# Chapter 9

# Basic Signal Processing

## Motivation

Many aspects of computer graphics and computer imagery differ from aspects of conventional graphics and imagery because computer representations are digital and discrete, whereas natural representations are continuous. In a previous lecture we discussed the implications of quantizing continuous or high precision intensity values to discrete or lower precision values. In this sequence of lectures we discuss the implications of sampling a continuous image at a discrete set of locations (usually a regular lattice). The implications of the sampling process are quite subtle, and to understand them fully requires a basic understanding of signal processing. These notes are meant to serve as a concise summary of signal processing for computer graphics.

### Reconstruction

Recall that a framebuffer holds a 2D array of numbers representing intensities. The display creates a continuous light image from these discrete digital values. We say that the discrete image is *reconstructed* to form a continuous image.

Although it is often convenient to think of each 2D pixel as a little square that abuts its neighbors to fill the image plane, this view of reconstruction is not very general. Instead it is better to think of each pixel as a point sample. Imagine an image as a surface whose height at a point is equal to the intensity of the image at that point. A single sample is then a “spike;” the spike is located at the position of the sample and its height is equal to the intensity associated with that sample. The discrete image is a set of spikes, and the continuous image is a smooth surface fitting the spikes as shown in Figure 9.1. One obvious method of forming the continuous surface is to *interpolate* between the samples.

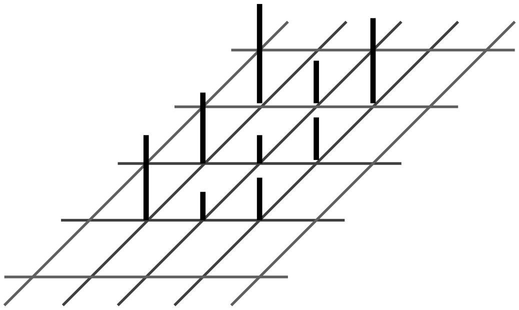


Figure 9.1: A continuous image reconstructed from a discrete image represented as a set of samples. In this figure, the image is drawn as a surface whose height is equal to the intensity.

## Sampling

We can make a digital image from an analog image by taking samples. Most simply, each sample records the value of the image intensity at a point.

Consider a CCD camera. A CCD camera records image values by turning light energy into electrical energy. The light sensitive area consist of an array of small cells; each cell produces a single value, and hence, samples the image. Notice that each sample is the result of all the light falling on a single cell, and corresponds to an integral of all the light within a small solid angle (see Figure 9.2). Your eye is similar, each sample results from the action of a single photoreceptor. However, just like CCD cells, photoreceptor cells are packed together in your retina and integrate over a small area. Although it may seem like the fact that an individual cell of a CCD camera, or of your retina, samples over an area is less than ideal, the fact that intensities are averaged in this way will turn out to be an important feature of the sampling process.

A vidicon camera samples an image in slightly different way than your eye or a CCD camera. Recall that television signal is produced by a raster scan process in which the beams moves continuously from left to right, but discretely from top to bottom. Therefore, in television, the image is continuous in the horizontal direction. and sampled in the vertical direction.

The above discussion of reconstruction and sampling leads to an interesting question: Is it possible to sample an image and then reconstruct it without any distortion?

## Jaggies, Aliasing

Similarly, we can create digital images directly from geometric representations such as lines and polygons. For example, we can convert a polygon to samples by testing

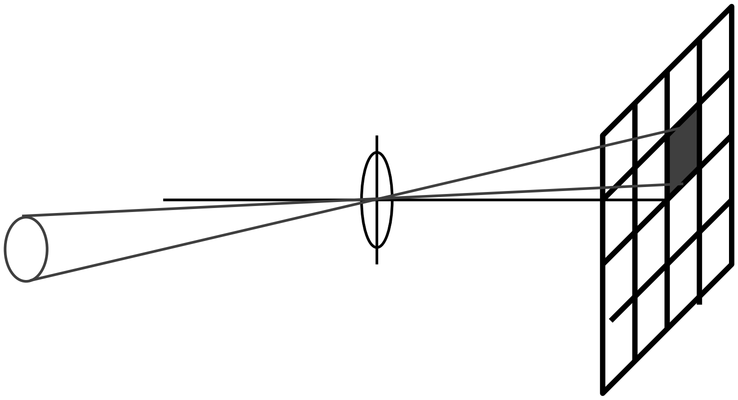


Figure 9.2: A CCD camera. Each cell of the CCD array receives light from a small solid angle of the field of view of the camera. Thus, when a sample is taken the light is averaged over a small area.

whether a point is inside the polygon. Other rendering methods also involve sampling: for example, in ray tracing, samples are generated by casting light rays into the 3D scene.

However, the sampling process is not perfect. The most obvious problem is illustrated when a polygon or checkerboard is sampled and displayed as shown in Figure 9.3. Notice that the edge of a polygon is not perfectly straight, but instead is approximated by a staircased pattern of pixels. The resulting image has *jaggies.*

Another interesting experiment is to sample a zone plate as shown in Figure 9.4. Zone plates are commonly used in optics. They consist of a series of concentric rings; as the rings move outward radially from their center, they become thinner and more closely spaced. Mathematically, we can describe the ideal image of a zone plate by the simple formula: $\sin r^2 = \sin(x^2 + y^2)$. If we sample the zone plate (to sample an image given by a formula $f(x, y)$ at a point is very easy; we simply plug in the coordinates of the point into the function $f$), rather than see a single set of concentric rings, we see several superimposed sets of rings. These superimposed sets of rings beat against one another to form a striking *Moire pattern.*

These examples lead to some more questions: What causes annoying artifacts such as jaggies and moire patterns? How can they be prevented?

## Digital Signal Processing

The theory of signal processing answers the questions posed above. In particular, it describes how to sample and reconstruct images in the best possible ways and how to avoid artifacts dues to sampling.

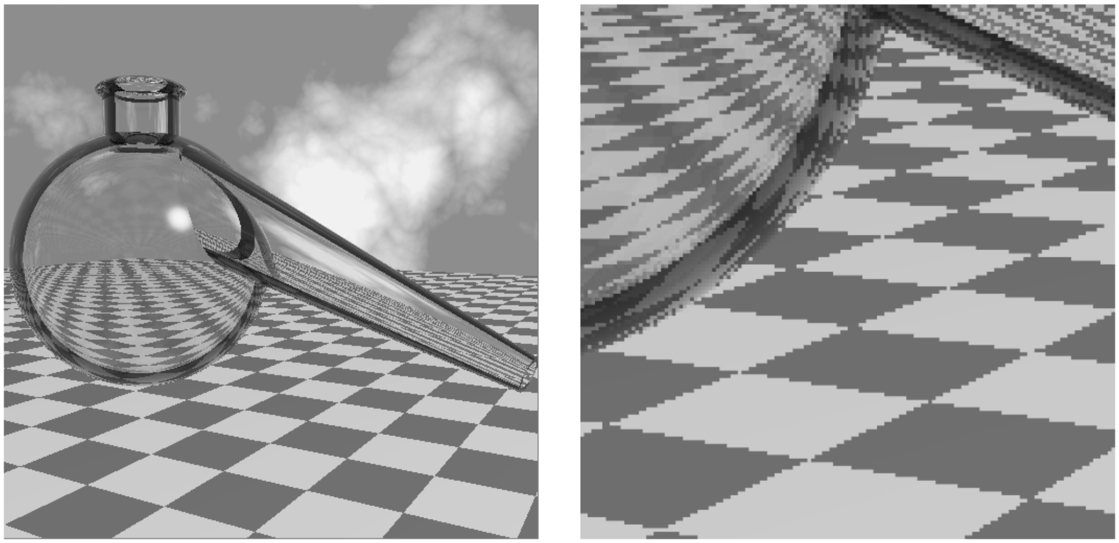


Figure 9.3: A ray traced image of a 3D scene. The image is shown at full resolution on the left and magnified on the right. Note the jagged edges along the edges of the checkered pattern.

Signal processing is very useful tool in computer graphics and image processing. There are many other applications of signal processing ideas, for example:

1. Images can be *filtered* to improve their appearance. Sometimes an image has been blurred while it was acquired (for example, if the camera was moving) and it can be sharpened to look less blurry.

2. Multiple signals (or images) can be cleverly combined into a single signal, so that the different components can later be extracted from the single signal. This is important in television, where different color images are combined to form a single signal which is broadcast.

## Frequency Domain vs. Spatial Domain

The key to understanding signal processing is to learn to think in the *frequency domain*.

Let's begin with a mathematical fact: Any periodic function (except various monstrosities that will not concern us) can always be written as a sum of sine and cosine waves.

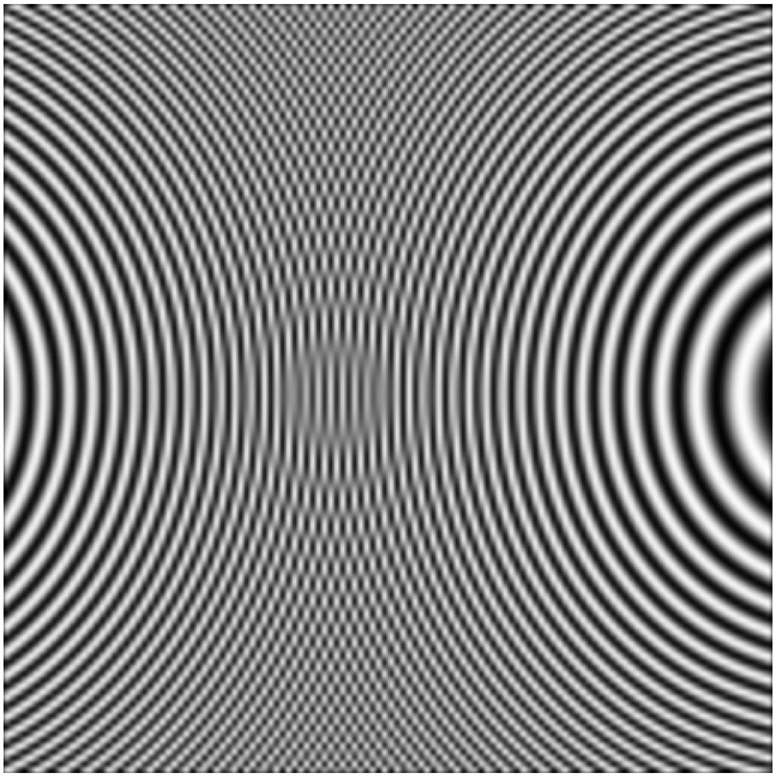


Figure 9.4: Sampling the equation $\sin(x^2 + y^2)$. Rather than a single set of rings centered at the origin, notice there are several sets of superimposed rings beating against each other to form a pronounced Moire pattern.

A periodic function is a function defined in an interval $T$ that repeats itself outside the interval. The sine function—$\sin x$—is perhaps the simplest periodic function and has an interval equal to $2\pi$. It is easy to see that the sine function is periodic since $\sin(x + 2\pi) = \sin x$. Sines can have other frequencies, for example, the sine function $\sin 2\pi f x$ repeats itself $f$ times in the interval from $0$ to $2\pi$. $f$ is the frequency of the sine function and is measured in cycles per second or Hertz.

If we could represent a periodic function with a sum of sine waves each of whose periods were harmonics of the period of the original function, then the resulting sum will also be periodic (since all the sines are periodic). The above mathematical fact says that such a sum can always be found. The reason we can represent all periodic functions is that we are free to choose the coefficients of each sine of a different frequency, and that we can use an infinite number of higher and higher frequency sine waves.

As an example, consider a rather nasty function—a square pulse. This function is nasty because it is discontinuous in value and derivative at the beginning and ending points of the pulse. A square pulse is the limit as $n \to \infty$ of

$$\begin{aligned} S_n(x) &= \frac{1}{2} + \frac{2}{\pi} \sum_{k=1}^{n} (-1)^{k-1} \frac{\cos(2k-1)\omega x}{2k-1} \\ &= \frac{1}{2} + \frac{2}{\pi} (\cos \omega x - \frac{1}{3} \cos 3\omega x + \frac{1}{5} \cos 5\omega x + \ldots) \end{aligned}$$

Where the angular frequency (in radians) $\omega = 2\pi f$. A plot of this formula for four different values of $n$ is shown in Figure 9.5. Notice that as $n$ increases, the sum of sines more closely approximates the ideal square pulse.

More generally, a non-periodic function can also be represented as a sum of sin's and cos's, but now we must use all frequencies, not just multiples of the period. This means the sum is replaced by an integral.

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega x} d\omega$$

where $e^{i\omega x} = \cos \omega x + i \sin \omega x$ ($i = \sqrt{-1}$). $F(\omega)$ are the coefficients of each sine and cosine; $F(\omega)$ is called the spectrum of the function $f(x)$.

The spectrum can be computed from a signal using the Fourier transform.

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

Unfortunately, we do not have time to derive these formulas; the reader will have to accept them as true. For those interested in their derivation, we refer you to Bracewell.

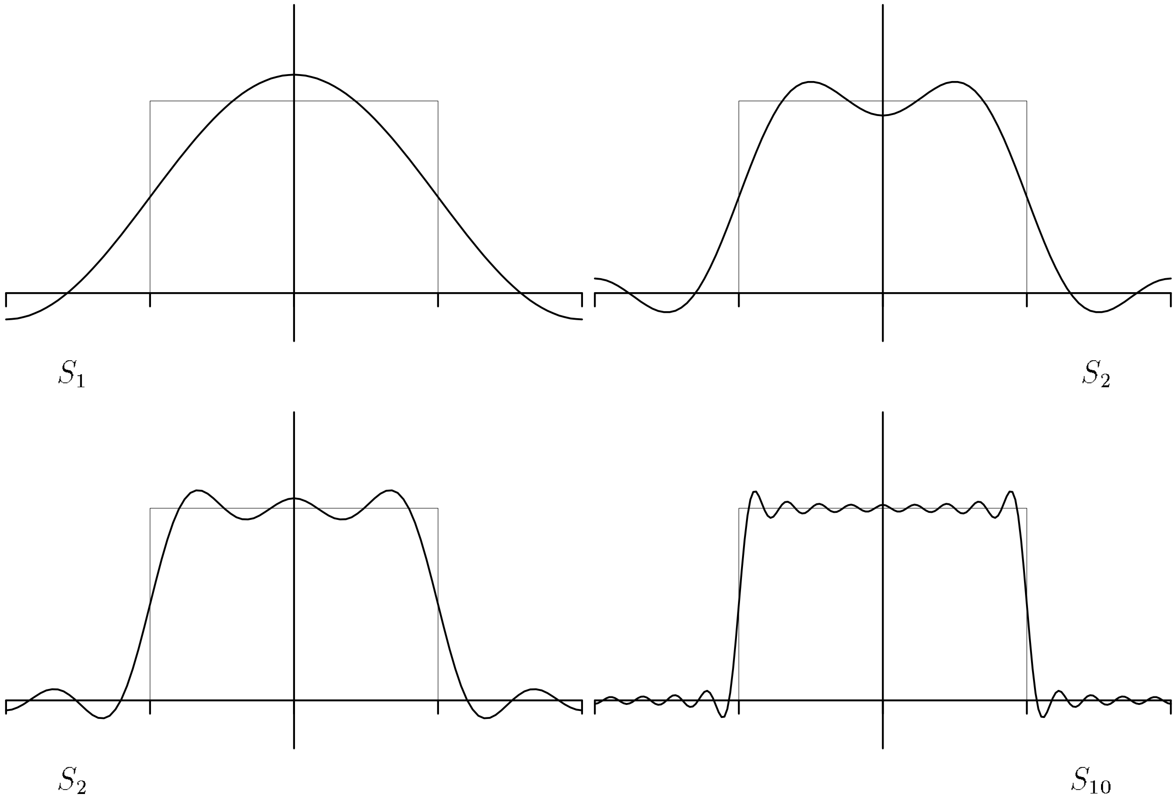
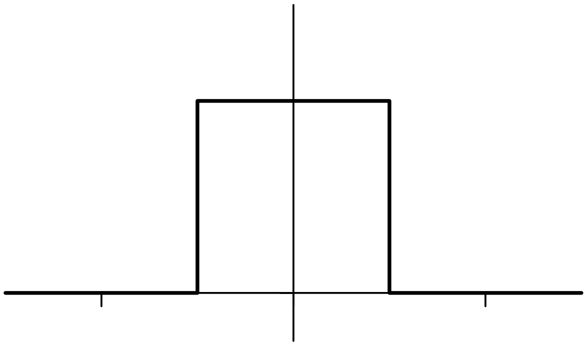


Figure 9.5: Four approximations to a square pulse. Notice that each approximation involves higher frequency terms and the resulting sum more closely approximates the pulse. As more and more high frequencies are added, the sum converges exactly to the square pulse. Note also the oscillation at the edge of the pulse; this is often referred to as the Gibbs phenomena.

To illustrate the mathematics of the Fourier transform, let us calculate the Fourier transform of a square pulse. A square pulse is described mathematically as

$$\mathrm{square}(x) = \begin{cases} 1 & |x| \le \frac{1}{2} \\ 0 & |x| > \frac{1}{2} \end{cases}$$



The Fourier transform of this function is straightforward to compute.

$$\begin{aligned} \int_{-\infty}^{\infty} \mathrm{square}(x)\, e^{-i\omega x}\, dx &= \int_{-\frac{1}{2}}^{\frac{1}{2}} e^{-i\omega x}\, dx \\ &= \frac{e^{-i\omega x}}{-i\omega} \Big|_{-\frac{1}{2}}^{\frac{1}{2}} \\ &= \frac{e^{i\frac{1}{2}\omega} - e^{-i\frac{1}{2}\omega}}{2i\,\frac{1}{2}\omega} \\ &= \frac{\sin \frac{1}{2}\omega}{\frac{1}{2}\omega} \\ &= \mathrm{sinc} f \end{aligned}$$

Here we introduce the $\mathrm{sinc}$ function defined to be

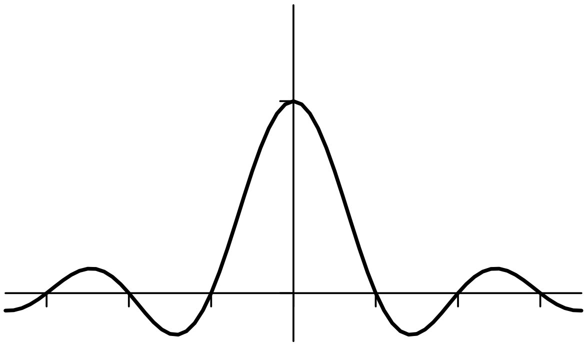
$$\mathrm{sinc}\, x = \frac{\sin \pi x}{\pi x}$$

Note that $\sin \pi x$ equals zero for all integer values of $x$, except $x$ equals zero. At zero, the situation is more complicated: both the numerator and the denominator are zero. However, careful analysis shows that $\mathrm{sinc}\, 0 = 1$.

Thus,

$$\mathrm{sinc}(n) = \begin{cases} 1 & n = 0 \\ 0 & n \ne 0 \end{cases}$$

A plot of the sinc function is shown below. Notice that the amplitude of the oscillation decreases as $x$ moves from the origin.



It is important to build up your intuition about functions and their spectra. Figure 9.6 shows some example functions and their Fourier transforms.

The Fourier transform of $\cos\,\omega x$ is two spikes, one at $-\omega$ and the other at $+\omega$. This should be intuitively true because the Fourier transform of a function is an expansion of the function in terms of sines and cosines. But, expanding either a single sine or a single cosine in terms of sines and cosines yields the original sine or cosine. Note, however, that the Fourier transform of a cosine is two positive spikes, whereas Fourier transform of a sine is one negative and one positive spike. This follows from the property that the cosine is an even function ($\cos -\omega t = \cos \omega t$) whereas the sine is an odd function ($\sin -\omega t = -\sin \omega t$).

The Fourier transform of a constant function is a single spike at the origin. Once again this should be intuitively true. A constant function does not vary in time or space, and hence, does not contain sines or cosines with non-zero frequencies.

Comparing the formula for the Fourier transform with the formula for the inverse Fourier transform, we see that they differ only in the sign of the argument to the exponential. This implies that Fourier transform and the inverse Fourier transform are qualitatively the same. Thus, if we know the transform from the space domain to the frequency domain, we also know the transform from the frequency domain to the space domain. Thus, the Fourier transform of a single spike at the origin consists of sines and cosines of all frequencies equally weighted.

In the above discussion we have used the term *spike* several times without properly defining it. A delta function has the property that it is zero everywhere except at the origin.

$$\delta(x) = 0 \qquad x \neq 0$$

The value of the delta function is not really defined, but its integral is. That is,

$$\int_{\infty}^{\infty} \delta(x)dx = 1$$

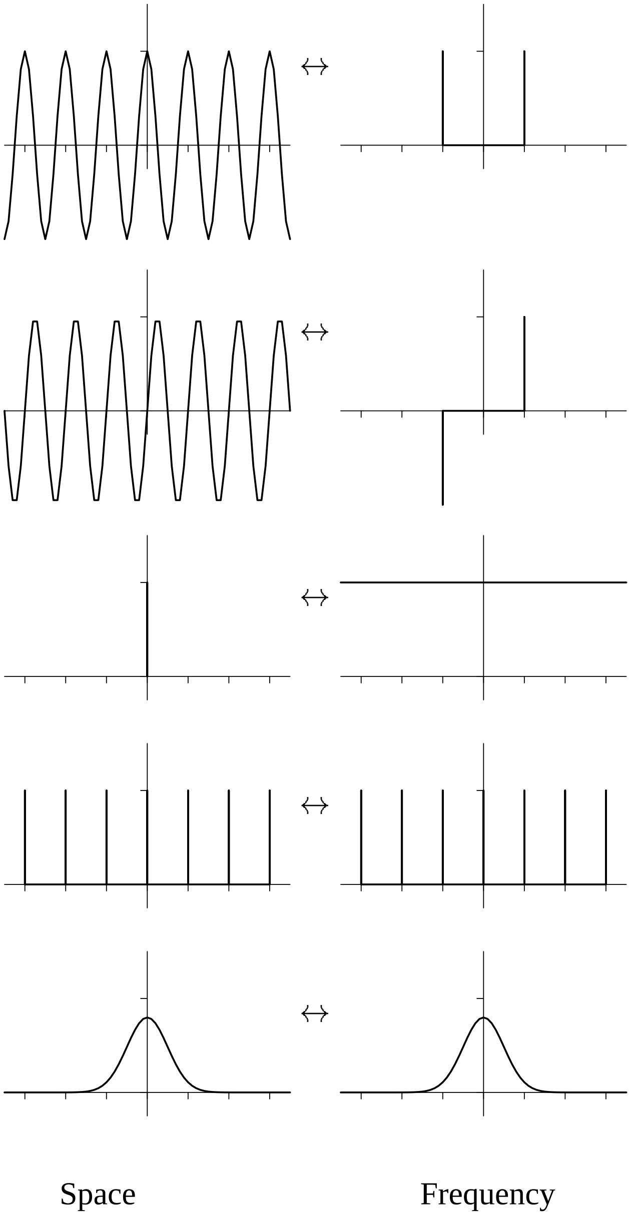


Figure 9.6: Fourier transform pairs. From top to bottom: $\cos \omega x$, $\sin \omega x$, $\delta(x)$, $\mathrm{shah}(x)$, and $e^{-x^2}$.

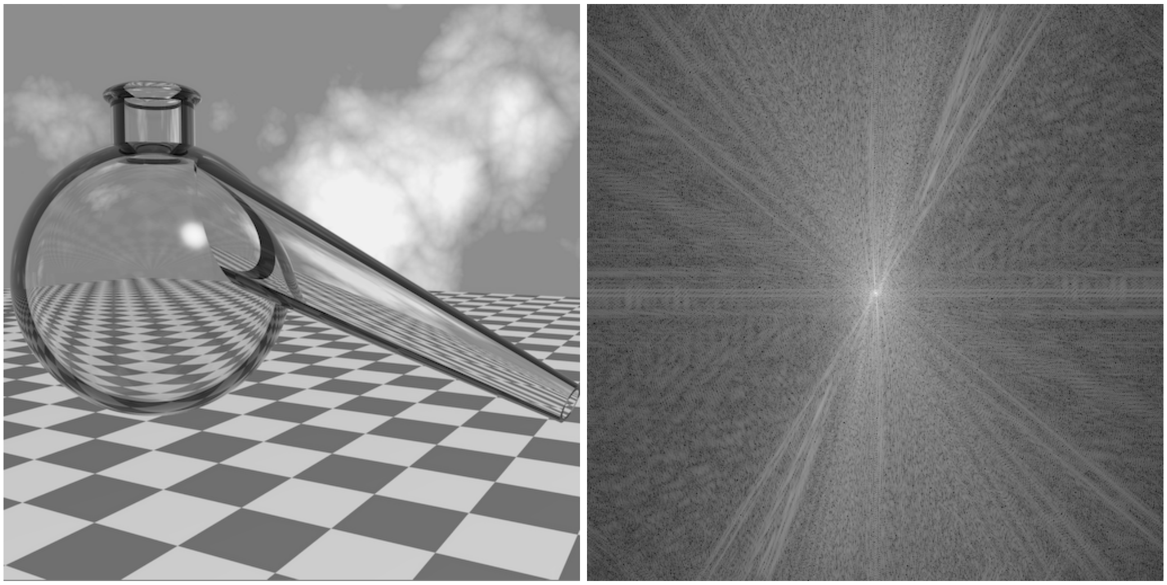


Figure 9.7: An image and its Fourier transform

One imagines a delta function to be a square pulse of unit area in the limit as the base of the pulse becomes narrower and narrower and goes towards zero.

The example Fourier transform pairs also illustrate two other functions. The Fourier transform of a sequence of spikes consist of a sequence of spikes (a sequence of spikes is sometimes referred to as the *shah* function). This will be very useful when discussing sampling.

It also turns out that the Fourier transform of a Gaussian is equal to a Gaussian.

The spectrum of a function tells the relative amounts of high and low frequencies in the function. Rapid changes imply high frequencies, gradual changes imply low frequencies. The zero frequency component, or dc term, is the average value of the function.

The above ideas apply equally to images. Figure 9.7 shows the ray traced picture and its Fourier transform. In an image, high frequency components contribute to fine detail, sharp edges, etc. and low frequency components represent large objects or regions.

Another important concept is that of a *bandlimited function*. A function is bandlimited if its spectrum has no frequencies above some maximum frequency. Said another way, the spectra of a bandlimited function occupies a finite interval of frequencies, not the entire frequency range.

To summarize: the key point of this section is that a function $f$ can be easily converted from the space domain (that is, a function of $x$) to the frequency domain

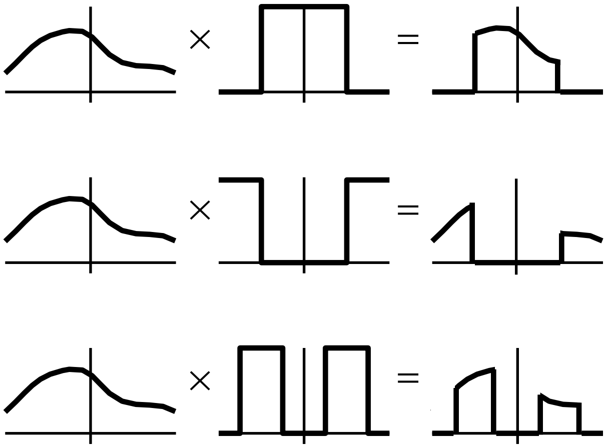


Figure 9.8: Perfect low-pass (top), high-pass (middle), and band-pass (bottom) filters.

(that is, a function, albeit a different function, of $\omega$), and vice versa. Thus, a function can be interpreted in either of two domains: the space or the frequency domain. The Fourier transform and the inverse Fourier transform can be used to interconvert between the two domains. Some properties and operations on functions are easier to see in the space domain, others are easier to see in the frequency domain.

## Convolution and Filtering

The spectrum of a function can be modified to attenuate or enhance different frequencies. Modifying a signal or an image in this way is called *filtering*. Mathematically, the properties of filters are easiest to describe in the frequency domain.

$$H(\omega) = F(\omega) \times G(\omega)$$

Here, $H$ is the spectrum of the filtered function, $F$ is the spectrum of the original function, and $G$ is the spectrum of the filter. The symbol $\times$ indicates simple multiplication. Each frequency component of the input function is multiplied by the corresponding frequency component of the filter function to compute the value of the output function at that frequency component.

The effects of filters are shown in Figure 9.8. Filters are characterized by how they change different frequency components. A *low-pass filter* attenuates high frequencies relative to low frequencies; a *high-pass filter* attenuates low frequencies relative to high frequencies; a *band-pass filter* preserves a range of frequencies relative to those outside that range. A perfect low-pass filter leaves all frequencies below

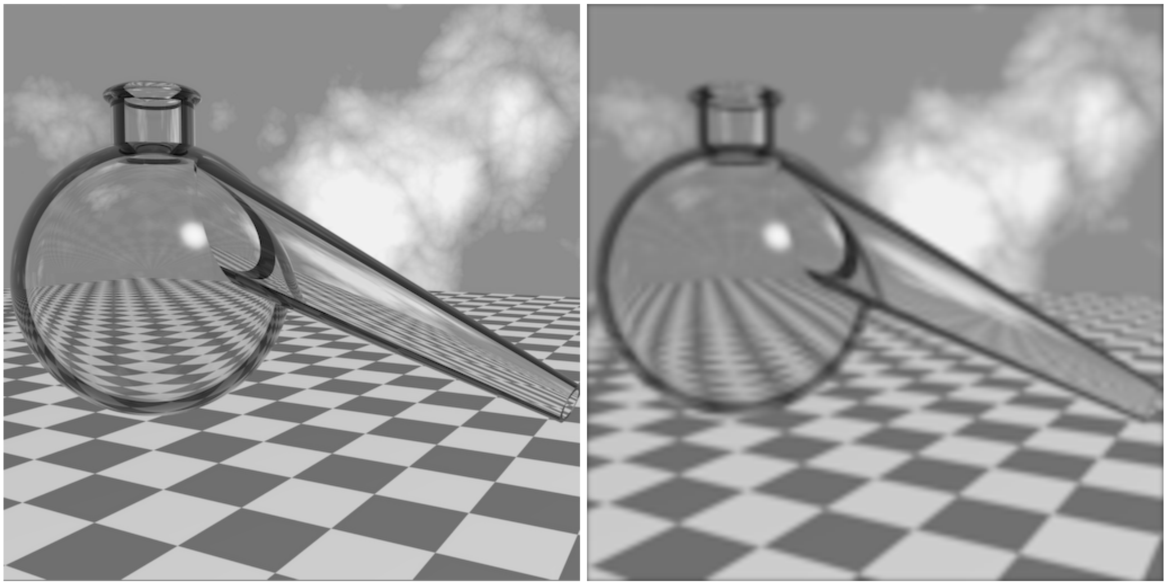


Figure 9.9: Application of a low-pass filter to an image. Notice that the resulting image on the right is blurry. This is because the filter removes all the high frequencies which represent fine detail.

it *cut-off* frequency and removes all frequencies above the cut-off frequency. Thus, in the frequency domain, a low-pass filter is a square pulse (see Figure 9.8). Similarly, a perfect high-pass filter completely removes all frequencies below the cut-off frequency, and a perfect band-pass filter removes all frequencies outside its band.

When an image is filtered, the effect is very noticeable. Removing high frequencies leaves a blurry image (see Figure 9.9). Removing low frequencies enhances the high frequencies and creates a sharper image containing mostly edges and other rapidly changing textures (see Figure 9.10). The cutoff frequency for the high pass and the low pass filter is the same in the examples shown in Figure 9.9 and Figure 9.10. Since the sum of the low and high pass filters is 1, the sum of the filtered pictures must equal the original picture. Therefore, if the pictures in the right hand column are added together, the original picture in the left hand column is returned.

The properties of filters are easiest to see in the frequency domain. However, it is important to be able to apply filters in either the frequency domain or the space domain. In the frequency domain, filtering is achieved by simply multplying spectra, value by value. In the space domain, filtering is achieved by a more complicated operation called *convolution*.

$$h(y) = f \otimes g = \int_{-\infty}^{\infty} f(x)g(y-x)dx$$

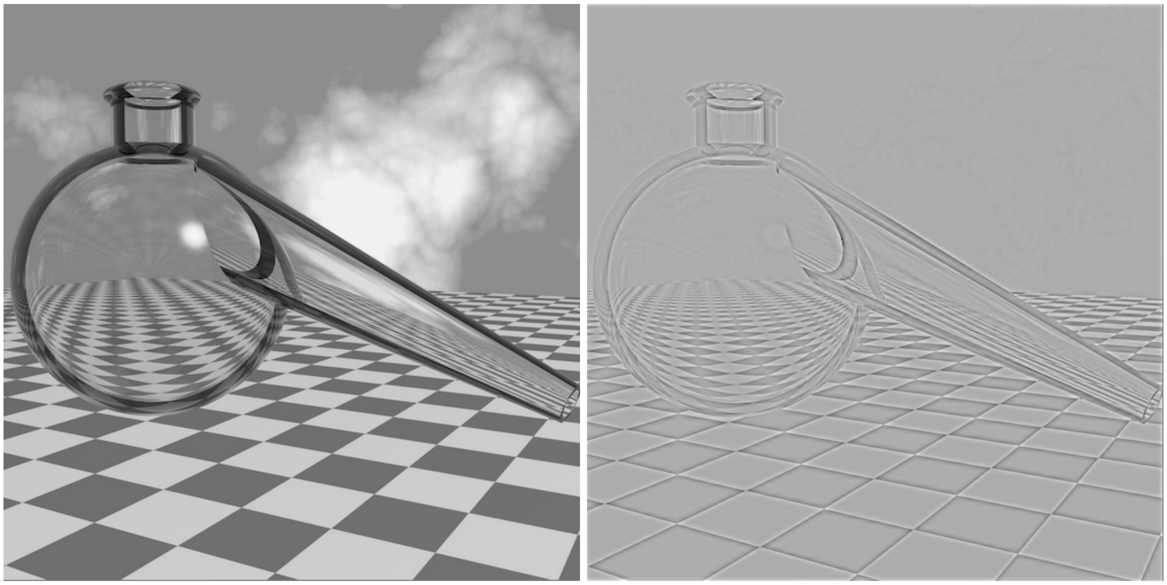


Figure 9.10: Application of a high-pass filter to an image. Notice that in the resulting image the low frequencies have been removed and only places in the image that are changing, such as edges, remain.

Where the binary operator $\otimes$ represents convolution.

There are two different ways to conceptualize convolution:

**Forward:** Slide the filter $g$ along the axis $x$ defining the input function $f$. At each position $x$ , multiply the function $g$ by the value $f(x)$. The scaled and translated function $f(x)g(y-x)$ is then accumulated into $h$ and the process proceeds by sliding $g$ to the next position.

In this view of convolution, the outermost loop iterates over different input values. It is sometimes referred to as forward convolution because each single input value $f(x)$ maps forward to several output values $h(y+x)$.

**Backward:** Slide the filter g along the axis $y$ defining the output function $h$. Now, for each value of $g$, multiply it by the corresponding value of $f$ and sum the results. The sum is then written as $h(y)$.

In this view of convolution, the outermost loop iterates over different output values. It is sometimes referred to as backward convolution because each single output value $f(x)$ is computed by mapping backwards into several input values $f$.

To illustrate convolution, suppose the input function consists of a single spike at the origin. In the forward view of convolution, we center the filter function at each

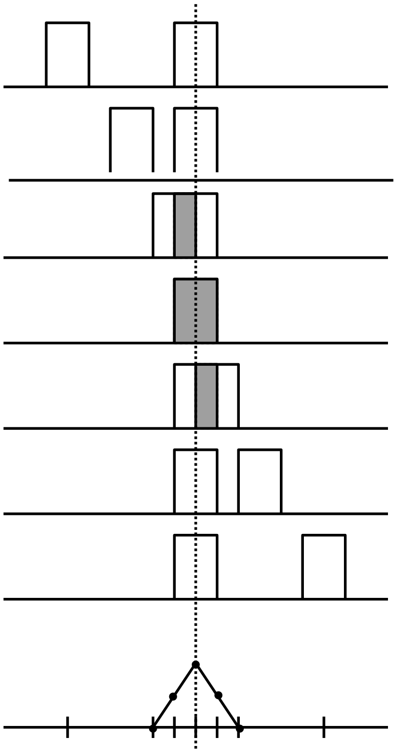


Figure 9.11: Convolution of two square pulses

point along the input function and multiply the filter everywhere by the value of the function. If the input is a single spike at the origin, then the input function is zero everywhere except at zero. Thus, the filter is multiplied by zero everywhere except at the origin where it is multipled by one. Therefore, the result of convolving the filter by a delta function is the filter itself. Mathematically, this follows immediately from the definition of the delta function.

$$\int_{-\infty}^{\infty} \delta(x) g(y-x) dx = g(y)$$

Many physical processes may be described as filters. If such a process is driven a delta function, or impulse, the output will the be characteristic filtering function of the system. For this reason, filters are sometimes referred to as *impulse response functions*.

Convolution is a very important idea so let us consider another example—the convolution of two square pulses as shown in Figure 9.11. The figure shows the convolution as a backward mapping. One square pulse, the one corresponding to the input signal, is shown stationary and centered at the origin. The other square pulse, representing the filter, moves along the output axis from left to right. Each

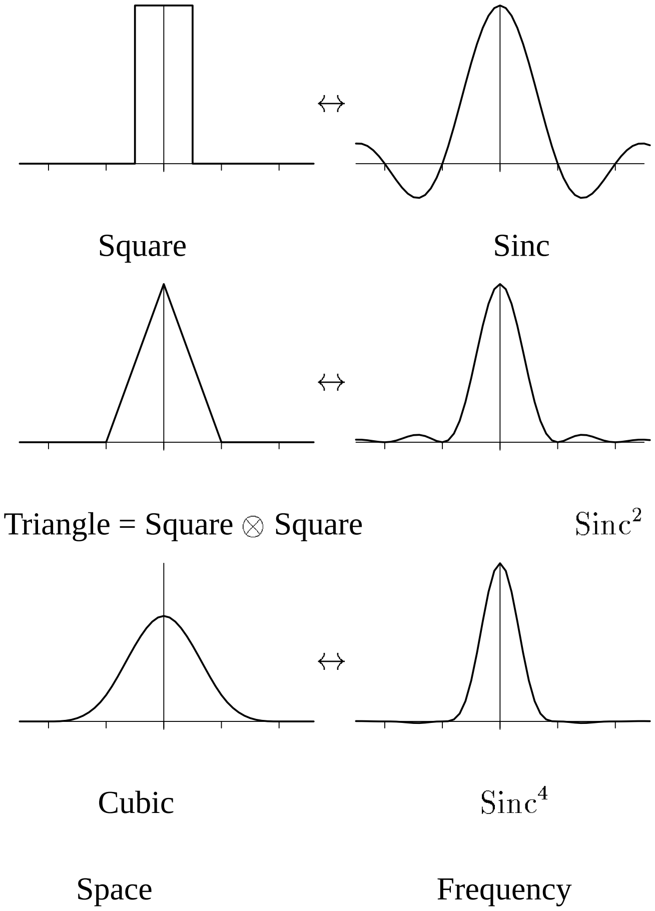


Figure 9.12: The results of convolving a square pulse with itself multiple times.

output value is the sum of the product of the filter and the input. In the case of two pulses, this equals the area of overlap of the two square pulses. This area starts out zero when the pulese are disjoint, begins to increase linearly when they first touch, reaches a maximum when they are superimposed, and then begins to decrease until they are just touching, after which it returns to zero. The result is a triangle or tent function.

Convolving a function or an image with a square pulse is an interesting operation. First, notice that this can be interpreted as setting the output to the average of the input function over the area where the pulse is non-zero. Make sure that you are convinced of this! Second, recall that the Fourier transform of a square pulse is a sinc function. Referring to Figure 9.12, notice that the sinc function goes to zero at higher frequencies. Thus, a sinc function is a low-pass filter. This property should be intuitively true, since averaging an input image over a region should blur it and remove high frequencies.

What is the spectrum of the function resulting from convolving two square pulses? Convolving two functions corresponds to multiplying their spectra, therefore, convolving a square pulse with a square pulse corresponds to the multiplication of two sinc functions. Similarly, the convolution of $n$ pulses corresponds to the sinc raised the the n'th power. The function produced by convolving a pulse with itself $n$ times is called a *B-spline*. We will encounter B-splines again when discussing methods for representing curves and surface. Another interesting fact is that in the limit as $n$ goes to infinity the convolution of $n$ pulses approaches a Gauusian,

The *convolution theorem* states that multiplying two spectra in the frequency domain corresponds to convolving the functions in the space domain.

$$f \otimes g \leftrightarrow F \times G$$

Because the Fourier transform and the inverse Fourier transform are so similar, a symmetric interpretation is also true. That is, multiplying two functions in the space domain corresponds to convolving the functions in the frequency domain.

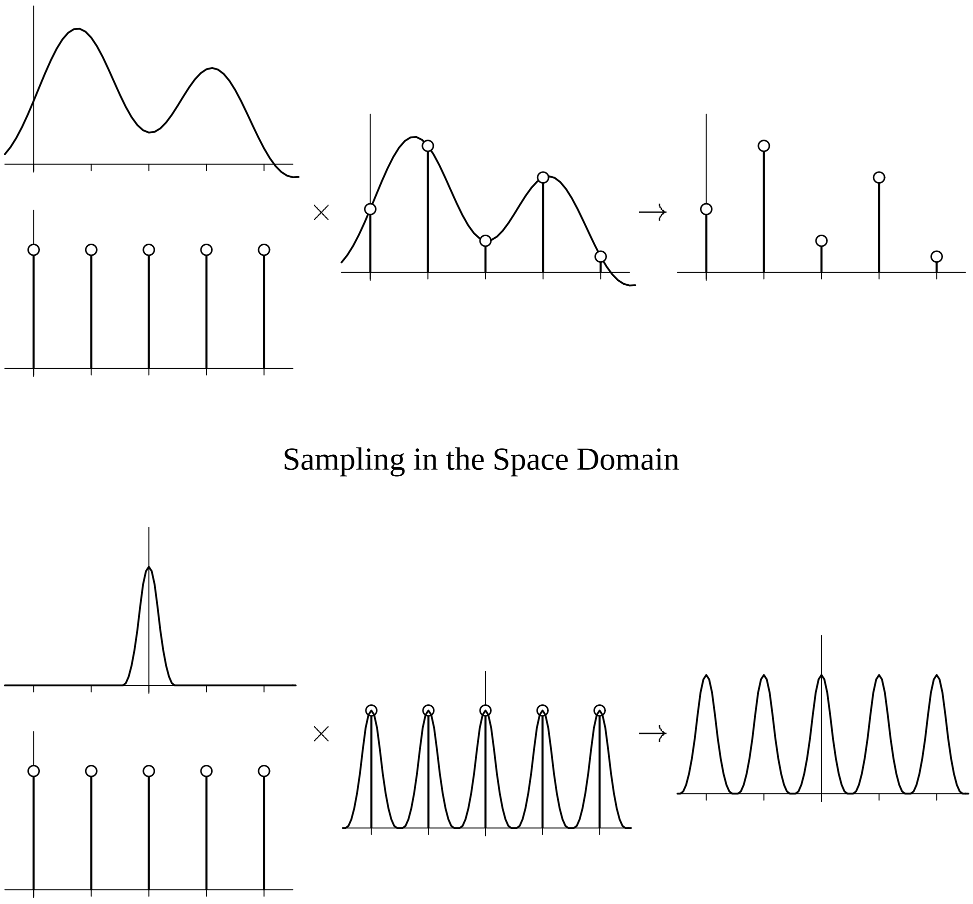
$$f \times g \leftrightarrow F \otimes G$$

## Sampling and Reconstruction

With this background on frequency space and convolution, we can now analyze the processes of sampling and reconstruction.

In the space domain, sampling can be viewed simply as multiplying the signal by sequence of spikes with unit area. Since the spikes are zero everywhere except at integer values, this has the result of throwing away all the information except at the sample points. At the sample points, the result is the value of the function at that point. This view of sampling in the space domain is illustrated in the top half of Figure 9.13.

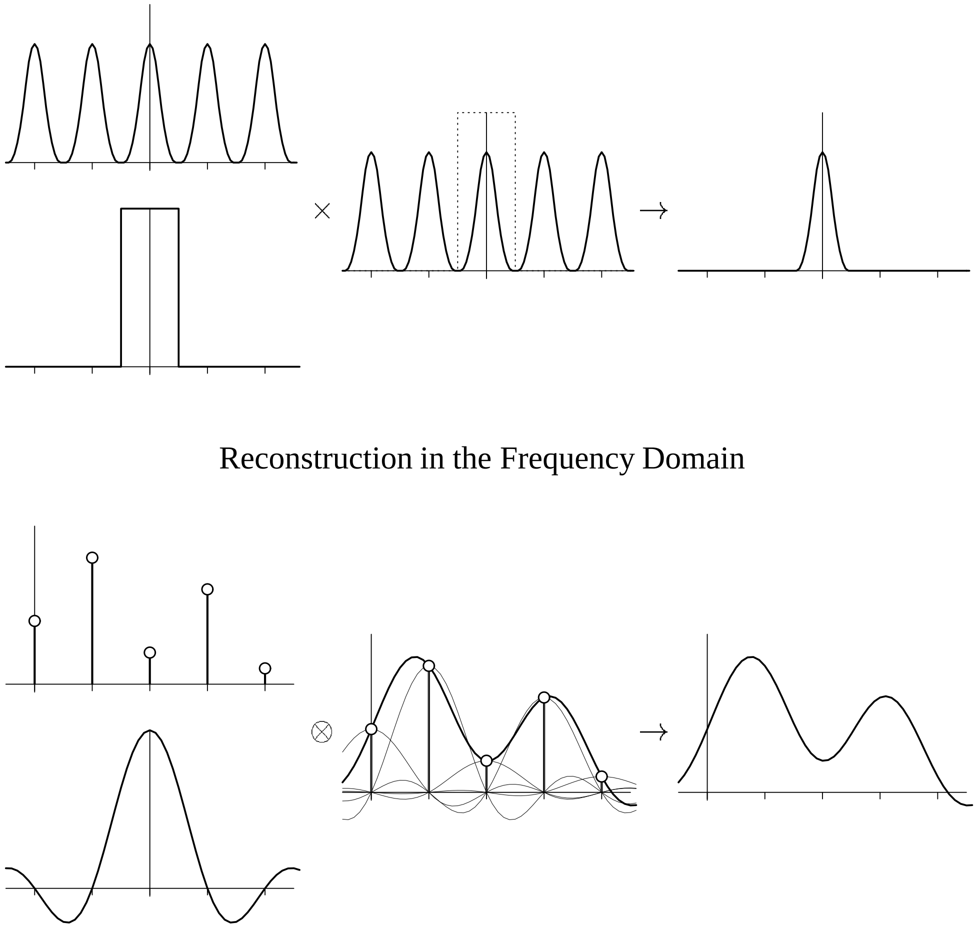
Additional insight into the sampling process, however, can be gained by considering sampling in the frequency domain. Recall the convolution theorem. This theorem states that multiplying two signals in one domain (in this case, the space domain) corresponds to convolving the signals in the other domain (the frequency domain). Thus, multiplying the function by a sequence of spikes in the space domain corresponds to convolving the spectrum of the original function with the spectrum of a sequence of spikes. However, recall that the Fourier transform of a sequence of spikes is itself a sequence of spikes. Thus, in the frequency domain, sampling corresponds to convolving the spectrum of the function with a sequence of spikes. Convolving with a sequence of spikes causes the original function to be replicated—a



Sampling in the Space Domain

Sampling in the Frequency Domain

Figure 9.13: Sampling



Reconstruction in the Frequency Domain

Reconstruction in the Space Domain

Figure 9.14: Reconstruction

new copy of the spectrum is centered at a spike. The view of sampling in the frequency domain is illustrated in the bottom half of Figure 9.13.

Now let us consider the reconstruction process. The process of recovering the original signal from the sampled signal is easiest to analyze in the frequency domain. Remember, the sampling process resulted in the replication of the spectrum of the original function. If these replicas do not overlap, then the original can be recovered by the application of a perfect low-pass filter. Multiplying the replicated spectrum by a square pulse centered on the original signals spectrum will remove all the extra copies of the spectrum. This is illustrated in Figure 9.14.

Of course, for every process in one domain we can create a dual process in the

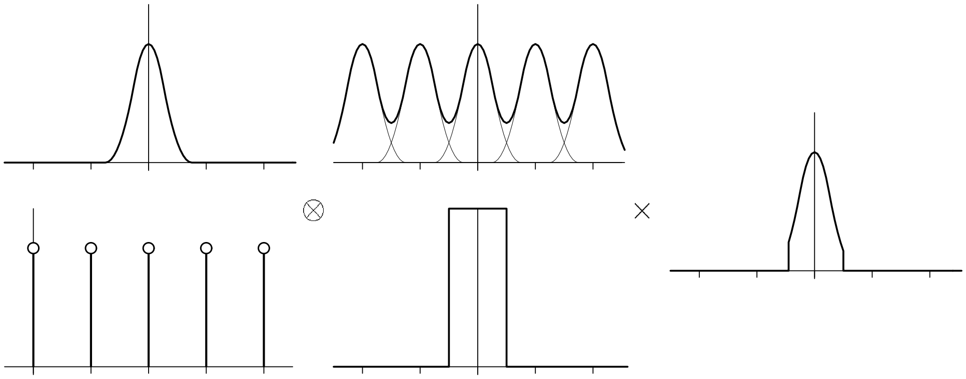


Figure 9.15: Undersampling a function results in aliasing.

other domain. In this case, multiplying the replicated spectrum by a square pulse in the frequency domain corresponds to convolving the samples with a sinc function (the Fourier transform of the square pulse) in the spatial domain. The sinc function interpolates the samples, and therefore reconstructs the continuous image from the set of samples. This is illustrated in Figure 9.14.

Note that a miraculous thing has happened: The result of the sampling and reconstruction process is the original function. That is, no information was lost in sampling process. This result is known as the *Sampling Theorem* and is due to Claude Shannon who first discovered it in 1949.

> A signal can be reconstructed from its samples without loss of information, if the original signal has no frequencies above $\frac{1}{2}$ the sampling frequency.

For a given bandlimited function, the rate at which it must be sampled is called the *Nyquist Frequency.*

## Aliasing: Pre- and Post-

There are two reasons the above sampling and reconstruction process may not work out.

First, when a function is sampled, the replicas of the function's spectrum may overlap. In fact, this will occur for any function that is not bandlimited, or for any function which is sampled at less than its Nyquist frequency. When overlap occurs there is no hope of recovering the original function.

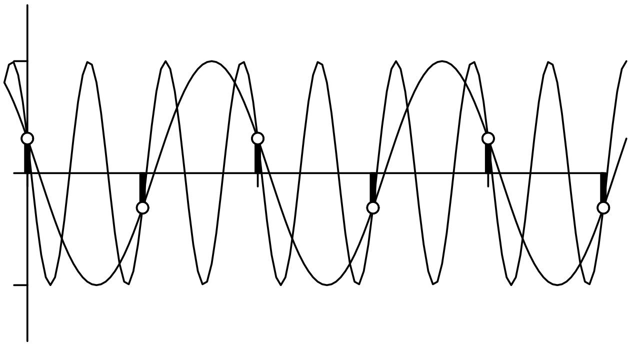


Figure 9.16: Sampling a sine wave. Sampling the function $\sin 1.5\omega x$ yields the same values as sampling the function $\sin 0.5\omega x$. Thus, the higher frequency $1.5\omega$ which is above the Nyquist frequency, cannot be distinguished from the lower frequency $0.5\omega$.

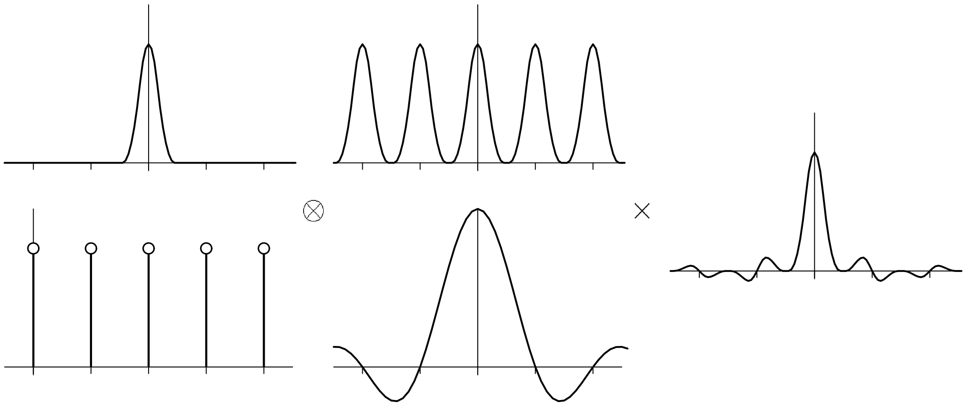


Figure 9.17: Poor Reconstruction results in aliasing.

If copies of the spectra overlap, then some frequencies will appear as other frequencies. In particular, high frequencies will foldover and appear as low frequencies. This sudden appearance of some frequencies at other frequencies is referred to as *aliasing*. The result of the foldover is that the reconstruction process can not differentiate between the original spectrum and the aliased spectrum, and, hence, the function cannot be perfectly reconstructed. This effect is shown in Figure 9.15.

To illustrate aliasing consider the following thought experiment. Consider a sine wave with a frequency of 1.5 cycles per sample. Now sample the sine wave. This sampling rate is less than the frequency of the function, and hence we may expect aliasing to result. This is seen in Figure 9.16. That figure shows that sampling $\sin{(2\pi 1.5)x}$ yields the same values as sampling $\sin(2\pi 0.5)x$.

Implicit in the sampling theorem is that the function be perfectly reconstructed. Unfortunately, this is often not possible in practice. For one, the perfect low-pass filter is a $\mathrm{sinc}$ function. However, convolving the sampled with a sinc function is impractical because the $\mathrm{sinc}$ function has infinite extent. Also, in general, reconstruction is a property of the hardware and media. For example, most displays employ a two step process. In the first step the digital value is converted to an analog value using a D/A convertor. Most D/A convertors sample the input and hold them constant until the next input is available. This corresponds to convolving the sampled signal with a square pulse. In the second step the analog voltage is converted to light using phosphors on the monitor. Most phosphors emit a small Gaussian spot of light centered at the location of the electron beam. This has the effect of convolving the signal with a Gaussian. Although the combination of these two steps is a low-pass filter, the filtering is not perfect.

This is illustrated in Figure 9.17. Suppose the function is reconstructed with a square pulse. That would correspond to multiplying its spectra times the transform of the pulse—a $\mathrm{sinc}$. However, a $\mathrm{sinc}$ does not perfectly remove all the replicas of the spectra produced by the sampling process, and so aliasing artifacts would be visible.

In both these cases, frequencies may masquerade as other frequencies. The first cause—due to undersampling—is called *pre-aliasing*; the second cause—due to bad reconstruction—-is called *post-aliasing*.