Subject: Re: Rubberband box for object graphics
Posted by David Fanning on Fri, 11 Feb 2005 14:10:11 GMT
View Forum Message <> Reply to Message

"Rainnie, JA \(John\)" <J.A.Rainnie@rl.ac.uk> writes:

> I'm delving into the IDL graphics objects, and specifically I wish to
> develop a plot window using the IDLgrPlot class. Thanks to several
> examples (mostly Dave Fanning's xPlot) I've mananged to do this.
> However, I really need to implement a rubberband box so the user can
> interactively zoom into this plot (like DF's zPlot). The conversion from
> device to data coordinates is of course done in direct graphics with the
> Convert_Coord function. However, I can't find any examples of how this
> is done for object graphics.
>
> Has anyone done this - and are there any examples you can direct me to.

I've never done this, but my little program SCALE_VECTOR
has solved an awful lot of problems like this for me, and
I think that is where I would start. You will know the
endpoints of your box (in the X direction), and you will
know the X extent of your View in the draw widget, so
the conversion would look something like this:

```
dataCoords = Scale_Vector([boxPt[0], boxPt[1]], view_x[0], veiw_x[1])
```

Scale_Vector is found is the usual place. :-)

Cheers,

David

P.S. After writing the code for Scale_Vector (I had a flash
of insight after working on an object graphics project) I
promptly forgot how it is that it works. But it does work, spectacularly
well. So I think whatever the algorithm is in there, that is what you
want to employ in any object graphics coordinate conversion
routine.
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Rubberband box for object graphics
Posted by b_gom on Fri, 11 Feb 2005 17:15:52 GMT
View Forum Message <> Reply to Message

Hi John,

I set out to do what you're intending to do a few years ago, and I'm still modifying the code. I can send you my zoomable plot object if you like, but it is probably not the simplest example of how to do this.

The problem I'm trying to solve right now is that IDLgrPolyline objects do not get clipped by the axis the way that IDLgrPlot objects do, so now I'm having to redesign the object heirarchy so that I can add images, polylines, etc to my plot object and have them zoomable the same way the IDLgrPlot is. If anyone has examples of doing THIS, I'd be interested..

Brad

Rainnie, JA (John) wrote:
> Hi
>
> I'm delving into the IDL graphics objects, and specifically I wish to
> develop a plot window using the IDLgrPlot class. Thanks to several
> examples (mostly Dave Fanning's xPlot) I've mananged to do this.
> However, I really need to implement a rubberband box so the user can
> interactively zoom into this plot (like DF's zPlot). The conversion
from
> device to data coordinates is of course done in direct graphics with
the
> Convert_Coord function. However, I can't find any examples of how
this
> is done for object graphics.
>
> Has anyone done this - and are there any examples you can direct me
to.
>
> Cheers
>
> John
>
> Dr. John A. Rainnie
> Imaging Systems Division
> Space Science and Technology Department
> Rutherford Appleton Laboratory
> Oxfordshire, OX11 0QX
> UK

Subject: Re: Rubberband box for object graphics

In article <MPG.1c766aa9e08148df989908@news.frii.com>, "David Fanning"
<davidf@dfanning.com> wrote:

> "Rainnie, JA \(John\)" <J.A.Rainnie@rl.ac.uk> writes:
>> I'm delving into the IDL graphics objects, and specifically I wish to
>> develop a plot window using the IDLgrPlot class. Thanks to several
>> examples (mostly Dave Fanning's xPlot) I've mananged to do this.
>> However, I really need to implement a rubberband box so the user can
>> interactively zoom into this plot (like DF's zPlot). The conversion
>> from device to data coordinates is of course done in direct graphics
>> with the Convert_Coord function. However, I can't find any examples of
>> how this is done for object graphics.
>> Has anyone done this - and are there any examples you can direct me to.

Hi John,

The simplest example I can make, shoehorned into David's xPlot is to add
the /button_events and /motion_events to the draw widget vis.

```
drawID = Widget_Draw(tlb, XSize=400, YSize=400, Color_Model=0, $
   Graphics_Level=2, Expose_Events=1, Retain=0, /button_events,/motion_events,$
   Event_Pro='XPlot_Draw_Widget_Events')
```

then add a few variables to the INFO struct to handle the rubberband box..

```
....      i_rubber_band:[0,0,0L,0L], $
         tr_rubber_band:[0,0,0.0,0.0], $
         rb_down:0L }
```

Then modify the xplot_draw_widget_events to

```
PRO XPlot_Draw_Widget_Events, event
    ; This event handler handles draw widget expose events.
Widget_Control, event.top, Get_UValue=info, /No_Copy
    ; Draw the graphic.

if(event.type eq 4) then $
  info.thisWindow->Draw, info.plotView

if(event.type eq 0) then begin
    info.i_rubber_band[0]=event.x
    info.i_rubber_band[1]=event.y
info.rb_down=1
```

```
endif else if (event.type eq 1) then begin
    info.i_rubber_band[2]=event.x
    info.i_rubber_band[3]=event.y
    info.rb_down=0

info.thisPlot->getproperty, xcoord_conv=xs,ycoord_conv=ys
info.thisWindow->getproperty,dimensions=sd

 info.tr_rubber_band[[0,2]]=info.i_rubber_band[[0,2]]*1.0/sd[ 0]
 info.tr_rubber_band[[1,3]]=info.i_rubber_band[[1,3]]*1.0/sd[ 1]

 info.tr_rubber_band[[0,2]]=(info.tr_rubber_band[[0,2]]-xs[0] )/xs[1]
 info.tr_rubber_band[[1,3]]=(info.tr_rubber_band[[1,3]]-ys[0] )/ys[1]

;plot the data
info.thisPlot->getproperty, data=d
xr=info.tr_rubber_band[[0,2]] & xr=xr[sort(xr)]
yr=info.tr_rubber_band[[1,3]] & yr=yr[sort(yr)]
plot, d[0,*], d[1,*], xrange=xr, yrange=yr

endif

if(event.type eq 2 and rb_down eq 1) then begin
 ;draw a rubberband box with polylines here.
endif

  ;Put the info structure back.
Widget_Control, event.top, Set_UValue=info, /No_Copy
END

.....
;;;;;;;;;
```

This will draw a direct graphics plot for you of the zoomed region. The basic method is to use the button and motion events to get the DEVICE coordinates. Convert these to normalised coordinates using the window dimensions, then convert the NORMAL coordinates to DATA coordinates using the plot object ycoord_conv and xcoord_conv (backwards).

There are obviously many things you need to be wary of, this example only works on the trivial case of one plot-one line. If you have a number of lines in your plot, then you need to set those up as well. This is probably easier in object graphics because you just put the IDLgrPlot objects into a new View/Window and your done. The same would be true of you have multiple views in a window, and models..

Chris.

Subject: Re: Rubberband box for object graphics
Posted by David Fanning on Sat, 12 Feb 2005 15:33:23 GMT
View Forum Message <> Reply to Message

Christopher Lee writes:

> This will draw a direct graphics plot for you of the zoomed region.

This is what you might call, literally, thinking outside the box.
I wonder if it is what the original poster has in mind. :-)

Cheers,

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

Subject: Re: Rubberband box for object graphics
Posted by Chris Lee on Sat, 12 Feb 2005 18:22:51 GMT
View Forum Message <> Reply to Message

In article <MPG.1c77cfaadd9ea45b98990b@news.frii.com>, "David Fanning"
<davidf@dfanning.com> wrote:

> Christopher Lee writes:
>
>> This will draw a direct graphics plot for you of the zoomed region.
> This is what you might call, literally, thinking outside the box. I
> wonder if it is what the original poster has in mind. :-) Cheers,
> David

Well...that's just as doable. Now that I actually tried to run zplot,
just change the PLOT command to a command to recalculate the axis and
plot, e.g. (where oxrange is the default xrange which gets re-used for a single
click with no drag along the x axis., and position is the position
keyword in xplot)

```
if(xr[0] eq xr[1]) then xr=info.oxrange

   xs2=Normalize(xr, Position=[info.position[0], info.position[2]])
   info.thisPlot->setproperty, xrange=xr, xcoord_conv=xs2
   info.xAxis1->setproperty,range=xr,/exact, xcoord_conv=xs2
   info.thisWindow->Draw, info.plotView
```

;;
You'd also need recompute=2 on the XTICK text object to make the font
stay pretty. It works like zplot, without the overlay graphics during the
drag phase..

It also needs some code to stop you zooming in too far, like zplot,
 the axis drawing breaks down after a while.

Chris.

Subject: Re: Rubberband box for object graphics
Posted by Mark Hadfield on Sun, 13 Feb 2005 20:33:33 GMT
View Forum Message <> Reply to Message

Rainnie, JA (John) wrote:
> I'm delving into the IDL graphics objects, and specifically I wish to
> develop a plot window using the IDLgrPlot class. Thanks to several
> examples (mostly Dave Fanning's xPlot) I've mananged to do this.
> However, I really need to implement a rubberband box so the user can
> interactively zoom into this plot (like DF's zPlot). The conversion from
> device to data coordinates is of course done in direct graphics with the
> Convert_Coord function. However, I can't find any examples of how this
> is done for object graphics.
>
> Has anyone done this - and are there any examples you can direct me to.

My Motley library, hosted at

  http://www.dfanning.com/hadfield/idl/README.html,

has an Object Graphics rubberband box for zooming into 2D or 3D plots.
It's implemented as one of several different "mouse handlers" that can
be selected via a drop-down list on the botttom of plot windows. The
relevant one is called the "Zoom" handler and the code for it can be
seen in the file mgh_mouse_handler_library.pro.

Make of it what you will. It's quite tigthly coupled with the rest of
the code in the library, so pulling it out would take a bit of work.

--
Mark Hadfield "Ka puwaha te tai nei, Hoea tatou"
m.hadfield@niwa.co.nz
National Institute for Water and Atmospheric Research (NIWA)
