Subject: Re: Rotation of 3D image in Object Graphics
Posted by G Karas on Fri, 12 Apr 2002 18:47:52 GMT
View Forum Message <> Reply to Message

very good answer..

i would also like to point out that, in my opinion,
3D rotation of objects is supplied for display purposes
and not for any serious volume rotations.. i did not see
any interpolation options in the objects.. and that could
be a major drawback with partial volume effects in
medical imaging.

cheers!

Giorgos
"Karl Schultz" <kschultz@devnull.researchsystems.com> wrote in message
news:a96rh3$j41$1@news.rsinc.com...
>
> "Akhila" <idlfreak@yahoo.com> wrote in message
> news:b1ad7b05.0204111547.3c64146a@posting.google.com...
>> HI,
>>   I've written the code to perform rotation. I used IDLgrModel->Rotate
>> property. But it doesn't do what i need to. Can anybody tell me why
>> this is happening and what should i do to obtain a 3D rotation of the
>> image.
>>   Thanks for any help.
>>
>>   Cheers,
>>   Akhila.
>
> IDLgrImage objects don't rotate in the way you are expecting them to here.
> The location of the image corners are affected by a model transform, but
the
> image is always drawn in a box whose sides are parallel to the window
sides.
>
> The docs say:
>
> An image object represents a mapping from a two-dimensional array of data
> values to a two dimensional array of pixel colors, resulting in a flat
> 2D-scaled version of the image, drawn at Z = 0.
> The image object is drawn at Z =0 and is positioned and sized with respect
> to two points:
>
> p1 = [LOCATION(0), LOCATION(1), 0]
>
> p2 = [LOCATION(0) + DIMENSION(0), LOCATION(1) + DIMENSION(1), 0].

>
> where LOCATION and DIMENSION are properties of the image object. These
> points are transformed in three dimensions, resulting in screen space
points
> designated as p1' and p2'. The image data is drawn on the display as a 2D
> image within the 2D rectangle defined by (p1'[0], p1'[1] - p2'[0],
p2'[1]).
> The 2D image data is scaled in 2D (not rotated) to fit into this projected
> rectangle and then drawn with Z buffering disabled
>
>
> So, if you really want to rotate the image, you can texture map it onto a
> polygon. The code below, modified from yours, does this. I also changed
it
> to roate around the Z axis.
>
> The other option is to rotate your data before putting it into the image
> object.
>
> Karl
>
>
>
>
> ;-- -------------
>
> PRO rotation_event, event
>
> Widget_Control, event.top, Get_UValue = state
> state.oWindow -> Draw, state.oView
>
> END
>
> ;-- -------------
>
> PRO rotateleft_event, event
>
> Widget_Control, event.top, Get_UValue = info
> info.oModel->rotate, [0,0,1], 5
> info.oWindow -> Draw, info.oView
> Widget_Control, event.top, Set_UValue = info, /No_Copy
>
> END
>
> ;-- -------------
>
> PRO rotation
>

```
> filename = FILEPATH(Subdirectory = ['examples', 'data'], 'head.dat')
> OPENR, lun, filename, /GET_LUN
> data = BYTARR(80,100,57)
> READU, lun, data
> FREE_LUN,lun
> SHADE_VOLUME, data, 50, v, p, /LOW, /VERBOSE
> SCALE3, XRANGE = [0,80], YRANGE = [0,100], ZRANGE = [0,57]
> image = POLYSHADE(v,p, /T3D)
>
> xsize = 512
> ysize = 512
>
> tlb = Widget_Base(Title='Image Window/Leveling Example', Column=1,$
> MBar=menuID, Base_Align_Center=1)
> trb = Widget_base(tlb, /Row)
> Button7 = Widget_Button(trb, VALUE = 'Rotate Left', UVALUE = $
> 'rotateleft', Event_Pro = 'rotateleft_event')
> drawID = Widget_Draw(tlb, XSize=xsize, YSize=ysize, /BUTTON_EVENTS, $
> /EXPOSE_EVENTS, retain = 0, GRAPHICS_LEVEL = 2)
>
> Widget_Control, tlb, /Realize
> Widget_Control, drawID, Get_Value=oWindow
>
> sclimage = Bytscl(image, Min = displayMin, Max = displayMax)
> oImage = Obj_New('IDLgrImage', image)
> oPoly = obj_new('idlgrpolygon', [0,400,400,0],[0,0,400,400],
> TEXTURE_MAP=oImage, $
>    color=[255,255,255], TEXTURE_COORD=[[0,0],[1,0],[1,1],[0,1]])
> oView = Obj_New('IDLgrView', VIEWPLANE_RECT = [0,0,512,512], COLOR = $
> [0,0,0], PROJECTION = 2)
> oModel = Obj_New('IDLgrModel')
> oModel -> Add, oPoly
> oView -> Add, oModel
> oWindow -> Draw, oView
>
> info = { oModel:oModel, $
> oView:oView, $
> oWindow:oWindow}
>
> Widget_Control, tlb, Set_UValue=info, /No_Copy
>
> XManager, 'rotation', tlb, /No_Block
>
> END
>
>
>
```