Subject: FG translation and fg getData/setData
Posted by Helder Marchetto on Fri, 13 Mar 2015 16:18:36 GMT
View Forum Message <> Reply to Message

Hi,
I'm setting up a system to draw annotation in data space on an image and I've encountered (another!) problem with fg.

So here is the (relatively) short story: I draw a square on an image in data space and I handle events a "graphic event handler class". Say I have an object selected, then I want to move this by clicking anywhere on the image and dragging the mouse and releasing. So what I do is:
- catch the position of the click in the mouse down event
- calculate the difference between current position in the mouse motion event with respect to the last position (eventually given by mouse down event or updated after processing)
- translate the object in one of either methods:

[Translation method]
delta = newPos-PreviousPos
self.myObj->tranlsate, delta[0], delta[1]

[getData/setData method]
delta = newPos-PreviousPos
self.myObj->getData, xx, yy
xx += delta[0]
yy += delta[1]
self.myObj->setData, xx, yy

Now here are the problems with these two approaches:
[Translation method]
This works, as long as one does not try to translate in data space. This proved to be tricky because if the object was scaled before, also the translations where scaled. I could not find a consistent pattern to reproduce, so forgive me for just throwing this out like this. However, using the translate method with normal coordinates works well.

[getData/setData method]
This proved to generate a clear error. The highlighted corners of the square are place elsewhere. Yeah, can't describe this any better, but the good thing, is that it's reproducible and the code is below.

After a couple of hours spent trying to figure out what was happening with data coordinates and with the get/set method, I resorted to using normal coordinates with the translation method. However, there is a bug hidden somewhere with the get/set.

Below is the code to test.

Cheers,
Helder

Instructions: Run the code below, then click on the edge of the square and move it with the mouse. In my case the result is terrible and obvious to see...

```
function SqObjEvent::MouseMotion, oWin, x, y, KeyMods
if self.isCorner then return, 1
if self.lbPressed then begin
   pos = oWin->ConvertCoord(x, y, /device, /to_data)
   delta = pos[0:1]-[self.x0,self.y0]
   self.sq->getData, xx, yy
   xx += delta[0]
   yy += delta[1]
   self.sq->setData, xx, yy
   self.x0 = pos[0]
   self.y0 = pos[1]
endif
return, 1
end

function SqObjEvent::MouseDown, oWin, x, y, button, keyMods, clicks
if button ne 1 then return, 1
if clicks gt 1 then return,1
pos = oWin->ConvertCoord(x, y, /device, /to_data)
self.x0 = pos[0]
self.y0 = pos[1]
self.lbPressed = 1b
self.sq->getData, xx, yy
nPos  = oWin->ConvertCoord( x,  y, /device, /to_norm)
nData = oWin->ConvertCoord(xx, yy, /device, /to_norm)
self.isCorner =  min(sqrt((nPos[0]-nData[0,0:3])^2+(nPos[1]-nData[1,0:3])^2)) lt 0.03
return, 1
end

function SqObjEvent::MouseUp, oWin, x, y, button
self.lbPressed = 0b
self.isCorner = 0b
return, 1
end

function SqObjEvent::Init, io, sq
self.io=io
self.sq=sq
self.x0 = 0d
self.y0 = 0d
self.isCorner = 0b
self.lbPressed = 0b
return, 1
end
```

```
pro SqObjEvent__define
void = {SqObjEvent, $
        inherits GraphicsEventAdapter, $
        io:obj_new(),$
        sq:obj_new(),$
        x0:0d,$
        y0:0d,$
        isCorner:0b,$
        lbPressed:0b}
end

pro testMoveSquare
tlb = widget_base(/column)
wWindow = widget_window(tlb, xsize=500, ysize=500);,$
widget_control, tlb, /realize
widget_control, wWindow, get_value=oWin
io = image(dist(1000), image_dimensions=[500,500], current=oWin, margin=0)
sq = polygon([0.4,0.6,0.6,0.4]*500.0,[0.4,0.4,0.6,0.6]*500.0, '-b2', fill_background=0, /data,
target=io)
oWin.event_handler = obj_new('SqObjEvent', io, sq)
xmanager, 'testMoveObjects', tlb, event_handler = 'testMoveObjects_event', /no_block
end
```