Subject: Re: rotate and transpose images in xobjviewer
Posted by Dick Jackson on Sun, 01 Apr 2007 04:41:42 GMT
View Forum Message <> Reply to Message

Hi vijay,

"vijay" <vijayansiva@gmail.com> wrote in message
news:1175343528.461497.166670@p15g2000hsd.googlegroups.com.. .
>
> Hi, i am having a problem in rotating an image in the xobjviewer. the
> problem is not with the rotate button but my image itself rotated and
> transposed in the xobjviewer. So i want the image in the xobjviewer to
> be transposed and rotated for 270 degrees. i can't able to do this
> using the rotate command also. So tell me how to rotate and transpose
> the image.

The IDLgrImage object does not create a real 3D rotatable object, but what I think you want is a rectangular IDLgrPolygon object with an image as its texture map. (I'm guessing here, because you didn't give a lot of detail on what you had tried so far)

```
;;    Get an image

file = Filepath('rose.jpg', Subdir=['examples', 'data'])
Read_JPEG, file, image

;;    Make an IDLgrImage object

oImage = Obj_New('IDLgrImage', image)
XObjView, oImage, Title='oImage'        ; Cannot really rotate the image

;;    Make an IDLgrPolygon object with the image as its texture map

oImage -> GetProperty, Dimensions=dims  ; [width, height]
oPoly1 = Obj_New('IDLgrPolygon', $
              [[0,0,0],[dims[0],0,0],[dims[0],dims[1],0],[0,dims[1],0]], $
              Color=[255,255,255], Texture_Map=oImage, $
              Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
XObjView, oPoly1, Title='oPoly1', XOffset=300

;;    Try with no lighting to avoid change in shading when rotated

oHiddenLight = Obj_New('IDLgrLight', Type=0, Color=[255,0,0], /Hide)
XObjView, oPoly1, Title='oPoly1 with no lighting', XOffset=600, $
        Stationary=oHiddenLight

;;    Create polygon with coordinates for polygon rotated 270 degrees
```

```
oPoly2 = Obj_New('IDLgrPolygon', $
            [[0,dims[0],0],[0,0,0],[dims[1],0,0],[dims[1],dims[0],0]], $
            Color=[255,255,255], Texture_Map=oImage, $
            Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
XObjView, oPoly2, Title='oPoly2 with no lighting', XOffset=600, YOffset=300, $
        Stationary=oHiddenLight
```

There are other subtle issues with image row order settings (the Order property to IDLgrImage) and resampling of image pixels (see the Texture_Map property of IDLgrPolygon), but is this what you were looking for?

Cheers,
-Dick

--
Dick Jackson Software Consulting http://www.d-jackson.com
Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com

Subject: Re: rotate and transpose images in xobjviewer
Posted by eduardo.iturrate on Sun, 01 Apr 2007 20:21:34 GMT
View Forum Message <> Reply to Message

You can also use Revolution IDL, available at:

http://www.ittvis.com/codebank/search.asp?FID=473

> From the main menu of the application, choose Add -> Image. Choose the "Open File" option and then select your image file. Once displayed, click on the "Source Code" tab, select Render to: xObjView and click on the "IDL>" button. The following source code will be generated:

```
PRO scg_og
;
; Generated by Revolution IDL, the Source Code Generator for IDL Object Graphics.
; Code created on  1 Apr 2007 14:21:47
; Version 1.6
;
; This code is provided 'as-is', without warranty of any kind.
; Permission to use, modify or distribute this code is granted,
; including its use for commercial applications.
;
; For more information about IDL, visit  http://www.ittvis.com/idl/
;
```

```
;
; ========================
; Part 1 - Data Definition
; ========================
;

_red = -1  &   _green = -1   &   _blue = -1
_img = READ_IMAGE('C:\RSI\IDL63\examples\data\avhrr.png', _red,
_green, _blue)
_sz = size(_img)
if _red[0] ne -1 then begin   ; just in case it's an indexed color
image
 _img2 = bytarr(3, _sz[1], _sz[2])
 _img2[0,*,*] = _red[_img]
 _img2[1,*,*] = _green[_img]
 _img2[2,*,*] = _blue[_img]
 _img = _img2
endif
if _sz[0] eq 2 then _proportion = _sz[2] / float(_sz[1]) else
_proportion = _sz[3] / float(_sz[2])
img1048 = obj_new('IDLgrImage', /no_copy, $
  _img, loc=[0.0,0.0], dim=[0.01,0.01], hide=1)
_xp = 1.0
_yp = 1.0 * _proportion
_zp = 0.001
points3617 = [[-_xp, -_yp, _zp],[_xp, -_yp, _zp],[_xp, _yp, _zp],[-
_xp, _yp, _zp]]
;
;
; ========================================
; Part 2 - Object Definition and Properties
; ========================================
;
oView = OBJ_NEW('IDLgrView')

oView->SetProperty, $
   COLOR= [255,255,255], $
   DEPTH_CUE= [0.000000,0.000000], $
   DIMENSIONS= [1280,1280], $
   DOUBLE= 0, $
   EYE= 4.0000000, $
   LOCATION= [0.000000,0.000000], $
   OBLIQUE= [0.000000,0.000000], $
   PROJECTION= 2, $
   TRANSPARENT= 0, $
   VIEWPLANE_RECT= [-1.4142135, -1.4142135, 2.8284271, 2.8284271], $
   ZCLIP= [2.0000000,-5.0000000]
```

```
oModel0 = OBJ_NEW('IDLgrModel')

oModel0->SetProperty, $
   HIDE= 0

oView->Add, oModel0

oModel1 = OBJ_NEW('IDLgrModel')

oModel1->SetProperty, $
   TRANSFORM= [[0.88302278, -0.46859059, 0.026355628, 0.00000000], $
    [0.32139380, 0.64465485, 0.69363295, 0.00000000], $
    [-0.34202014, -0.60402277, 0.71984677, 0.00000000], $
    [0.00000000, 0.00000000, 0.00000000, 1.0000000]], $
   HIDE= 0

oModel0->Add, oModel1

oModel2 = OBJ_NEW('IDLgrModel')

oModel2->SetProperty, $
   HIDE= 0

oModel1->Add, oModel2

oModel3 = OBJ_NEW('IDLgrModel')

oModel3->SetProperty, $
   HIDE= 0

oModel2->Add, oModel3

oPolygon0 = OBJ_NEW('IDLgrPolygon', points3617)

oPolygon0->SetProperty, $
   COLOR= [255,255,255], $
   TEXTURE_COORD= [[0,0], [1,0], [1,1], [0,1]], $
   TEXTURE_MAP= img1048, $
   XCOORD_CONV=[0.00000000,1.0000000],
YCOORD_CONV=[0.00000000,1.0000000], ZCOORD_CONV=[0.00000000,1.0000000]

oModel3->Add, oPolygon0

oLight0 = OBJ_NEW('IDLgrLight')

oLight0->SetProperty, $
   INTENSITY= 0.500000, $
```

```
   LOCATION= [2.0000000, 2.0000000, 5.0000000], $
   TYPE= 2

oModel0->Add, oLight0


oLight1 = OBJ_NEW('IDLgrLight')

oLight1->SetProperty, $
   INTENSITY= 0.500000, $
   TYPE= 0

oModel0->Add, oLight1


;
; ================================
; Part 3 - Destination and Render
; ================================
;
XOBJVIEW, oModel0, XSIZE=650, YSIZE=650


END
```

Not the most efficient code, but it's all done for you!

Eduardo.

Subject: Re: rotate and transpose images in xobjviewer
Posted by Steven Houston on Tue, 03 Apr 2007 10:06:10 GMT
View Forum Message <> Reply to Message

Dick Jackson wrote:
> Hi vijay,
>
> "vijay" <vijayansiva@gmail.com> wrote in message
> news:1175343528.461497.166670@p15g2000hsd.googlegroups.com.. .
>> Hi, i am having a problem in rotating an image in the xobjviewer. the
>> problem is not with the rotate button but my image itself rotated and
>> transposed in the xobjviewer. So i want the image in the xobjviewer to
>> be transposed and rotated for 270 degrees. i can't able to do this
>> using the rotate command also. So tell me how to rotate and transpose
>> the image.

```
>
>  The IDLgrImage object does not create a real 3D rotatable object, but what I
>  think you want is a rectangular IDLgrPolygon object with an image as its texture
>  map. (I'm guessing here, because you didn't give a lot of detail on what you had
>  tried so far)
>
>
>  ;;    Get an image
>
>  file = Filepath('rose.jpg', Subdir=['examples', 'data'])
>  Read_JPEG, file, image
>
>  ;;    Make an IDLgrImage object
>
>  oImage = Obj_New('IDLgrImage', image)
>  XObjView, oImage, Title='oImage'        ; Cannot really rotate the image
>
>  ;;    Make an IDLgrPolygon object with the image as its texture map
>
>  oImage -> GetProperty, Dimensions=dims  ; [width, height]
>  oPoly1 = Obj_New('IDLgrPolygon', $
>                [[0,0,0],[dims[0],0,0],[dims[0],dims[1],0],[0,dims[1],0]], $
>                Color=[255,255,255], Texture_Map=oImage, $
>                Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
>  XObjView, oPoly1, Title='oPoly1', XOffset=300
>
>  ;;    Try with no lighting to avoid change in shading when rotated
>
>  oHiddenLight = Obj_New('IDLgrLight', Type=0, Color=[255,0,0], /Hide)
>  XObjView, oPoly1, Title='oPoly1 with no lighting', XOffset=600, $
>           Stationary=oHiddenLight
>
>  ;;    Create polygon with coordinates for polygon rotated 270 degrees
>
>  oPoly2 = Obj_New('IDLgrPolygon', $
>                [[0,dims[0],0],[0,0,0],[dims[1],0,0],[dims[1],dims[0],0]], $
>                Color=[255,255,255], Texture_Map=oImage, $
>                Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
>  XObjView, oPoly2, Title='oPoly2 with no lighting', XOffset=600, YOffset=300, $
>           Stationary=oHiddenLight
>
>
>  There are other subtle issues with image row order settings (the Order property
>  to IDLgrImage) and resampling of image pixels (see the Texture_Map property of
>  IDLgrPolygon), but is this what you were looking for?
>
>  Cheers,
>  -Dick
```

>
> --
> Dick Jackson Software Consulting http://www.d-jackson.com
> Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com
>
>

Hi,

As of IDL 6.2 you can apply 3D transforms to an IDLgrImage, but you need
to set TRANSFORM_MODE=1 (the default is to use the 2D transform mode for
backwards compatibility).

So in 6.2 and later you can do:

```
;;    Get an image
file = Filepath('rose.jpg', Subdir=['examples', 'data'])
Read_JPEG, file, image

;;    Make an IDLgrImage object
oImage = Obj_New('IDLgrImage', image, TRANSFORM_MODE=1)

;; Create a model to apply the 270 degree rotation
oModel = Obj_New('IDLgrModel')
oModel->Rotate, [0,0,1], 270
oModel->Add, oImage

XObjView, oModel, Title='oImage'
```

Cheers,
Steve.