Subject: Re: Ring in function graphics
Posted by Helder Marchetto on Thu, 19 Feb 2015 23:22:08 GMT
View Forum Message <> Reply to Message

On Thursday, February 19, 2015 at 7:41:14 PM UTC+1, Chris Torrence wrote:
> On Thursday, February 19, 2015 at 9:07:27 AM UTC-7, Helder wrote:
>> Hi,
>> is it possible to draw a ring in fg?
>> For example, given two circles:
>> c = [0.5,0.5]
>> rl = 0.1
>> rh = 0.2
>> i = image(/test)
>> io = ellipse(c[0], c[1], major=rh, fill_background=1, /norm, target=i)
>> ic = ellipse(c[0], c[1], major=rl, fill_background=0, /norm, target=i)
>>
>> I would like to "see through" the inner circle (ic).
>>
>> Thanks,
>> Helder
>
> You can use a single call to POLYGON:
>
> a = findgen(101)/100*2*!PI
> x = 0.5 + [0.3*SIN(a), 0.2*SIN(a)]
> y = 0.5 + [0.3*COS(a), 0.2*COS(a)]
> w = WINDOW()
> p = POLYGON(x, y, FILL_COLOR='red', /NORMAL, LINESTYLE='none')
>
> -Chris

Hi Chris,
thanks for the tip. Very useful. I kept on thinking in ellipses() and not polygons().

Let me expand a bit on why/what I'm doing. The idea is that I would like to obtain a line profile not only from lines, but also from ellipses. These line profiles have a "thickness". For simplicity, if you have a horizontal line, a thickness of 5 (implicitly an odd number) for each line profile point you average 5 points, two lying above the actual line and two below and the point of course.

The thing is, that when you start to go away from "easy" graphics and don't use device coordinates you would like to know how thick 5 or 35 pixels look on the image. So here's my solution.

Be aware... It's quite boring and if you don't need such stuff... well... there's probably not much you can learn (except: "how not to code" :-o)

```
function getEllipseProperties, pts
mnx = min(pts[0,*],max=mxx)
```

```
mny = min(pts[1,*],max=mxy)
xc = (mnx+mxx)/2.0
yc = (mny+mxy)/2.0
mxDist = -1.0
mxPos = 0l
mnDist = sqrt((pts[0,0]-xc)^2+(pts[1,0]-yc)^2)
dList = list()
for i=0,n_elements(pts[0,*])-1 do begin
    new_mxDist = max(sqrt((pts[0,i]-xc)^2+(pts[1,i]-yc)^2), new_mxPos, min=mnDistTmp)
    if new_mxDist gt mxDist then begin
       mxPos = i
       mxDist = new_mxDist
    endif
    mnDist = mnDistTmp < mnDist
endfor
angle = atan(pts[1,mxPos]-yc,pts[0,mxPos]-xc)*!radeg
if abs(mxDist-mnDist) lt 0.0001d then begin
   mnDist = mxDist
   angle = 0d
endif
return, {center:[xc,yc], major:mxDist, minor:mnDist, angle:angle}
end


;Main test program

;start with an image
i = image(dist(500), dimensions=[500,500])
;make an ellipse
e = ellipse(0.5,0.5,major=0.2,minor=0.1,theta=32, fill_transparency=50,/norm,target=i)

;####################
;### now you may move/rotate/scale the ellipse as you like with the mouse
;####################

;Retrieve data points from ellipse
e.getData, xx, yy
pts = i->ConvertCoord(xx, yy, /norm, /to_device)
nPts = (pts.dim)[1]
thickness = 31 ;pixels, odd number
hThick = (thickness-1d)/2d
;reconstruct the ellipse properties (center, major, minor, axis)
ep = getEllipseProperties(pts)
;generate inner and outer ellipse properties
center = i->ConvertCoord(ep.center, /to_norm, /device)
axes  = i-> ConvertCoord([ep.major-hThick,ep.minor-hThick,ep.major+hThic k,ep.minor+hThick],
[0.0,0.0,0.0,0.0], /to_norm, /device)
axes = reform(axes[0,*])
```

```
a = 2d*!dpi*dindgen(nPts)/(nPts-1)
phi = ep.angle/!const.RtoD
ex = center[0] + [axes[0]* cos(a)*cos(phi) - axes[1]* sin(a)*sin(phi), axes[2]*cos(a)*cos(phi) -
axes[3]*sin(a)*sin(phi)]
ey = center[1] + [axes[1]* sin(a)*cos(phi) + axes[0]* cos(a)*sin(phi), axes[3]*sin(a)*cos(phi) +
axes[2]*cos(a)*sin(phi)]
p = polygon(ex, ey, lineStyle=' ', /norm, target=i, fill_background=1, fill_transparency=50,
fill_color='yellow')
for i=0,20 do begin & p.hide = ~p.hide & wait, 0.1 & endfor
p.delete
```

Regards,
Helder
