Subject: Re: conflict between widget_draw and plot objects
Posted by David Fanning on Sun, 22 Jun 2003 15:39:34 GMT
View Forum Message <> Reply to Message

Charles writes:

> In this program I'm writing, the user can click a button which pops up
> an interface that allows them to select and view an image, and choose
> coordinates in that image, and then when they close, exports those
> coordinates back.
>
> The next stage for the user uses that same image, calculates the
> centroid of the object near the previously selected coordinates
> (selected in the previous window), does a line cut through the
> centroid, then shows a plot of the values in that line cut (using the
> plot command).
>
> That works fine.
>
> But, when I close the plot window, then go back to the first step of
> selecting/viewing an image, the pixel coordinates generated are really
> wierd, just way out there in neverland. I'm making the (possibly
> false) assumption that this has something to do with the plotting done
> in the second step. If I take out the plot line, I can go back to the
> first stage no problems.
>
> To get the pixel values I'm using:
>
> pix = (convert_coord(event.x,event.y,/DEVICE,/TO_DATA))[0:1]
>
> It seems to work fine, unless I've plotted something.
>
> I'm guessing somehow I'm capturing the wrong device coordinates or
> something, but I don't understand why.

When you plot something in a window, IDL has to set up
a mapping between the virtual data coordinate space and
the actual physical coordinate space in the window. In
other words, some pixels (but not others) have to be lit
up. Once an axis is established, for example, this mapping
is stored in the S field of the system variable responsible
for keeping track of the axis information. (In this case,
!X.S.)

There are various other bits and bobs of information
that is needed to do this transformation properly.
Most are saved in the ![XYZ] system variables, but some
are stored in the !P system variable and in the !Map

system variable (if you are working with map projections).
You could ferret all this information out of these
system variables, but in practice it is much easier to
just save everything and restore it when you need it.

So, for example, if you have two plots and you want
to draw both, then come back and overplot information
on the first, you have to restore the system variables
to their state at the end of the first plot and not to
how they are at the end of the second plot. Do do that
you will have had to save them somewhere. (In your case,
you have to restore the proper system variables so that
CONVERT_COORD will do the data to pixel conversion
properly.)

You would presumably store this information in the
info structure of your widget program. The code where
you store the information might look like this:

```
   Plot, mydata
   info.p = !P
   info.x = !X
   info.y = !Y
   info.z = !Z
   info.map = !Map
```

The code where you come back to do something to the
plot, might look like this:

```
   !P = info.p
   !X = info.x
   !Y = info.y
   !Z = info.z
   !Map = info.map
   Oplot, moredata
   data_coords = Convert_Coord(pix_coords, /Device, /To_Data)
```

Cheers,

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