Subject: Re: oplot in Object Graphics
Posted by Miguel ?ngel C?rdoba on Fri, 14 Nov 2003 12:29:22 GMT
View Forum Message <> Reply to Message

Hello Rick,
Yes, I'm capturing the mouse movement with the MOTION_EVENTS
keyword. The user moves the mouse over one image. This image is a part
off a volume o 17 images. Then when the user moves the mouse over the
image I plot the vertical profile off the point where is the mouse and the
vertical profile off the 8 neighbours. The vertical profile is the polyline
with the 17 points, one per image.
Then in the for statment I modify the DATA property off the 9 plot Objects.

Any ideas?

Rick Towler wrote:

> "Miguel Angel Cordoba" wrote ...
>
>
>
>> In my program the for statement only does 9 plots and before this,
>> I create the Window, the View, the model and a ObjArr of IDLgrPlot.
>> Then when the user moves the mouse in the for statement I only
>> change the data property of the IDLgrPlot.
>>
>>
>>
>
> How are you capturing the mouse movement? If you set the MOTION_EVENTS
> keyword to WIDGET_DRAW you'll be generating a *lot* of events and if inside
> your event handler you have a FOR loop which updates the DATA property of
> multiple (in your case 9) IDLgrGraphic objects you will be forcing IDL to do
> a lot of work.
>
> IDLgrGraphic objects cache internal properties so that they can be redrawn
> quickly. When you change certain external properties (such as the DATA
> property) the object marks the cache as dirty and recalculates these
> internal properties upon the next call to IDLgrGraphic::Draw. As the number
> of data points increase, so does the time it takes to update the internal
> properties of the object.
>
> So your program has a number of possible problems. Do you have to change
> the DATA property? For all of the objects every event? Can you minimize
> the size of DATA? Can you minimize the number of events by acquiring mouse
> input in a differnt way? Why are you changing the DATA property?
>

> I would suspect that the problem can be solved by taking a look at a pared
> down version of your code or at least a description of your event loop.
>
> -Rick
>
>
>
>> Karl Schultz wrote:
>>
>>
>>
>>> It is hard to tell why it may be slow without seeing the entire program.
>>>
>>> However, I have the sneaking suspicion that you are running all of the
>>>
>>>
> code
>
>
>>> listed below for each time you draw to the screen, as you would have to
>>>
>>>
> do
>
>
>>> in Direct Graphics. In Object Graphics, this is not the case. You
>>>
>>>
> create
>
>
>>> all the objects (view, model, plots) just once and then call only the
>>> window's draw method when you want to draw.
>>>
>>> The overall program logic would look something like this:
>>>
>>> Create Window
>>>
>>> Create View, Model, and Plots (essentially the code quoted below)
>>>
>>> Draw the view (e.g., oWindow->Draw, oView)
>>>
>>> REPEAT
>>> user does something.
>>> modify the data in the plot objects according to what the user did (if
>>> needed)
>>> Redraw the view (e.g., oWindow->Draw, oView)
>>> UNTIL user wants to quit

>>>
>>> The important thing is that you want to perform the step of creating the
>>> view, model, and plot objects only once. Then everytime you want to
>>>
>>>
> redraw
>
>
>>> the window, you just call Draw.
>>>
>>> If you really are doing it in the way I have just described, then you'll
>>> have to tell us more about your program. Maybe you can post the entire
>>> thing if it is not too long. How big is your plot data? How does the
>>>
>>>
> code
>
>
>>> work that triggers the drawing operation, etc..
>>>
>>> Karl
>>>
>>>
>>>
>
>
>
>

--

Miguel Angel Cordoba   mailto:cordoba@grahi.upc.es
TEL. 93 4017371
http://campus.uab.es/~2034008

Grup de Recerca Aplicada en Hidrometeorologia (GRAHi-UPC)
http://www.grahi.upc.es
