Subject: Line-Mouse widget tool
Posted by Howie[1] on Thu, 24 Mar 2005 15:30:15 GMT
View Forum Message <> Reply to Message

I would like to build or use a simple pre-developed tool which
wouuld draw a plot with data points in a certain set of x-y
(possibly completely different) axes with specific units set by
the user and is able to interactively draw a "moveable" lines
on that plot with mouse clicks and notify the user of the positions
in the data-space of the endpoints of the lines. This widget tool
should then allow the user to to slightly alter the positions of
the endpoints of the drawn lines and to tell the user specifically
(to within a a user specified number of decimal places) where in
the data space the endpoints are currently located, in real time.
I would like this tool to be able to work with two separate lines,
but I guess one would be enough if only that was possible.

It would basically allow me to more easliy determine positions
in a data space which I have been having to do interactively
which takes a long time. Any ideas anybody?

cheers, Howie

Subject: Re: Line-Mouse widget tool
Posted by David Fanning on Tue, 29 Mar 2005 23:25:34 GMT
View Forum Message <> Reply to Message

Robert Barnett writes:

> I render a direct graphics plot onto a widget_draw.
> How do you transform a mouse click event which is specified in pixels to
> data-space when using direct graphics? I can compute it from the
> xmargin, ymargin, xtick_get, ytick_get, xsize and ysize. But, surely
> there must be an easier way?

Well, remember that every object I create and every-
thing I draw into a window has a coordinate system
associated with it. This is, effectively, the *data*
coordinate space. So, a medical image, for example,
might have a coordinate system in mm or cm, something
like that. And typically, these objects (images are
good examples, as are plots) know something about
where they are located in the display window.

So, these objects have a Pixel_To_Value method that
simply takes a window pixel X and Y value and can
return the proper "data" coordinate. (And also tell

me if I am inside or outside the object.)

So a call looks like this:

```
data = theObject -> Pixel_To_Value(event.x, event.y, Inside=in)
IF in THEN BEGIN
   Print, 'Data Coordinates are: ', data
ENDIF ELSE Print, 'Outside of object'
```

The Pixel_To_Value method looks something like this (stripped to the essentials):

```
PRO myObject::Pixel_To_Value, x, y, INSIDE=inside
inside = 0
self -> ApplyCoordinates ; Set up data coordinate space.
c = Convert_Coord(x, y, /Device, /To_Data)
retval = retValue = [c[0,0], c[1,0]]
IF (retValue[0] GE !X.CRange[0]) AND $
   (retValue[0] LE !X.Crange[1]) AND $
   (retValue[1] GE !Y.CRange[0]) AND $
   (retValue[1] LE !Y.Crange[1]) THEN inside = 1
RETURN, retval
END
```

The "inside" test varies a little bit, depending upon the object. What I've shown you is for some kind of plot object. If this were an image, I would be checking to see if the point is inside the image's "location" in the window.

The notion of a coordinate system associated with every object is amazingly powerful. Today I built a "ruler" object for an image, which is just a small scale that you want to display on an image to give you some idea of the size of features you are looking at. The ruler has it's own coordinate system, so I can position it in a window, and it has a holder for the parent coordinate system. It uses the parent coordinate system to see how big it needs to be, and its own coordinate system to display itself.

So I just popped it into an image object and it grows or retracts in size as I zoom in and out of the image, all the while staying in the same place in the window. Amazing!! Magic!

Let's just say there are days when you've *got* to love IDL!

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Line-Mouse widget tool
Posted by David Fanning on Tue, 29 Mar 2005 23:29:27 GMT
View Forum Message <> Reply to Message

David Fanning writes:

> The Pixel_To_Value method looks something like this (stripped
> to the essentials):
>
> PRO myObject::Pixel_To_Value, x, y, INSIDE=inside

Well, of course, that would be a FUNCTION, not a procedure. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Line-Mouse widget tool
Posted by Robert Barnett on Wed, 30 Mar 2005 00:05:57 GMT
View Forum Message <> Reply to Message

Thanks David,

That was very insightful. It seems that I totally forgot about the Convert_Coord function. Given that Convert_Coord relies on "system variables to compute coordinate conversion factors" I can see the significance of needing to call
self -> ApplyCoordinates ; Set up data coordinate space.

It's so nice to see code that makes sense. It makes IDL such a delightfully pleasant experience.
--

nrb@

Robbie Barnett
imag
Research Assistant
wsahs
Nuclear Medicine & Ultrasound
nsw
Westmead Hospital
gov
Sydney Australia
au
+61 2 9845 7223

Subject: Re: Line-Mouse widget tool
Posted by David Fanning on Wed, 30 Mar 2005 01:27:09 GMT
View Forum Message <> Reply to Message

Robert Barnett writes:

> That was very insightful. It seems that I totally forgot about the
> Convert_Coord function. Given that Convert_Coord relies on "system
> variables to compute coordinate conversion factors" I can see the
> significance of needing to call
> self -> ApplyCoordinates ; Set up data coordinate space.

Indeed, Convert_Coord *does* rely on system variables, but
not as many as you would think. In fact, setting ![XY].Window
and ![XY].S is really all that is needed for many (most?)
coordinate systems. With these two (or four, depending on
how you count) pieces of information, Convert_Coord is
happy to do its thing. :-)

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Line-Mouse widget tool
Posted by Robert Barnett on Wed, 30 Mar 2005 03:24:56 GMT
View Forum Message <> Reply to Message

While I'm on the topic, I've just been playing around with an object
wrapper for plot and oplot. I know that there are similar things out

there. However, none of them seemed like particularily good candidates for the task at hand so I cooked up my own.

Documentation at
http://www.zipworld.com.au/~retsil/idl/robbies_tools/rt_guip lot/doc/auto/

Source files at
http://www.zipworld.com.au/~retsil/idl/rt_guiplot304005.zip
http://www.zipworld.com.au/~retsil/idl/rt_guiplot304005.tar. gz

Make of it what you will.

--

nrb@
Robbie Barnett
imag
Research Assistant
wsahs
Nuclear Medicine & Ultrasound
nsw
Westmead Hospital
gov
Sydney Australia
au
+61 2 9845 7223