Subject: POLYFILL erases my tick values - solution?
Posted by noymer on Fri, 22 Sep 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Dear c.l.i-p,

I am using POLYFILL to shade various regions of an X-Y plot different shades of grey. I use PLOT (with /nodata) to draw the axes, then I use PLOTS to draw a bunch of lines, etc. (In this case, I'm making a simple diagram, not really plotting data, but the question is pretty general.)

I'm making PostScript output so I am limited to solid colors as opposed to patterns (is there a good reason for that???), but so far that's not my real problem.

The problem is that POLYFILL seems to obliterate everything in its path, including axis tick values. For the lines I draw, there is an easy solution: I just shade first, then draw. But I am working with \DATA coordinates, so I'm quite sure (???) I have to PLOT before I can POLYFILL, hence the problem that the tick values are over-shaded.

Even the RSI "Using IDL" manual, 1998 paper version, p. 190, shows a (very ugly!) example of shading that erases tick values.

The best solution I have found is to use the AXIS procedure after POLYFILL, to re-draw the axis.

IS THIS THE BEST SOLUTION?

TIA,
Andrew

Subject: Re: POLYFILL erases my tick values - solution?
Posted by Craig Markwardt on Wed, 27 Sep 2000 07:00:00 GMT
View Forum Message <> Reply to Message

"J.D. Smith" <jdsmith@astro.cornell.edu> writes:
> I have used:
>
> plot,x,y,XRANGE=xr,YRANGE=yr,XSTYLE=5,YSTYLE=5,/NODATA,POSIT ION=p
>
> to setup data coordinates in the case they are needed before the plot which

> would have set them.

I was going to suggest it too, but I wanted to my post to be short and sweet :-) Anyway, this still doesn't get around the use of *two* PLOT commands plus the POLYFILL.

> This brings up an interesting side problem: I have a direct graphics widget
> application which undershades various parts of a plot. These shaded regions can
> be moved with the mouse or arrow keys after selection with a mouse click. Since
> they *underly* the plot, the only reliable way of moving them I've found is
> re-shading and re-plotting at each step of the move. As you can imagine, this
> causes the updates to be somewhat... unappealing. A pixmap in the normal usage
> won't seem to do the job, since usually the trick is to restore some portion of
> the window and then overplot some changing feature (like a rubber-band selection
> outline).

...

You could re-render the entire thing in an off-screen pixmap, and then dump the pixmap to the screen (or at least the relevant portion). I believe this would be a classic example of so-called double buffering in computer graphics.

If re-rendering the static scene is too expensive, then it might be possible to precompute that part of it, along with an mask which simply records the part of the image where the background doesn't show through. Then you would continuously render the changing "underneath" part of the image, and composite them together like so:

outimage = (static AND mask) + (underneath AND (NOT mask))

Obviously this can be improved further if the background color is zero, then (static AND mask) is redundant; also (NOT mask) can be precomputed.

Have fun,
Craig

P.S. It would be nice if you kept your lines < 80 characters. The default line wrap murders your text otherwise.

--

---------- ----------
Craig B. Markwardt, Ph.D. EMAIL: craigmnet@cow.physics.wisc.edu
Astrophysics, IDL, Finance, Derivatives | Remove "net" for better response
---------- ----------

Subject: Re: POLYFILL erases my tick values - solution?
Posted by davidf on Wed, 27 Sep 2000 07:00:00 GMT
View Forum Message <> Reply to Message

J.D. Smith (jdsmith@astro.cornell.edu) writes:

> This brings up an interesting side problem: I have a direct graphics widget
> application which undershades various parts of a plot. These shaded regions can
> be moved with the mouse or arrow keys after selection with a mouse click. Since
> they *underly* the plot, the only reliable way of moving them I've found is
> re-shading and re-plotting at each step of the move. As you can imagine, this
> causes the updates to be somewhat... unappealing. A pixmap in the normal usage
> won't seem to do the job, since usually the trick is to restore some portion of
> the window and then overplot some changing feature (like a rubber-band selection
> outline). There is no equivalent way for putting things *under* the restored
> area. One solution I've thought of is the SET_GRAPHICS_FUNCTION keyword of
> device with GXor, but I can't seem to make that work (since it doesn't stay
> "inside" of IDL's color table but runs the full gamut of the device's color
> table). I could also use tvrd() to store the plot and the shadings without plot
> separately, and OR them by hand. This saves all the replotting but at the
> expense of tvrd() and tv()'ing (which may make updates just as bad or worse).
>
> Any thoughts?

I think you may be stuck with this unappealing, but workable,
solution.

It is a common misconception of object graphics that when something
changes in a scene the whole thing doesn't need to be re-rendered.
It does. (I was under this misconception too when I first started
working with object graphics. There must be something in the
documentation that leads people to believe this, but I've forgotten
now what it was.)

You have a few more tricks available to you in object graphics,
of course. For example, it is possible, with some trickery and
judicious use of model objects, to take a "snap-shot" of all the
non-moving objects in a scene (this is called an "instance") and
render that over and over along with the object that is changing.
And what is "under" something else is easier to keep track of
(assuming you have a decent OpenGL compliant graphics card, of
course). But you still have to render the scene each time
something changes, and this can be fast or slow depending upon
factors that are not always in your control.

> Is this something which 3 lines of Object Graphics code could do
> in a snap?

Three lines!? No, probably not three lines. Thirty, maybe. And

you better hope it is a program that is worth the effort. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155

Subject: Re: POLYFILL erases my tick values - solution?
Posted by John-David T. Smith on Wed, 27 Sep 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Andrew wrote:
>
> Dear c.l.i-p,
>
> I am using POLYFILL to shade various regions of an X-Y plot different
> shades of grey. I use PLOT (with /nodata) to draw the axes, then I
> use PLOTS to draw a bunch of lines, etc. (In this case, I'm making a
> simple diagram, not really plotting data, but the question is pretty
> general.)
>
> I'm making PostScript output so I am limited to solid colors
> as opposed to patterns (is there a good reason for that???), but
> so far that's not my real problem.
>
> The problem is that POLYFILL seems to obliterate everything in
> its path, including axis tick values. For the lines I draw, there is
> an easy solution: I just shade first, then draw. But I am working with
> \DATA coordinates, so I'm quite sure (???) I have to PLOT before I can
> POLYFILL, hence the problem that the tick values are over-shaded.
>
> Even the RSI "Using IDL" manual, 1998 paper version, p. 190,
> shows a (very ugly!) example of shading that erases tick values.
>
> The best solution I have found is to use the AXIS procedure
> after POLYFILL, to re-draw the axis.
>
> IS THIS THE BEST SOLUTION?
>
> TIA,
> Andrew

I have used:

plot,x,y,XRANGE=xr,YRANGE=yr,XSTYLE=5,YSTYLE=5,/NODATA,POSIT ION=p

to setup data coordinates in the case they are needed before the plot which would have set them.

This brings up an interesting side problem: I have a direct graphics widget application which undershades various parts of a plot. These shaded regions can be moved with the mouse or arrow keys after selection with a mouse click. Since they *underly* the plot, the only reliable way of moving them I've found is re-shading and re-plotting at each step of the move. As you can imagine, this causes the updates to be somewhat... unappealing. A pixmap in the normal usage won't seem to do the job, since usually the trick is to restore some portion of the window and then overplot some changing feature (like a rubber-band selection outline). There is no equivalent way for putting things *under* the restored area. One solution I've thought of is the SET_GRAPHICS_FUNCTION keyword of device with GXor, but I can't seem to make that work (since it doesn't stay "inside" of IDL's color table but runs the full gamut of the device's color table). I could also use tvrd() to store the plot and the shadings without plot separately, and OR them by hand. This saves all the replotting but at the expense of tvrd() and tv()'ing (which may make updates just as bad or worse).

Any thoughts? Is this something which 3 lines of Object Graphics code could do in a snap?

Thanks,

JD

--

J.D. Smith /*\ WORK: (607) 255-6263
Cornell University Dept. of Astronomy */ (607) 255-5842
304 Space Sciences Bldg. /*\ FAX: (607) 255-5875
Ithaca, NY 14853 */

Subject: Re: POLYFILL erases my tick values - solution?
Posted by John-David T. Smith on Tue, 03 Oct 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Craig Markwardt wrote:

> You could re-render the entire thing in an off-screen pixmap, and then
> dump the pixmap to the screen (or at least the relevant portion). I
> believe this would be a classic example of so-called double buffering

> in computer graphics.

Double Buffering was definitely a good idea, Craig. It was very simple to implement and is much faster than I would have expected. In fact, it worked well enough that I think David should add it to his book as a technique to avoid flashing displays in the case that you can't simply save and device,COPY the relevant part of your graphics output. It seems to work especially well in the context of plotting. If you want to rapidly redisplay a plot with moving/changing parts and multiple display commands being invoked, this technique is definitely for you! Flicker free performance.

Here's the synopsis:

```
;; Standard widget_draw/pixwin setup, in an Init function maybe
widget_control,base,/REALIZE
widget_control,self.wDraw,GET_VALUE=win  ; Put *after* realizing
self.win=win
window,/FREE,/PIXMAP,XSIZE=xsize,YSIZE=ysize
self.pixwin=!D.WINDOW

;; When plotting
wset,self.pixwin ; instead of wset,self.win
erase
polyfill,blah,blah
plot,blah,blah,/NOERASE
oplot,blah,blah
polyfill, blah, blah  ; and etc.
;; Two additional commands complete the double buffer
wset,self.win
device,COPY=[0,0,!D.X_SIZE,!D.Y_SIZE,0,0,self.pixwin]
```

This is really just the same idea (though easier!) as the normal image buffering techniques typified by the "rubber band" example we all know and love, but with the seemingly costly step of re-displaying each and every time through, rather than saving a fixed image for copying. It does make a *big* difference in the smoothness of moving plots, and really takes only 2 additional lines surrounding your plot code (plus getting the pixwin in the first place).

Also, don't forget to free those pixwins in your cleanup routines/methods!

JD

--
J.D. Smith | WORK: (607) 255-6263
Cornell Dept. of Astronomy | (607) 255-5842

304 Space Sciences Bldg. | FAX: (607) 255-5875
Ithaca, NY 14853 |