What We Need Around Here Is More Aliasing

James F. Blinn, Caltech

Isn't it weird that when you apply for something you can either "fill in a form" or "fill out a form"? If a chicken is "boned" what about one that's "deboned"? If "flammable" means that something can catch fire, what does "inflammable" mean? If you want to create something out of sheet metal you can "stamp it out." If you want to destroy something you also "stamp it out." (And some people want to program computers in English!)

Why do I bring this up? I recently attended a product demo of a display system at which the demonstrator promised "we can get rid of the jagged edges by using aliasing." (He was probably in the marketing department.)

I think he meant "antialiasing." I suppose the mere mention of the word in either a positive or negative sense means you are aware of the problem and are doing something about it. Anyway, I thought this would be a good time to give a brief tutorial on what aliasing means, show plots of some relevant functions, describe some of the conventional wisdom about aliasing, and tell why that wisdom may not be so wise.

Some basic knowledge

To understand aliasing, you need to start with three basic mathematical concepts: the Fourier transform, convolution, and the convolution theorem.

I want to do this in an intuitive way, so I will mostly state results without proving them, and I'll mostly explain things with pictures rather than equations. (Much as I love algebra, this is an area where the algebra can make things look more complicated than they really are.) You can get more information from any book on signal processing. My favorite is *Transmission and Display of Pictorial Information* by D.E. Pearson, Halsted Press, John Wiley and Sons, New York, 1975.

The Fourier transform

Loosely speaking, the Fourier transform (FT) is a way of representing a function as the sum of a bunch of sine waves of various frequencies. It's as though there were a Twilight Zone version of each function. Normal functions exist in the "spatial domain" while the FT doppelgangers exist in the "frequency domain."

There are actually two numbers to specify each frequency component: an amplitude and a phase offset (or horizontal position). These two are usually encoded as a single complex number. A complete FT plot would then require three dimensions (real and imaginary axes versus frequency) and be a bit confusing, so I will just plot the magnitudes (amplitudes).

If an input picture happens to be symmetric about the origin, the imaginary part of the FT is zero. In this case I will plot the real part of the FT, which may have negative values, instead of the magnitude.

Remember, there's no new information in a Fourier transform; it's just a different way of looking at an existing function.

Convolution

A convolution takes two input functions and generates a third. It is a sort of sliding weighted average of the first function with the second function providing the weights. Glossing over a few details, it is formed as follows: Multiply the two functions together and integrate the result. Plot this number at 0. Slide the weighting function to the right by x. Multiply and integrate again; plot this number at x. Repeat this for every x you want to plot. Look at the convolutions of Figure 1 to get a feel for this.

The convolution theorem

Convolutions are interesting because of how they interact with Fourier transforms. Suppose you multiply two functions together to get a new function. What do their FTs look like? It turns out that if you convolve the FTs of the original functions, you get the FT of the new function. Symmetrically, if you convolve two functions to get a third, you can multiply their FTs to get the FT of the third. Multiplication in the spatial domain becomes convolution in the frequency domain and vice versa. It's sort of like logarithms, in which multiplication of normal numbers becomes addition in the "logarithm domain." (You remember what logarithms are, don't you? They're what they use to make slide rules.)

What does this have to do with pictures?

Let's look at the image-making process in just one dimension, along a single 16-pixel scan line. I'll present the story in parallel with spatial functions in Figure 1 and with the FT of these functions in Figure 2.

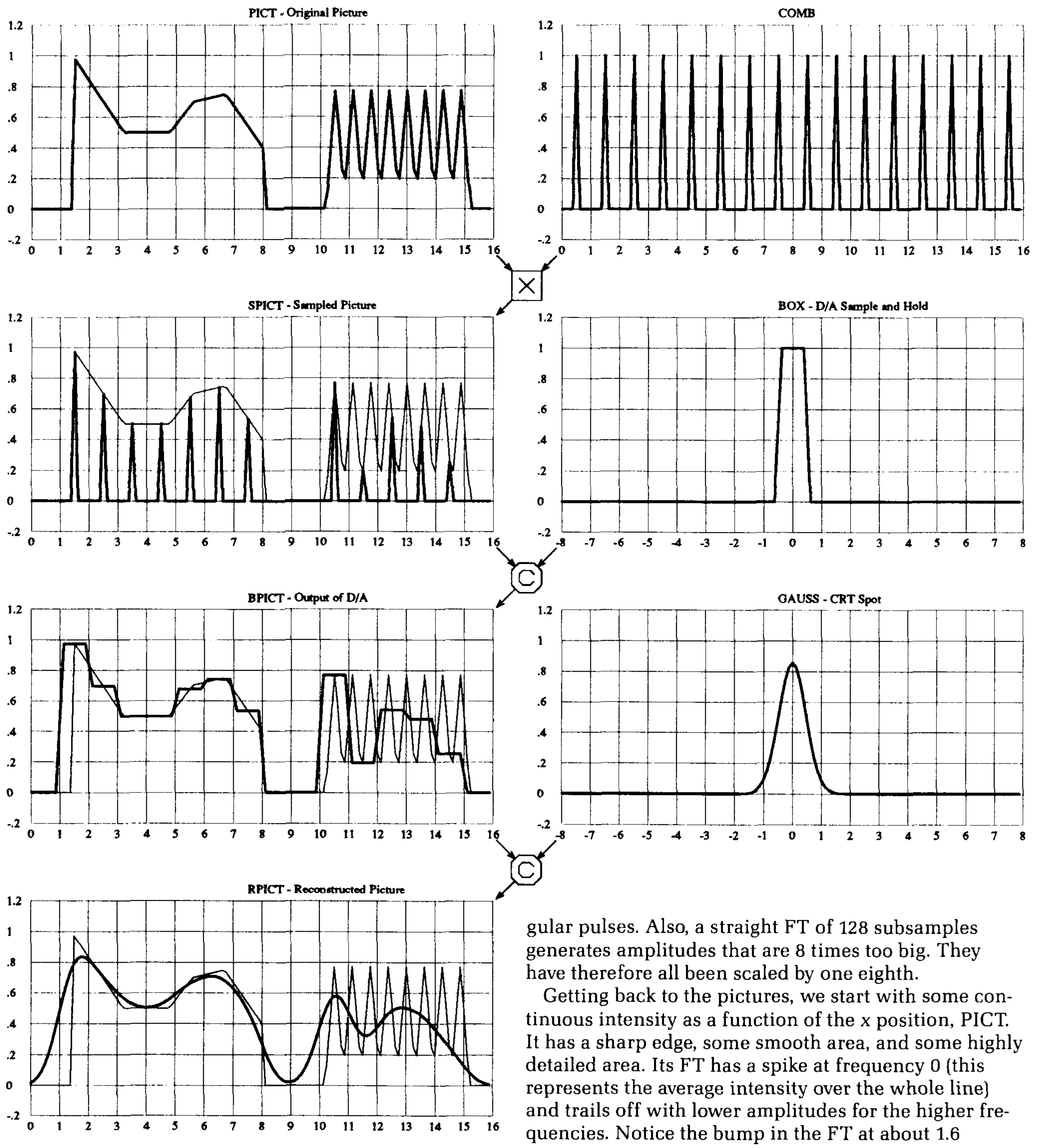

Figure 1. Spatial domain.

The horizontal scales of the FTs are in units of cycles/pixel. The little boxes down the center show how the functions are combined, "X" for multiplication and "C" for convolution.

First, a few things about the production of the pictures. To get an approximately continuous plot, I supersampled the functions by 8 times, giving a total of 128 subsamples. This is not an awful lot, so functions that are supposed to be impulses appear as tall skinny triangular pulses. Also, a straight FT of 128 subsamples generates amplitudes that are 8 times too big. They have therefore all been scaled by one eighth.

Getting back to the pictures, we start with some continuous intensity as a function of the x position, PICT. It has a sharp edge, some smooth area, and some highly detailed area. Its FT has a spike at frequency 0 (this represents the average intensity over the whole line) and trails off with lower amplitudes for the higher frequencies. Notice the bump in the FT at about 1.6 cycles/pixel that comes from the high detail region of the original image.

Next, sample the intensity at the center of each pixel. That's tantamount to multiplying PICT by the function COMB giving SPICT. Remember, multiplying spatial functions implies convolving frequency functions. The FT of COMB happens to be another comb with teeth spaced at 1 cycle/pixel. So, looking at the sampling process in the frequency domain, what we have done is convolve the original FT with a comb function. That's the same as adding together a whole lot of copies of the FT, all shifted 1 cycle/pixel apart. Since the original

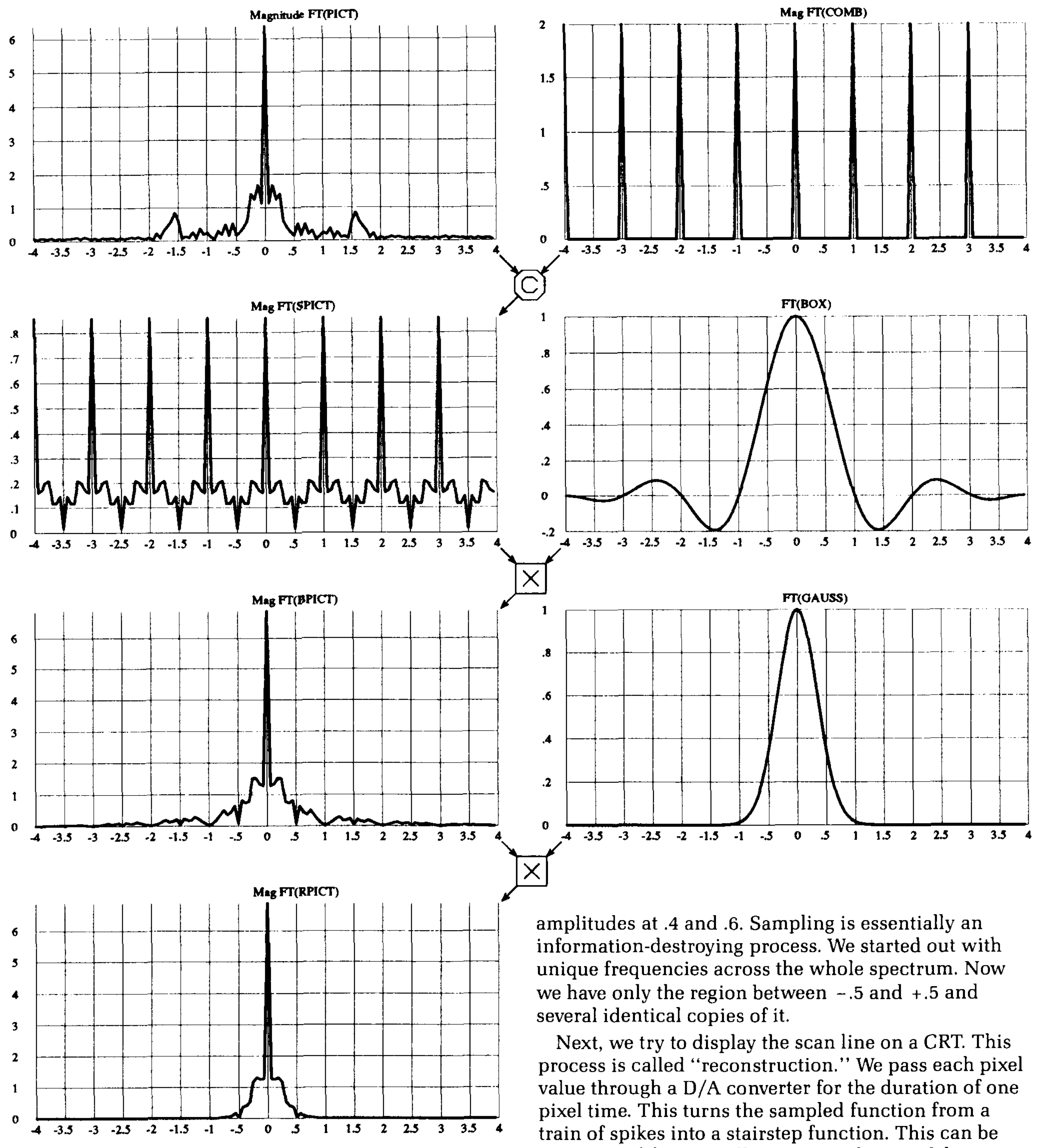

Figure 2. Frequency domain.

frequency function stretches quite far in the plus and minus directions, the copies will overlap. This is where "aliasing" comes from. The high frequencies of a shifted copy overlap onto the low frequencies of the central copy. This shows up with the 1.6 cycles/pixel bump producing the little spikes in the FT of SPICT at frequencies of .6, −.4, etc., and the −1.6 bump making spikes at −.6, +.4, etc. Once these have been added in, there is no way to tell them apart from the original amplitudes at .4 and .6. Sampling is essentially an information-destroying process. We started out with unique frequencies across the whole spectrum. Now we have only the region between −.5 and +.5 and several identical copies of it.

Next, we try to display the scan line on a CRT. This process is called "reconstruction." We pass each pixel value through a D/A converter for the duration of one pixel time. This turns the sampled function from a train of spikes into a stairstep function. This can be represented functionally as a convolution of the sampled picture with a box one pixel wide, BOX. (I have plotted the stairsteps centered at the pixels to make it easier to compare with the original picture. Really they should be shifted half a pixel to the right.) The frequency domain interpretation of this is to multiply the sampled FT with the FT of BOX. This shrinks down the extra copies of the spectrum, but there is still a bit of them left. Remember that all the stuff above .5 c/p and below −.5 c/p looks like information, but isn't. It's just mathematical debris left over from the frequency-replicating sampling process. The frequency .5 is called

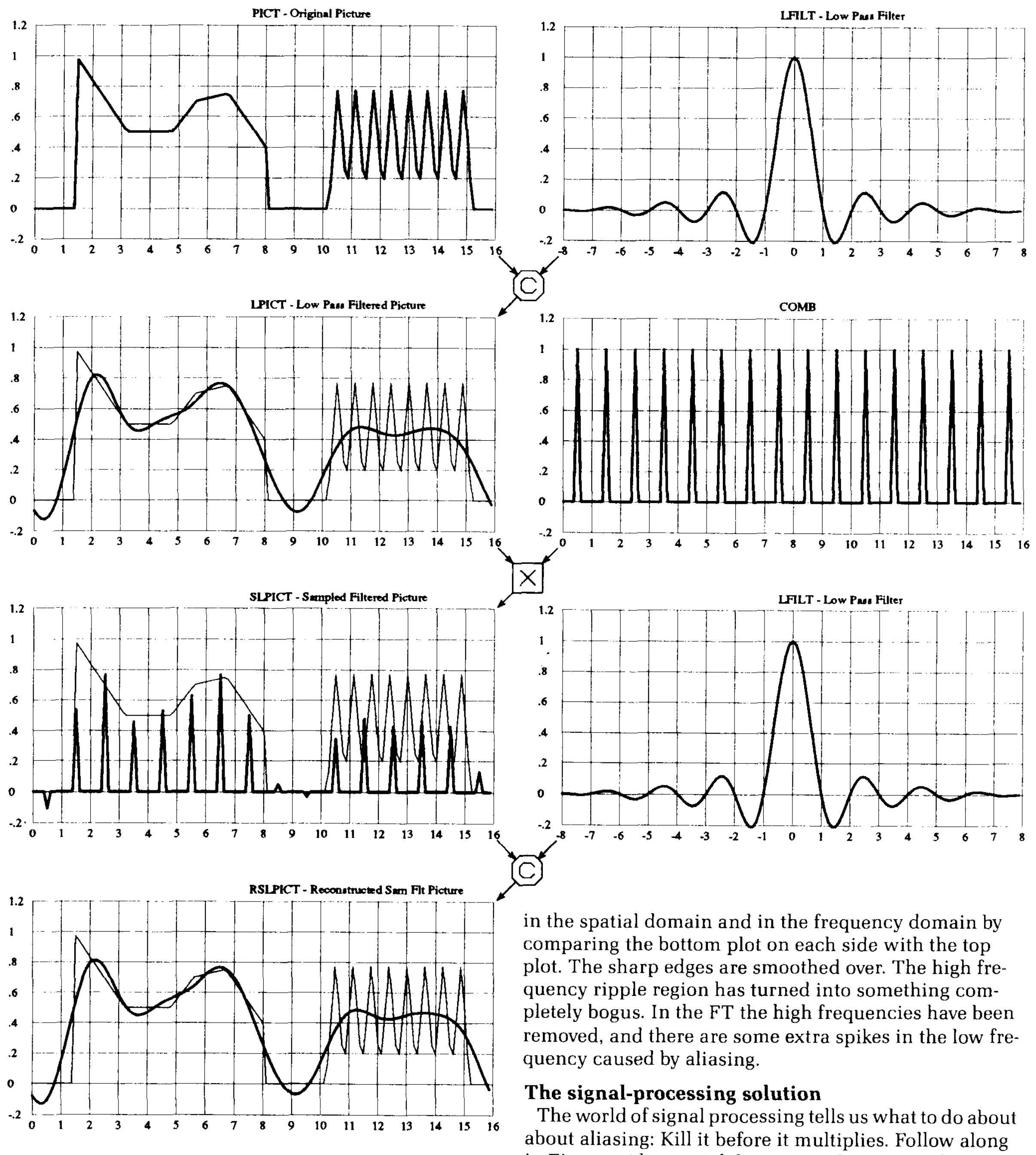

Figure 3. Spatial domain.

the Nyquist frequency.

Finally, the spot on the CRT itself has an approximately Gaussian intensity distribution. Stroking across the screen, the electron beam convolves itself with the output of the D/A converter. In the frequency domain this means multiplication by the FT of the Gaussian (another Gaussian as it happens).

You can see what damage is done to the picture both in the spatial domain and in the frequency domain by comparing the bottom plot on each side with the top plot. The sharp edges are smoothed over. The high frequency ripple region has turned into something completely bogus. In the FT the high frequencies have been removed, and there are some extra spikes in the low frequency caused by aliasing.

The signal-processing solution

The world of signal processing tells us what to do about about aliasing: Kill it before it multiplies. Follow along in Figure 3 (the spatial domain) and Figure 4 (the frequency domain).

Since we can't represent high frequencies accurately, and since they will return to haunt us disguised as low frequencies, we have to get rid of them before sampling. It's obvious how to do this in the frequency domain. Just multiply the FT of the picture by zero for frequencies beyond .5 cycles/pixel. What does this translate into in the spatial domain? Convolve the original picture with the inverse FT of a box function, the function (sin x)/x. This is interesting; we would not have thought of this particular function off the tops of

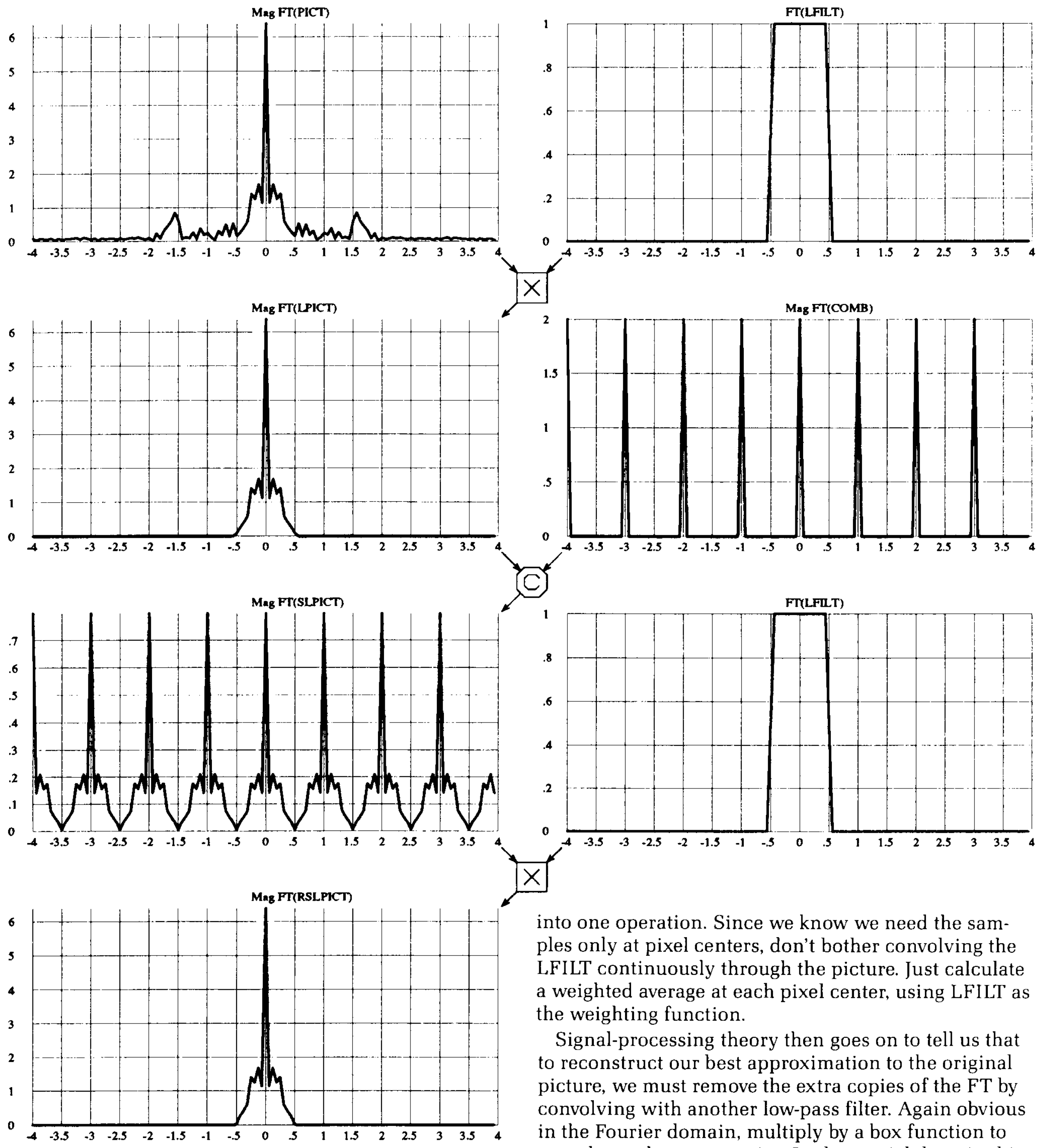

Figure 4. Frequency domain.

our heads. The result is a blurred image whose FT has no frequencies greater than .5 cycles/pixel.

The sampling process then operates as before, but now when we go to convolve the FT with a comb we have the happy consequence that the FTs do not overlap. No aliasing occurs because there are no high frequencies to alias.

Actually, the filtering and sampling can be combined into one operation. Since we know we need the samples only at pixel centers, don't bother convolving the LFILT continuously through the picture. Just calculate a weighted average at each pixel center, using LFILT as the weighting function.

Signal-processing theory then goes on to tell us that to reconstruct our best approximation to the original picture, we must remove the extra copies of the FT by convolving with another low-pass filter. Again obvious in the Fourier domain, multiply by a box function to squash out the extra copies. In the spatial domain this consists of placing copies of LFILT at each pixel, scaled by the sample value, and adding them up.

So...simple. The problem is solved. We can all go home now. Or can we?

What's wrong with this picture?

Most signal processing was invented to deal with sound or radar signals. But remember, this is computer graphics—it's not supposed to be this easy. And, as usual, the real world doesn't disappoint us in this regard. Unfortunately, I seem to be out of room, so the problems and their solution must wait for next time. ■