Subject: Re: Fractional Pixels Origin?
Posted by mmiller3 on Wed, 22 Feb 2006 14:30:36 GMT
View Forum Message <> Reply to Message

>>>> > "JD" == JD Smith <jdsmith@as.arizona.edu> writes:

> My concept of fractional pixels treats them as a physical
> unit, like inches.
[...]
> My point is that there are reasons for adopting a given
> pixel origin which are entirely separate from the
> convenience of falling in line with what other tools or
> data systems have adopted (e.g. [1,1] in FITS, [0,0] in
> IDL, etc.).

That is the way I think of them as well. Generally, my code handles three coordinate systems: the data indices, spatial coordinates that represent where the data was measured, (or when, or ...), and device coordinates that correspond to physical pixels on the screen. The mapping between these coordinates is just what IDL's T3D and CONVERT_COORD are intended to handle. In fact, if I correctly handle these transformations, I don't care at all about how IDL or anyother software deals with pixels on the screen.

When I first started trying to sort this all out, a lot of my confusion came from refering to the data elements in my measurements (pixels in medical images) as "pixels" and then getting confused when I didn't want a 1 to 1 mapping between physical pixels on a display device and image pixels in my data. It helps me a lot to reserve the word "pixel" to mean "physical pixels on a display device."

For those of you who are not too bored by all this (yet), here's how I handle the coordinate transformations. I promise to mention fractional pixels, so it is still somewhat on topic!

If points in space are represented by homogeneous space coordinates r = [x,y,z,1], and homogeneous data coordinates by p = [i,j,k,1], then I create a matrix, using T3D, to transform between them: p = V # r and r = inverse(V) # p. For the simple 2d case, where I've measured at Nx by Ny points on a uniform grid with spacings dx and dy, V is just

```
   [ 1/dx    0      (Nx-1)/2 ]
V = [ 0       1/dy   (Ny-1)/2 ]
   [ 0       0      1        ]
```

In IDL, that is

T3D, /RESET
T3D, SCALE=[1.0/dx, 1.0/dy, 0.0]
T3D, TRANSLATE=[(Nx-1)/2.0, (Ny-1)/2.0, 0]

To go between device and space coordinates, I use CONVERT_COORD
after the appropriate PLOT command (usually with /NODATA) to set
up the needed system variables. To store the plot system
variables, I use the very handy savesysvar and restsysvar from
ICG_LIB, Forschungzentrum Juelich GmbH,
http://www.fz-juelich.de/icg/icg-i/idl_icglib/idl_lib_intro. html.

So, if I want to draw some images, I follow these steps

1 - set up device coordinates with PLOT and save them with
savesysvar

2 - find space coordinates of device pixels using CONVERT_COORD,
restoring plot system variables with restsysvar

3 - for each data set that I'm dealing with, calculate
(fractional) data array indices with p = V # r, using the
appropriate V for each data set. (these are what I used to
think of as fractional pixels, but I'm reserving the p word
for device pixels)

4 - INTERPOLATE the data set at the (fractional) data array
indices p. This gives me an array of data values that map 1
to 1 to the (integer) device pixels on the screen.

5 - combine multiple data sets as wanted (alpha blend, color
wash, whatever useful or glitzy method I like)

6 - TV the data to the screen

7 - if I want to overlay a plot, say using contour, I CONTOUR
after calling restsysvar

Mike

--
Michael A. Miller mmiller3@iupui.edu
Imaging Sciences, Department of Radiology, IU School of Medicine