OVERVIEW OF IMAGE SCALING

Image scaling is important for many situations in both image processing and image analysis. Consequently, a number of different scaling functions. Binary images at high resolution (300 to 400 ppi) can be used to make low resolution binary images for analysis or display on a low resolution device (such as a monitor). To make a version of a page image that is readable on a grayscale display, the *scale-to-gray* function can be used to produce a grayscale image (of at least 4 bpp) at about 100 ppi. Scale-to-gray can also be used to convert a binary image to a very low resolution grayscale page icon thumbnail at about 15 ppi. At such resolution, a 10 x 7 inch image shrinks to 150 x 105 pixels, which appears on a 100 ppi high resolution monitor as an icon of size 1.5 x 1.05 inches.

When scaling a grayscale or RGB color image, two things must be kept in mind. First, if the image is significantly reduced, it is useful to first apply a lowpass filter whose size is roughly the same as the subsampling factor, in order to reduce aliasing. Second, if the image is significantly enlarged, it is useful (though more expensive) to apply a linear interpolation to the original image, in order to avoid constructing blocks of equivalent pixels that are visibly "jagged" when simple pixel replication is used.

Finally, there is the *upscale-to-binary* operation, in which a grayscale image is upscaled to a higher resolution binary image. This is the inverse of scale-to-gray, and can be thought of as an example of image restoration; namely, restoring a higher resolution binary image, that had been previously downscaled to gray. We don't provide any estimation routines for this explicitly, but the operation can be performed with good results by a succession of two operations: grayscale upscaling using linear interpolation, followed by binarization using a threshold or dithering. (We provide several efficient composite operations of these types.)

Aliasing occurs when a signal is (sub)sampled at less than twice the wavelength of the highest frequency. To avoid aliasing, the high frequencies in the signal are removed with a lowpass filter before sampling. The sinc filter is ideal in that its representation in the fourier domain has constant value up to the cutoff frequency, where it goes to zero. Thus, it will remove aliasing with the smallest amount of blurring of the result. However, the sinc has infinite extension, and is impractical for efficient image processing. For our purposes, a reasonable compromise between acceptable blur and computational efficiency is to use a constant height convolution filter. The function pixBlockconv() has a computational time independent of the size of the filter, and is used for some of the lowpass filtering.

We enumerate and comment on these (four) operations in the next section, and then go into detail after that. See the table at the end of this page for an overview.

Source: http://www.leptonica.com/scaling.html