Subject: Re: PLOTting into a 2-D array
Posted by JD Smith on Mon, 19 Dec 2005 19:53:26 GMT
View Forum Message <> Reply to Message

On Mon, 19 Dec 2005 09:42:29 -0800, M. Katz wrote:

> Gurus,
> Is there a way to use a command like PLOTS (or something related) to put
> "z" data from an arbitrary x-y path into a 2-D (floating-point, double, or
> long) image array?
>
> With
>   PLOT, x, y, /nodata
>   PLOTS, x, y, color=zarray, /data
> IDL draws a picture that can be captured as a 2D image array using simple,
> direct graphics. And IDL does this so much faster than I could possibly do
> it the long way.
>
> Here's a statement of the problem. I have experimental data gathered while
> a system is "scanning" an arbitrary (x,y) path. The data arrays can
> contain hundreds of MB of data. When I use PLOTS, and BYTSCL() the
> "signal," I can generate a decent visualization of my data fairly quickly.
> But to capture the resultant "image" means converting to whatever the
> device-limited color-scale is--like 8-bit.
>
> The "long way" solution would be to discretize the (x,y) path coordinates
> into pixel-x-y values, and then do some
> summing/histogramming/averaging/etc. on the z values. This is obviously
> very time consuming. What would be ideal is to "draw" the data into the
> array pixels, using the speed of PLOTS or a similar, optimized routine.
>
> I understand that there are some subtleties regarding multiple data points
> assigned to the same pixel and such, but I'm willing to live with a
> slightly coarse rendering if I can make back the speed of PLOTS.

I'm not sure if you are more worried about the "long way" taking too much of your computer's time, or too much of your time. As far as computer time, if you have many XY points, I'd suspect that pre-binning to some reasonable image size (like 1024x1024) for display would actually be much *faster* than attempting to draw each and every one of many millions of points. If your XY data are in a regular grid, just treat it as a large array, and use REBIN to get it into a reasonable size. Very quick and easy (both for your time, and your computer's), and it will average for you. Otherwise, HISTOGRAM is fast, and reasonably easy for this type of a problem: have a look at HIST_2D.

The problem with the approach you outline, is you are letting your screen resolution, and the order in which points are drawn determine your final "image". Obviously, if you have points at 20,000 X positions, and you are plotting to a window of X-size 512, many points will fall right on top of each other, and the "value" will be determined by the last one which fell. You'll get much better control by taking a bit of time to learn how to use REBIN/HISTOGRAM effectively.

JD