Subject: Re: IDLgrSurface with transparent lines?
Posted by Karl Schultz on Thu, 29 Apr 2004 14:05:55 GMT
View Forum Message <> Reply to Message

"Antonio Santiago" <d6522117@est.fib.upc.es> wrote in message
news:4090B32E.1040409@est.fib.upc.es...
> Hi,
>
> is there a way i can draw an IDLgrSurface object with STYLE=1 (line
> style) and this lines be semi transparen, like a transparent texture
image.

A sort of quiet feature in IDL 6.0 is that texture mapping is applied to
lines. The following test demonstrates this ability:

```
pro test
   img = BYTARR(4,2,2)
   img[1,*,*] = 255 ; green
   img[3,*,*] = 100 ; alpha
   oTexture = OBJ_NEW('IDLgrImage', img)
   oSurface = OBJ_NEW('IDLgrSurface', DIST(45), $
      COLOR=[255,255,255], STYLE=1, THICK=3, TEXTURE_MAP=oTexture)
   READ_JPEG, FILEPATH('rose.jpg', SUBDIR=['examples','data']), img, /TRUE
   oTexture2 = OBJ_NEW('IDLgrImage', img)
   oPolygon = OBJ_NEW('IDLgrPolygon', [0,1,1,0]*45, [0,0,1,1]*45, $
      COLOR=[255,255,255], TEXTURE_MAP=oTexture2, $
      TEXTURE_COORD=[[0,0],[1,0],[1,1],[0,1]])
   oModel = OBJ_NEW('IDLgrModel')
   oModel->Add, [oPolygon, oSurface]
   XOBJVIEW, oModel
end
```

If you leave the XOBJVIEW settings at their default values, then when you
click on the model to rotate it, the "drag quality" goes to medium which
lets you easily see what it looks like without transparency.

There will be an easier way to do this in IDL 6.1.

Karl

Subject: Re: IDLgrSurface with transparent lines?
Posted by Antonio Santiago on Thu, 29 Apr 2004 15:45:43 GMT
View Forum Message <> Reply to Message

Hi Karl,

thanks for your code, it has been a lot of usefull.

But really i want something like the next code, but instead with a texture map image i want use the vert_colors options. In the next code i draw a texture map image with lines in the IDLgrSurface. If you change the ";" to use vert_colors and ommit texture_map then the IDLgrSurface is drawed without transparence.
i would like can specifie a level of transparence in the colors of IDLgrSurface (if possible).

Thanks for your time.
Antonio

```
pro test
    img = BYTARR(4,2,2)
    img[1,*,*] = 255 ; green
    img[3,*,*] = 200 ; alpha

 oTexture = OBJ_NEW('IDLgrImage', img, BLEND_FUNCTION=[3,4])
 READ_JPEG, FILEPATH('rose.jpg', SUBDIR=['examples','data']), img, /TRUE

 s = SIZE(img, /DIMENSIONS)
 img2 = BYTARR(s[0]+1, s[1], s[2])

 img2[0:2, *, *] = img
 img2[3, *, *] = 100

 oTexture2 = OBJ_NEW('IDLgrImage', img2, BLEND_FUNCTION=[3,4])


    oSurface = OBJ_NEW('IDLgrSurface', DIST(45), $
       COLOR=[255,255,255], STYLE=1, THICK=2, $
  TEXTURE_MAP=oTexture2, TEXTURE_COORD=[[0,0],[1,0],[1,1],[0,1]])
  ;VERT_COLORS=[90,90,90,90, 150,150,150,150, 200,200,200,200])

    oPolygon = OBJ_NEW('IDLgrPolygon', [0,1,1,0]*45, [0,0,1,1]*45, $
       COLOR=[255,255,255], TEXTURE_MAP=oTexture, $
       TEXTURE_COORD=[[0,0],[1,0],[1,1],[0,1]])
    oModel = OBJ_NEW('IDLgrModel')
    oModel->Add, [oPolygon, oSurface]
    XOBJVIEW, oModel
end
```

Subject: Re: IDLgrSurface with transparent lines?
Posted by Rick Towler on Thu, 29 Apr 2004 17:22:09 GMT
View Forum Message <> Reply to Message

"Antonio Santiago" wrote in message...

> But really i want something like the next code, but instead with a
> texture map image i want use the vert_colors options. In the next code i
> draw a texture map image with lines in the IDLgrSurface. If you change
> the ";" to use vert_colors and ommit texture_map then the IDLgrSurface
> is drawed without transparence.
> i would like can specifie a level of transparence in the colors of
> IDLgrSurface (if possible).

You can do this by creating a texture containing your palette and then selecting your texture coordinates appropriatly. It is a bit more work, but you'll have control of the color at every vertex.

-Rick

```
oPalette = obj_new('IDLgrPalette')
oPalette -> LoadCT, 13

;  Create a 256x256 texture map modulating color and opacity
imagedata = BYTARR(4, 256, 256, /NOZERO)
for alpha = 0, 255 do begin
   for color=0,255 do begin
      imagedata[0:2,alpha,color] = oPalette -> GetRGB(color)
      imagedata[3,alpha,color] = alpha
   endfor
endfor
oTexture = OBJ_NEW('IDLgrImage', imagedata, BLEND_FUNCTION=[3,4])


;  Create the surface object
z = DIST(45)
oSurface = OBJ_NEW('IDLgrSurface', DIST(45), COLOR=[255,255,255], $
   STYLE=1, THICK=2, TEXTURE_MAP=oTexture)
oModel = OBJ_NEW('IDLgrModel')
oModel -> Add, oSurface

;  Create the texture coordinates
;  Get the surface vertices
oSurface -> GetProperty, DATA=surfVerts
nVerts = SIZE(surfVerts, /DIMENSIONS)
nVerts = nVerts[1] * nVerts[2]
```

```
;  Reform so it is a bit easier to work with
surfVerts = REFORM(surfVerts, 3, nVerts)

;  Find the max
maxZ = MAX(surfVerts[2,*])

;  Create the texture coordinate array x&y for each vertex
texCoords = FLTARR(2,nVerts, /NOZERO)

;  set Color (1) and Alpha (0) according to height
;  Fudge the texcoord max otherwise you'll have a hole
;  in the top.
texCoords[0,*] = 0.0 > (surfVerts[2,*] / maxZ) < 0.99999
texCoords[1,*] = 0.0 > (surfVerts[2,*] / maxZ) < 0.99999

oSurface -> SetProperty, TEXTURE_COORD=texCoords



;  Compare to VERT_COLORS

;  Create the vert colors array
vertColors = BYTARR(3, nVerts, /NOZERO)
for v=0, nVerts-1 do $
   vertColors[*,v] = oPalette -> GetRGB(texCoords[0,v] * 255)

;  Create the vert colors surface
oSurfaceVC = OBJ_NEW('IDLgrSurface', DIST(45), COLOR=[255,255,255], $
   STYLE=1, THICK=2, VERT_COLORS=vertColors)
oModelVC = OBJ_NEW('IDLgrModel')
oModelVC -> Add, oSurfaceVC

XOBJVIEW, oModelVC
XOBJVIEW, oModel, /BLOCK

OBJ_DESTROY, [oModel, oModelVC, oTexture, oPalette]

end
```