Subject: Creating a sphere (Object Graphics)
Posted by Jason P. Meyers on Mon, 29 Jan 2001 17:43:28 GMT
View Forum Message <> Reply to Message

Hello All,

I have recently taken a plunge into object graphics. (I think my head is still above water but I dunno for sure!) I spent many hours last night trying to create a "simple" (hah!) spherical surface and display it using Dave's FSC_Surface program. (Dave's program was worth its weight in Gold, Oops electrons don't weight much do they?)

Anyway, I came up with the procedure listed below which produces 2-D x, y, and z arrays which display a sphere. My problem is that while this works for the most part, I can still see some seems and other "imperfections" when I rotate the surface in Dave's program.

I was a bit surprised to find that IDL doesn't have a 3-D sphere function/procedure up its sleeve. Nor could I find one at Eric Deutsch's IDL web search (http://www.astro.washington.edu/deutsch/idl/htmlhelp/index.h tml). Does anyone know of a better way (or minor improvement) of making a "better looking" sphere. I don't want to spend too much more time on this part of my project. But if I could have a cool shining sphere, that would be nice.

Thanks in advance for any and all suggestions,

Jason Meyers
PhD Student, Center for Imaging Science
Rochester Institute of Technology
jpm7934@rit.edu

```
PRO JPM_Make_Sphere, n, x=x, y=y, z=z

; This procedure creates 2-D arrays that can be used to display a sphere
; using the IDLgrSurface object

; Parameter:  n : array width (must be odd, but we'll make it odd if
it's not)
;               n = 31 produces a fairly nice sphere

; Keywords:  x,y,z : named variables to contain the 2-D arrays (n x
2n-1)

; Check for the proper number of parameters
If (N_Params() ne 1) then RETURN
```

```
; check for non-numeric n and ensure it is an odd integer
nType = size(n, /type)
Case nType of
   0 : RETURN
   7 : RETURN
   8 : RETURN
   10 : RETURN
   11 : RETURN
   Else : n = abs((fix(n)/2)*2+1)
EndCase ; Done checking n

; Tweeking the value of m adjusts the spacing of points along the
z-profile.
m=1.3

; Create the arrays
x = (y = (z = fltarr(n, 2*n-1)))
xx = ((findgen(n) - float(n/2))/float(n/2)) # Transpose(fltarr(n)+1.0)
yy = rotate(xx,1)
x1 = cos(atan(yy,xx))*(sqrt(abs(1-(1-abs(xx))^m)) >
sqrt(abs(1-(1-abs(yy))^m)))
y1 = sin(atan(yy,xx))*(sqrt(abs(1-(1-abs(xx))^m)) >
sqrt(abs(1-(1-abs(yy))^m)))

x[*,0:n-1] = x1
x[*,n-1:*] = reverse(x1,2)
y[*,0:n-1] = y1
y[*,n-1:*] = reverse(y1,2)

z = sqrt((1.0-x^2.0-y^2.0) > 0.0)
z[*,n-1:*] = -reverse(z[*,0:n-1],2)

end ; JPM_Make_Sphere
```