Subject: Object Surface Shaded by Elevation (LONG)
Posted by davidf on Sun, 01 Nov 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Hi Folks,

Several weeks ago someone asked on this newsgroup how to create a surface in object graphics in which the surface was shaded by its elevation. What they wanted was the equivalent of one of these statements in direct graphics:

```
   Surface, data, Shades=BytScl(data)
   Shade_Surf, data, Shades=BytScl(data)
```

I've been puttering around with this problem in my odd moments because, frankly, I didn't know how to do it either. With the help of the friendly support people at RSI I was finally able to work it out and I include a simple example here.

The essential elements of the solution are these:

(1) Turn shading OFF. I know this seems odd for a shaded surface, but that is what you have to do. You are going to simulate shading by coloring each individual vertex.

(2) Turn all lights off. (Well, ambient lights are OK, but unnecessary.) Lights affect the shading qualities and, as I say, you don't want shading.

(3) Add a palette to the window. You can actually add a palette to an image object and drape that on the surface with the Texture_Map keyword, but I find this results in a dull surface rather than a bright one.

(4) Color each individual vertex in the surface with a color from the palette. Use the Vert_Colors keyword for this.

Here is the example. I'll probably have a more elaborate example and explanation on my web page when I get a few extra minutes. I also include the program NORMALIZE, which I just can't write an object graphics program without.

To see a wire mesh surface with elevation shading type:

```
   IDL> Object_Shade_Surface
```

To see a shaded surface (notice it doesn't look *exactly*
like the direct graphics example) type this:

```
   IDL> Object_Shade_Surface, Style=2
```

Cheers,

David

```
--------------------------------------------------------
FUNCTION Normalize, range, Position=position

    ; This is a utility routine to calculate the scaling vector
    ; required to position a vector of specified range at a
    ; specific position given in normalized coordinates. The
    ; scaling vector is given as a two-element array like this:
    ;
    ;   scalingVector = [translationFactor, scalingFactor]
    ;
    ; The scaling vector should be used with the [XYZ]COORD_CONV
    ; keywords of a graphics object or model. For example, if you
    ; wanted to scale an X axis into the data range of -0.5 to 0.5,
    ; you might type something like this:
    ;
    ;   xAxis->GetProperty, Range=xRange
    ;   xScale = Normalize(xRange, Position=[-0.5, 0.5])
    ;   xAxis, XCoord_Conv=xScale

On_Error, 1
IF N_Params() EQ 0 THEN Message, 'Please pass range vector as argument.'

IF (N_Elements(position) EQ 0) THEN position = [0.0, 1.0] ELSE $
    position=Float(position)
range = Float(range)

scale = [((position[0]*range[1])-(position[1]*range[0])) / $
    (range[1]-range[0]), (position[1]-position[0])/(range[1]-range[0])]

RETURN, scale
END
 ;--------------------------------------------------------- -----------



PRO Object_Shade_Surf, data, x, y, Style=style

    ; Check for parameters.
```

```
IF N_Elements(data) EQ 0 THEN data = Dist(50)
s = Size(data, /Dimensions)
xsize = s[0]
ysize = s[1]
IF N_Elements(x) EQ 0 THEN x = Findgen(xsize)
IF N_Elements(y) EQ 0 THEN y = Findgen(ysize)
IF N_Elements(style) EQ 0 THEN style=1

   ; Create a view. Use RGB color. Charcoal background.

thisView = OBJ_NEW('IDLgrView', Color=[80,80,80], $
  Viewplane_Rect=[-1.0,-1.0,2.0,2.0])

   ; Create a model for the surface.

thisModel = OBJ_NEW('IDLgrModel')
thisView->Add, thisModel

   ; Create an surface object shaded by elevation.

numVerts = xsize * ysize
thisSurface = OBJ_NEW('IDLgrSurface', data, x, y, Style=style, $
  Shading=0, Vert_Colors=Reform(BytScl(data), numVerts))

   ; Add the surface to the model.

thisModel->Add, thisSurface

   ; Get the data ranges for the surface.

thisSurface->GetProperty,XRange=xrange,YRange=yrange,ZRange=zrange

   ; Set scaling parameters for the surface.

xs = Normalize(xrange, Position=[-0.5,0.5])
ys = Normalize(yrange, Position=[-0.5,0.5])
zs = Normalize(zrange, Position=[-0.5,0.5])

   ; Scale the surface.

thisSurface->SetProperty,XCoord_Conv=xs, YCoord_Conv=ys, ZCoord_Conv=zs

   ; Rotate the surface model to the standard surface view.

thisModel->Rotate,[1,0,0], -90  ; To get the Z-axis vertical.
thisModel->Rotate,[0,1,0],  30  ; Rotate it slightly to the right.
thisModel->Rotate,[1,0,0],  30  ; Rotate it down slightly.
```

```
    ; Get the window destination object. Add a palette to
    ; the window.

thisWindow = Obj_New('IDLgrWindow')
thisPalette=Obj_New('IDLgrPalette')
thisPalette->LoadCT, 5
thisWindow->SetProperty, Palette=thisPalette

    ; Draw the surface.

thisWindow->Draw, thisView
END
```

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438, Toll-Free Book Orders: 1-888-461-0155
Coyote's Guide to IDL Programming: http://www.dfanning.com/