuous and only one sample in width. Most of the terminology is different as well: the filters are called *splines* rather than filters, the kernel polynomials are called *basis functions*, and the sample points are called *uniformly spaced knots*.

The table below summarizes the properties and common names for the lowest order B-splines.

order	deg	filter name	spline name	interp	continuity
n=1	0	Fourier, box	point sampling	yes	none
n=2	1	Bartlett	linear interp.	yes	C^0 (value)
n=3	2	parabolic	quadratic B-spline	no	C^1 (tangent)
n=4	3	Parzen	cubic B-spline	no	C^2 (curvature)

A graphical comparison of B-splines is shown in figure 5. We see that the first two, point sampling and linear interpolation, interpolate the sample points while the higher order B-splines do not. The quadratic B-spline does, however, always interpolate the midpoints between samples.

When using the box^{*n} kernels as basis functions it is best to reparameterize them as polynomials in the fractional offset t:

$$f_n(x+t) = \sum_{i=-\infty}^{+\infty} f(x+i)h_i(t)$$

Except for n intervals centered on $x+t$, all of the basis functions h_i are 0, so the support of the spline is n samples.

In spline literature, it is customary to write the basis functions as $n \times n$ matrices rather than polynomials. The first four box^{*n} / B-splines are listed below:

Figure 5. Example B-splines of order 1-4.

order 1: point sampling:

$$f_1(x+t-1/2) = [1][1][f(x)]$$

order 2: linear interpolation:

$$f_2(x+t) = \begin{bmatrix} t & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} f(x) \\ f(x+1) \end{bmatrix}$$

order 3: quadratic:

$$f_3(x+t-1/2) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & -2 & 1 \\ -2 & 2 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} f(x-1) \\ f(x) \\ f(x+1) \end{bmatrix}$$

order 4: cubic:

$$f_4(x+t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} f(x-1) \\ f(x) \\ f(x+1) \\ f(x+2) \end{bmatrix}$$

The basis functions described by the matrices above are identical to the *box*^{*n} piecewise polynomial functions listed earlier except they've been reparameterized to the interval [0,1]. To interpolate a sampled function, the abscissa x' is broken into ''integer'' and ''fractional'' parts x and t, respectively. This is done differently for even n and odd n:

$$\begin{array}{lll} \text{for even } n, & x = int(x'), & t = x' - x \\ \text{for odd } n, & x = int(x' + 1/2), & t = x' - x + 1/2 \end{array}$$

These formulas ensure that the sample points which are weighted together are the nearest neighbors of x'

B-splines have the property that any point on the spline lies within the convex hull of the neighboring n knots [Gordon74a] . This is the case because all points on the spline are a weighted average *with positive weights* of the knots in their support, and all such points lie within the convex hull. The positivity of the basis functions is a result of the gaussian's positivity.

Evaluation of the matrix products in the formulas above requires on the order of n^2 multiplies and n^2 adds†. If speed is a great concern, in many situations the n polynomial evaluations can be done by table lookup, in which case the cost reduces to n multiplies and $n-1$ adds. If the basis functions are in a table, however, the cost of computing them becomes irrelevant, so it pays to use the best low pass filter for a

† Actually, we can do quite a bit better than that: $n=1$ requires 0×, 0+, $n=2$ requires 1×, 2+, $n=3$ requires 6×, 6+, and $n=4$ requires 7×, 10+.

given support, which is probably not an approximation to a gaussian. The cubic Catmull-Rom basis is a popular alternative to the B-spline since it interpolates [Smith83a] .

The splines we've been discussing can be generalized to multidimensional lattices of data points simply by repeated application. For instance, 2-D bilinear interpolation requires 4 sample accesses and 3 linear interpolations (1 for each of the 2 rows and 1 between rows). Use of the parabolic spline in 3-D would require 27 sample accesses and 9+3+1=13 quadratic interpolations. In general, the d-dimensional spline requires n^d samples and $(n^d-1)/(n-1)$ nth order interpolations.

8. Gaussian Random Numbers

Computer graphics and scientific programmers often need to generate pseudorandom numbers with gaussian distributions. A simple way to do this is to sum n uniform random numbers. As we saw earlier, adding independent random variables is analogous to convolving their distributions, so summing n uniform, independent random numbers $\mathbf{U}_i$ each between 0 and 1 results in a box^{*n} distribution, exactly. Subtracting the mean and dividing by the standard deviation results in a standard normal distribution:

$$\text{if} \qquad \mathbf{N} = \frac{\sum^{n} \mathbf{U}_i - n/2}{\sqrt{n/12}} \qquad \text{then} \qquad \text{Prob}(\mathbf{N}=x) \approx \text{nor}(x)$$

Unfortunately, this method is only approximate, and will never generate arbitrarily large numbers, as a true gaussian would. More accurate techniques for generating gaussian random numbers are discussed in [Knuth69a] .

9. Gaussians in Quantum Mechanics

Heisenberg's uncertainty principle says that a particle's position x and its momentum p cannot both be known with perfect certainty. Associated with x and p are complex *wave functions* forming a Fourier transform pair whose magnitudes determine probability distributions for the two variables [Powell61a] . The standard deviations of these distributions, Δx and Δp, respectively, are used to quantify the uncertainties. Heisenberg's principle states that their product is greater than or equal to a constant:

$$\Delta x \, \Delta p \geq \frac{h}{2}$$

This is similar to the scaling property of the Fourier transform, which states that a kernel's width is inversely proportional to the width of its spectrum. Heisenberg's relation is an equality only if the wave packet has gaussian shape [Feynman63a] . All other distributions result in loss of information about position or momentum.

10. Gaussians in Blobby Modeling

As an alternative to the usual ‘‘ball and stick’’ models of molecules, Blinn has simulated 3-D gaussian electron density distributions in order to make pictures of the Van der Waals surfaces of molecules [Max83a] . The technique, usually called *blobby modeling*, has proven useful for modeling many other things, especially metamorphosing, organic forms. The models are usually rendered using ray tracing, wherein intersection points are calculated between rays from the eye and the contour surfaces of the distribution. Since the cross section of a gaussian hump is a gaussian curve, the density function along any ray is a superposition of gaussians. Unfortunately, the roots of such a function cannot be found analytically, so Blinn resorted to heuristics and numerical methods [Blinn82a] . If the box^{*3} approximation is used, however, the distribution is piecewise quadratic, and roots can be found using the quadratic formula [Kawai85a] .

11. Acknowledgements

Thanks to John Lewis for asking about efficient interpolation techniques and to Pat Hanrahan for his comments on the paper.

12. References

Baclawski77a.

Kenneth Baclawski and Gian-Carlo Rota, *An Introduction to Probability and its Applications*, (MIT 18.313 course text) 1977.

Blinn82a.

James F. Blinn, ‘‘A Generalization of Algebraic Surface Drawing,’’ *ACM Transactions on Graphics* **1**(3) pp. 235-256 (July 1982).

Bracewell78a.

Ronald N. Bracewell, *The Fourier Transform and Its Applications*, McGraw-Hill, New York (1978).

Crow84a.

Franklin C. Crow, ‘‘Summed-Area Tables for Texture Mapping,’’ *Computer Graphics (SIGGRAPH ’84 Proceedings)* **18**(3) pp. 207-212 (July 1984).

Feynman63a.

Richard P. Feynman, Robert B. Leighton, and Matthew Sands, *The Feynman Lectures on Physics*, Addison-Wesley, Reading, MA (1963). vol. 1, chap. 6

Gordon74a.

William J. Gordon and Richard F. Riesenfeld, ‘‘B-spline Curves and Surfaces,’’ pp. 95-126 in *Computer Aided Geometric Design*, ed. Robert E. Barnhill and Richard F. Riesenfeld,Academic Press, New York (1974).

Greene85a.

Ned Greene and Paul S. Heckbert, *Creating Raster Omnimax Images from Multiple Perspective Views*, submitted to SIGGRAPH ’85 July 1985.

Kawai85a.
Toshiyuki Kawai, Makoto Hirai, Hitoshi Nishimura, Toru Kawata, Isao Shirakawa, and Koichi Omura, *A 3D Shape Generation Scheme by Means of Metaballs,* submitted to SIGGRAPH '85 July 1985.

Knuth69a.
Donald E. Knuth, *The Art of Computer Programming vol. 2, Seminumerical Algorithms,* Addison-Wesley, Reading, MA (1969).

Max83a.
Nelson L. Max, ''Computer Representation of Molecular Surfaces,'' *IEEE Computer Graphics and Applications* **3**(7) pp. 21-29 (Aug. 1983). reprinted in Nicograph '83 Proceedings, 1983, pp. 323-331.

Newman56a.
William R. Newman, *The World of Mathematics,* Simon and Schuster, New York (1956).

Oppenheim75a.
Alan V. Oppenheim and Ronald W. Schafer, *Digital Signal Processing,* Prentice-Hall, Englewood Cliffs, NJ (1975).

Perlin85a.
Kenneth Perlin, *Selective Image Filtering,* to appear, SIGGRAPH '85 State of the Art in Image Synthesis seminar notes July 1985.

Powell61a.
John L. Powell and Bernd Crasemann, *Quantum Mechanics,* Addison-Wesley, Reading, MA (1961).

Schaaf64a.
William L. Schaaf, *Carl Friedrich Gauss, Prince of Mathematicians,* Franklin Watts, New York (1964).

Schreiber85a.
William F. Schreiber and Donald E. Troxel, ''Transformation Between Continuous and Discrete Representations of Images: A Perceptual Approach,'' *IEEE Trans. on Pattern Analysis and Machine Intelligence* **PAMI-7**(2)(Mar. 1985).

Smith83a.
Alvy Ray Smith, ''Spline Tutorial Notes,'' pp. 64-75 in *SIGGRAPH '83 Introduction to Computer Animation seminar notes,* (July 1983).

Whitted81a.
Turner Whitted, ''The Causes of Aliasing in Computer Generated Images,'' in *SIGGRAPH '81 Advanced Image Synthesis seminar notes,* (Aug. 1981).