Subject: plotting data as it arrives using objects
Posted by clive_cook59 on Mon, 06 Jun 2005 16:38:02 GMT
View Forum Message <> Reply to Message

Hi,

I am trying to write an object based program which collects lidar data and plots the data as it arrives. I previously did this with direct graphics plotting each profile as it arrived, which would produce a contour plot. This was done by using the plots command and by scalling the data with a chosen colour table using the BYTSCL command.

I would like to do something similar with objects. I guess one way wold be to create a new profile using the idlgrplot object however we would be collecting several thousand lidar profiles over several hours and i can't imagine that you could create as many objects.

I hope this is clear

many thanks

Clive Cook

Subject: Re: plotting data as it arrives using objects
Posted by Rick Towler on Tue, 14 Jun 2005 17:21:10 GMT
View Forum Message <> Reply to Message

clive_cook wrote:
> Ok, i didn't do a good job of explaining myself. I've also done a
> little bit of research. Ultimately i'm looking to produce a contour
> plot (usually you can use the contour function in direct graphics or
> the idlgrcontour object). The idea is to produce this plot as data
> arrives, arriving approximately one profile every 2.5 seconds. Its a
> slow inefficient way to re-calculate the contour so in direct graphics
> i drew one profile at a time onto a plot using the plots command and
> scaling the colours to the data values (so a series of these profiles
> produces a contour plot however it is drawn on the fly).
>
> So now i am trying to do something similar with object graphics. I
> tried doing it by updating my data to the idlgrcontour object using
> contour->setproperty,data_values=new_data as new data arrives. The
> problem with this is as the new_data array increases in size the time
> taken to re-calculate the contour also increases, eventually it becomes
> far to slow. I'm looking for a way to do this in a more efficicent way.

O.K. Now you're making sense :)

First you'll want to get an idea if your design goals are practical. Determine how many vertices you will have in your final surface. You threw out some numbers a few posts ago but they don't really make sense in this context. Will you have 1,000, 10,000, 1,000,000? For instance, 1,000,000 vertices may never draw quickly on your system regardless of program design.

On my machine I noticed a few interesting things. I used the DIST() function to generate some surfaces of different sizes and then looked at the performance of the different IDLgr* objects which would be useful in your case

First thing I noticed is that there is a *lot* of overhead in IDLgrContour. Makes sense, a lot is going on. What was interesting was that a filled contour took noticeably longer to re-draw than a non-filled contour object + a filled surface object. You need to determine if you need to use IDLgrContour or if IDLgrSurface or IDLgrPolygon or some combination will be suitable.

Second thing I noticed was that even on my high end PC you aren't going to contour 1,000,000 vertices quickly. It took almost 2 minutes to draw the first instance of a *filled* contour with 50 levels. Contrast that with about 2 seconds for a surface and 27 seconds for a surface plus non-filled contour with 50 levels. Note that the N_LEVELS keyword has a large effect on draw time.

Do you really need the contour lines?

Performance wise, IDLgrPolygon would be the best way to go as there will be the least amount of overhead when updating data. As each profile comes in you'll need to mesh that row of data and append the vertex data onto your data array and the connectivity data onto your polygons array. You can get the same color by value effect with texturing but you will not have your contour lines. I have meshing code which would be of some help.

Or you may find that you can live with an IDLgrSurface object plus a non-filled IDLgrContour object. You may also find that in this case you can live with updating your contour object less frequently (say every 10-20 profiles) to limit overhead. You'll still have the surface object, the contour lines will just lag.

Determine what is possible with your hardware, if there are any upgrades that you could make to speed along the process (at this point you're CPU bound), and go from there.

-Rick
