Subject: Re: Convolution
Posted by Alex Schuster on Wed, 12 Sep 2001 14:20:38 GMT
View Forum Message <> Reply to Message

Kay Bente writes:

> I have to convolute a 256x256x128 Floating Point array with a 3D Gaussian
> Kernel of ~ 30x30x30, this lasts round about 45Minutes. So my question is,
> if there is any way how i can speed this up. I tried to separate this in
> each dimension with a 1D Kernel, but I donï¿½t know if I have done this
> correct (cause the procedure hangs up after a few loops)
>
> I know that the Convolution of two functions is a Multiplication in Fourier
> Space, but how can I do this with discrete arrays, do I have to enlarge my
> kernel to the size of the array i want to smooth? If so, the creation of the
> kernel with the dimensions of my array nearly lasts as long as the normal
> convolution :-(

I use the routine PSF_GAUSSIAN() to create these kernels, speed is no
problem there. The kernel has the same size as the orignal image, but
that's no problem in fourier space.
Here is some code I ripped from one of my programs. Computation takes
some seconds, not 45 minutes :)

```
pix = [ aat.x_pixel_size, aat.y_pixel_size, aat.z_pixel_size ]
r = float( radius ) / 10.0
xywidth = long( 0.5 + r/pix[0] )
zwidth = long( 0.5 + r/pix[2])

; dim[0] and dim[1] are powers of 2, make make dimz a power of 2, too,
; and use it instead of dim[2]
dimz = 4
while ( dim[2] ge dimz ) do dimz = dimz * 2
startz = (dimz-dim[2]) / 2

filter_kernel = complexarr( dim[0], dim[1], dimz )
filter_kernel[0,0,0] = psf_gaussian( $
   npixel=[dim[0]-1,dim[1]-1,dimz-1], $
   ndimen=3, $
   fwhm=[xywidth, xywidth, zwidth], /normalize )
filter_kernel = fft( shift( temporary( filter_kernel ), $
dim[0]/2+1, dim[1]/2+1, dimz/2+1 ) )

filt_image = complexarr( dim[0], dim[1], dimz )
filt_image[dim[0]*dim[1]*startz] = image
filt_image = fft( fft( temporary( filt_image ) ) * filter_kernel,
/inverse ) $
    * filter_mask
```

Alex

--

Alex Schuster Wonko@planet-interkom.de
alex@pet.mpin-koeln.mpg.de