
Subject: Re: Zooming in place
Posted by [ronn](#) on Fri, 04 Nov 2005 02:30:34 GMT
[View Forum Message](#) <> [Reply to Message](#)

Sorry for coming in late to this discussion, but Rick mentioned that doing an in place zoom in OG would be easy. And indeed it is! At least for images :) Just cut and copy the code below and you will have a simple zoom in place program for images. Left mouse is a factor of 2, right mouse is 3 and middle is 4.

The trick is to have two views where one is transparent and moveable. I suspect that you could do something similiar with regular OG plots, but I haven't tried it.

For the curious, I got this idea driving down a back country road and looking in the rear view mirror. I realized that by putting one view inside another I could display whatever I wanted! And yes, this does prove that I think about IDL way too much....

-Ronn

cut here and save to KROGwindow.pro -----

```
pro krogDrawEvent,event
;event handler
widget_control,event.top,get_uvalue=statePtr

case event.press of
  0 :
  1 : (*statePtr).zoomFactor = 2
  4 : (*statePtr).zoomFactor = 3
  2 : (*statePtr).zoomFactor = 4
  else :
endcase

if event.press ge 1 then begin
  widget_control, event.id, draw_motion_event=1 ;turn on motion
  (*statePtr).oView2->setProperty,hide=0 ;show the zoom view
  ;create an instance of the image view
  (*statePtr).oWin->draw,(*statePtr).oView,/create_instance ;
  ;get the data location
  r = (*statePtr).oWin->pickdata((*statePtr).oView,(*statePtr).oImage,
  $
      [event.x,event.y],xyzLoc)
  ;offset is the corner of the zoom lens
  offset = (*statePtr).kernalSize/2
  ;calculate the zoom lens effect based upon which mouse is clicked
  zoomParam = (*statePtr).kernalSize/(*statePtr).zoomFactor
```

```

;change the zoom view properties
(*statePtr).oView2->setProperty, $

location=[event.x-(*statePtr).kernalSize/2,event.y-(*statePtr).kernalSize/2],
$

viewplane_rect=[xyzLoc[0]-(zoomParam/2),xyzLoc[1]-(zoomParam /2), $
                zoomParam,zoomParam]
;only draw the changing part
(*statePtr).oWin->draw,(*statePtr).oView2,/draw_instance

endif else if event.release ge 1 then begin
;set everything back to normal
(*statePtr).oView2->setProperty,hide=1 ;hide the zoom view
widget_control, event.id, draw_motion_event=0
(*statePtr).oWin->draw,(*statePtr).oView
endif else if event.type eq 2 then begin
;get the data location
r = (*statePtr).oWin->pickdata((*statePtr).oView,(*statePtr).oImage,
$
                [event.x,event.y],xyzLoc)
offset = (*statePtr).kernalSize/2 ;offset is the corner of the zoom
lens
;calculate the zoom lens effect based upon which mouse is clicked
zoomParam = (*statePtr).kernalSize/(*statePtr).zoomFactor
;change the zoom view properties
(*statePtr).oView2->setProperty, $

location=[event.x-(*statePtr).kernalSize/2,event.y-(*statePtr).kernalSize/2],
$

viewplane_rect=[xyzLoc[0]-(zoomParam/2),xyzLoc[1]-(zoomParam /2), $
                zoomParam,zoomParam]
(*statePtr).oWin->draw,(*statePtr).oView2,/draw_instance
endif

return & end

;{{{::{{{::{{{::{{{::{{{::{{{::{{{::{{{::{{{::{{{::{{{::}}}

pro krOGwindow, image
;testing routine for an Object Graphics zoom lens
;Ronn Kling
;Kling Research and Software, inc
;www.rlkling.com
;www.kilvarock.com

if n_params() eq 0 then begin

```

```

file = dialog_pickfile(title='CHOOSE A TRUE COLOR IMAGE')
if file eq "" then return
image = read_image(file)
endif
device, get_screen_size=scrSize
sz = size(image,/dimen)
aspectRatio = sz[2]/float(sz[1])
xsize = (scrSize[1]-30)/aspectRatio
ysize = (scrSize[1]-30)
base = widget_base(column=1)
drawId = widget_draw(base,xsize=xsize, ysize=ysize, /button_events, $
    graphics_level=2,event_pro='krogDrawEvent', $
    renderer=1)
widget_control, base,/realize
widget_control, drawId, get_value=oWin
;create a viewplane that is the same size as the input image
oView = obj_new('IDLgrView',viewplane_rect=[0,0,sz[1],sz[2]])
oModel = obj_new('IDLgrModel')
olmage = obj_new('IDLgrImage',image,interleave=0) ;does assume 3,m,n
image
oModel->add, olmage
oView->add, oModel
;draw the image
oWin->draw,oView

;create a second view that is the size of our zoom lens
kernalSize = 200
zoomFactor = 2.0
oView2=
    obj_new('IDLgrView',dimen=[kernalSize,kernalSize],/transparent,viewplane_rect=[0,0,kernalSize/zoomFactor,kernalSize/zoom Factor])
oModel2 = obj_new('IDLgrModel')
;have to add the image object in as an alias since it is already part
;of the model above
oModel2->add, olmage,/alias
oView2->add, oModel2

statePtr = ptr_new({drawId:drawId, oWin:oWin, oView:oView, $
    oView2:oView2, olmage:olmage,
kernalSize:kernalSize, $
    zoomFactor:zoomFactor})
widget_control, base,set_uvalue=statePtr

xmanager,'krOGwindow',base,/no_block
return & end

```
