Subject: high quality 'old' direct graphics
Posted by greg.addr on Wed, 24 Nov 2010 13:53:24 GMT
View Forum Message <> Reply to Message

It's probably rather late to put out this program, now that people are using IDL 8 and its new graphics system, but perhaps some, like me, are still used to the old way of making plots. With this code you can add the most important parts that were missing: good quality screen text, smooth curves, and matching postscript output.

Essentially, it diverts the plot commands to a large pixmap, scaling up the character sizes, line thicknesses and everything else, and then shrinks the result back down for display. For postscript output, the comands are sent directly to the PS device, but introducing some calibration factors to ensure the results appear the same as on-screen. Additionally, there's an optional keyword to specify the standard character size in points to recalibrate the default IDL relative scale - this can be useful when a plot for publication requires a specific character size.

The code, example usage, and some sample output can be found here:

http://hrscview.fu-berlin.de/mex4/software/gmwindow/

Greg

```
; gmwindow object
; Greg Michael 2008-2010
;
; purpose
;   To produce high-quality anti-aliased screen plots using a syntax
as close as possible to the standard IDL
;   direct graphics commands, and to generate identical screen and
postscript output from the same
;   command sequence.
;
; requirements
;   frebin() - from IDL Astronomy User's Library at http://idlastro.gsfc.nasa.gov/
;   gm_oploterr - modified IDL library routine (added _extra keyword)
;
; initialisation
; Obj = OBJ_NEW('gmwindow', [keywords as for WINDOW command]
[,factor=value] [, ppcm=value] [,pt_size=value] [,pt_symsize=value]
[,filename=value])
;
; keywords as for direct graphics "window" command except:
;   factor - anti-aliasing factor. Object creates a plot 'factor'
```

```
times larger than required for display. The reduction of this image to
display
;              size produces intermediate pixel values on hard edges,
giving a smoother appearance. Higher values give better quality.
Default
;              value is 4.
;   ppcm - pixels/cm - if set, xsize,ysize understood in cm instead of
pixels; screen image will have dimensions [xsize,ysize]*ppcm
;            - allows simple equivalence between ps and screen plots
;   pt_size - in conjunction with ppcm, allows the standard charsize
(charsize=1) to be redefined to be a specific point size
;   pt_symsize - like pt_size, but for plot symbols
;   filename - postscript output filename

; supports following standard direct graphics commands as methods:
; erase, arrow, textwidth(), xyouts, oplot, plot, axis, tv, tvscl,
oploterr
;
; optionally supports these additional direct graphics routines from
David Fanning:
; symcat() (requires modified version, gm_symcat() with added _extra
keyword),  histoplot
;

;
; additional keywords to graphics command methods
;   no_draw - set to prevent redraw of anti-aliased window (can speed
up rendering when there any many overplots)
;

; additional methods
;   save - for ps, save and close file; for screen, send to png file
;   draw - redraw anti-aliased window (required only if image
constructed with no_draw keyword set)

;
;
;
;
;Example 1: Create simple screen plot
;
;device,decomposed=0
;loadct,0
;w=obj_new("gmwindow",title="gmwindow",xsize=600,ysize=400)
;w->plot,indgen(20),findgen(20)^2,psym=-5,ytitle="an axis
title",background=255,color=0,charsize=2
;w->oplot,indgen(20),400-findgen(20)^2,psym=-4,color=128
;w->xyouts,5,200,"a label",color=0,charsize=3;
```

```
;obj_destroy,w
;
;Note that after the object declaration, syntax is *exactly* as
standard IDL except for the initial w->
;
;
;
;
;Example 2: Create replica screen and postscript plots
;
;Two files are created, "example2.png" and "example2.eps" with
identical format plots usings lines, symbols (also
;a custom symbol from symcat), text labels and an image. The postscipt
image will have dimensions 6x4cm; the screen
;image will be 600x400 pixels. The standard font size is set at 6
points.
;
;
;pro Example2,output_folder
;    device,decomposed=0
;    loadct,0
;    for ps=0,1 do begin
;        ;when ps=0 make screen plot, when ps=1 do postscript...
;
w=obj_new("gmwindow",xsize=6,ysize=4,ppcm=100,pt_size=6,filename=output_folder
+"example2",ps=ps)
;
;        w->plot,indgen(20),findgen(20)^2,psym=-5,ytitle="an axis
title",background=255,color=0
;        w->oplot,indgen(20),400-findgen(20)^2,psym=-(w-
> symcat(11)),color=128
;        w->xyouts,5,200,"a label",color=0,charsize=1.5
;        w->tvscl,dist(50),10,10
;        w->save
;        obj_destroy,w
;    endfor
;end
;
;Example2,"d:\mydocs\tmp\"
```

Subject: Re: high quality 'old' direct graphics
Posted by Mark[1] on Thu, 25 Nov 2010 23:28:41 GMT

View Forum Message <> Reply to Message

Right now, I'm using line plots to look at trajectories of virtual
particles tracked in an ocean model. I have between between 10,000 and
50,000 such trajectories, each a few hundred data points long and I'm

showing them all on the same panel. You can't see every one, obviously, but you can see the central tendency of the distribution and the outliers. I know from experience that Direct Graphics can do this, no worries. I'm using Object Graphics, which is also pretty snappy and has the bonus that I can give each IDLgrPlot object a name and discover that name by clicking on it. I just used this to discover that trajectory 4584 was very odd--it seems to arise from a file-copying glitch.

New Graphics? Get outta here! A while back, I tried a plot with 40 line-plot objects and it took forever to display. With 10,000 I'd be waiting weeks.

Subject: Re: high quality 'old' direct graphics
Posted by penteado on Fri, 26 Nov 2010 00:26:07 GMT
View Forum Message <> Reply to Message

On Nov 25, 9:28 pm, Mark <mark.h...@gmail.com> wrote:
> New Graphics? Get outta here! A while back, I tried a plot with 40
> line-plot objects and it took forever to display. With 10,000 I'd be
> waiting weeks.

I am surprised it was fast with OG. I expected it to be about the same, since the backend of both iTools and NG is OG. There is a little more overhead in NG, but I did not expect it to be so different from OG.

Subject: Re: high quality 'old' direct graphics
Posted by Mark[1] on Mon, 29 Nov 2010 21:25:14 GMT
View Forum Message <> Reply to Message

On Nov 26, 1:26 pm, Paulo Penteado <pp.pente...@gmail.com> wrote:
> On Nov 25, 9:28 pm, Mark <mark.h...@gmail.com> wrote:
>
>> New Graphics? Get outta here! A while back, I tried a plot with 40
>> line-plot objects and it took forever to display. With 10,000 I'd be
>> waiting weeks.
>
> I am surprised it was fast with OG. I expected it to be about the
> same, since the backend of both iTools and NG is OG. There is a little
> more overhead in NG, but I did not expect it to be so different from
> OG.

I think the glacial performance of NG in this sort of case (and I confess I haven't looked into it at all thoroughly) is that every time

you add something, NG thinks very hard about how to modify the plot to accommodate it. (The Matlab graphics system has the same problem for complicated plots, for much the same reason, I suspect.)

The OG system itself is pretty snappy with line plots up to 10^5 or 10^6 points in length, surface plots up to 10^3 x 10^3. Not much slower than DG.

Subject: Re: high quality 'old' direct graphics
Posted by penteado on Mon, 29 Nov 2010 21:51:15 GMT
View Forum Message <> Reply to Message

On Nov 29, 7:25 pm, Mark <mark.h...@gmail.com> wrote:
> I think the glacial performance of NG in this sort of case (and I
> confess I haven't looked into it at all thoroughly) is that every time
> you add something, NG thinks very hard about how to modify the plot to
> accommodate it. (The Matlab graphics system has the same problem for
> complicated plots, for much the same reason, I suspect.)

Indeed that could be a factor. If so, it may be alleviated by setting explicit x and y ranges on the first plot (overplots automatically change the plot range to fit everything, unless the place where the are overplotting had explicit ranges set). And by disabling refresh during the overplots (as in p.refresh,/disable).
