Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by David Fanning on Tue, 08 Nov 2005 02:46:08 GMT
View Forum Message <> Reply to Message

Victor writes:

> I have 3 images (im1,im2,im3) which I need to display on XObjView (or
> any alternate object graphics viewer).

Any particular reason these have to be displayed in object graphics?
In direct graphics I would do this:

```
!P.Multi=[0,3,1]
TVImage, im1
TVImage, im2
TVImage, im3
!P.Multi=0
```

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Antonio Santiago on Tue, 08 Nov 2005 10:20:53 GMT
View Forum Message <> Reply to Message

Victor wrote:
> I have 3 images (im1,im2,im3) which I need to display on XObjView (or
> any alternate object graphics viewer). When I tried to add more than 1
> image to the same graphics object as follows:
>
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, im1
>> myModel->Add, im2
>
>
> IDL complains:
> "IDLGRMODEL::ADD: Objects can only have one parent at a time:
> <ObjHeapVar9869(IDLGRCOLORBAR)>"
>
> Huh! Okay so to get around this, I created separate objects for the 3
> images, hoping to add the 3 objects onto the same view as follows:

>
>
>> myModel1 = Obj_New('IDLgrModel') & myModel1->Add, im1
>> myModel2 = Obj_New('IDLgrModel') & myModel2->Add, im2
>> myModel3 = Obj_New('IDLgrModel') & myModel3->Add, im3
>
>
> Display these using:
> XOBJVIEW,myModel1,TITLE=myModel1'
> XOBJVIEW,myModel2,TITLE=myModel2 '
> XOBJVIEW,myModel3,TITLE=myModel3 '
>
> These obviously plot the three images into 3 separate XObjView windows.
> I want them to be displayed in the same XObjView window. So I try to
> add them to a container and display the it in XObjView as follows:
> myContainer = OBJ_NEW('IDL_Container')
> myContainer->Add,myModel_1, POSITION=0.1
> myContainer->Add,myModel_2, POSITION=0.5
> myContainer->Add,myModel_3, POSITION=0.9
> XOBJVIEW,myContainer,TITLE='All 3 images'
>
> IDL complains (This time a nastier one):
> "XOBJVIEW: IDLGRMODEL::ADD: The ALIAS keyword is only allowed for
> IDLgrComponent class objects."
>
> Looks like XOBJVIEW doesn't even work with IDL_Container or IDLgrView
> objects either (I tried that)
>
> Also,
>
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, [myModel_1, myModel_2, myModel_3], POSITION=[10,20,30]*B
>> XOBJVIEW,myModel,TITLE='All 3 images'
>
> Works, but doesn't give the expected results (just overlays the three
> images, Position keyword seems to have no effect whatso ever)
>

Try to use the LOCATION keyword on every IDLgrImage object. It works for
me. Every image has its particular location (and all three overlaped).

> I know the following might be useful, but I am not sure how to
> incorporate them in the above logic (which I'm not even sure is the
> best approach)
> 1) "!P.MULTI"
> 2) "Position"
>
> Sorry for the long post. I seem to be lost and would appreciate any

> insights into this matter. Thanks alot
>

--
---
Antonio Santiago Pï¿½rez
( email: santiago<<at>>grahi.upc.edu )
( www: http://www.grahi.upc.edu/santiago )
( www: http://asantiago.blogsite.org )
---
GRAHI - Grup de Recerca Aplicada en Hidrometeorologia
Universitat Politï¿½cnica de Catalunya
---

---

## Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Victor[1] on Tue, 08 Nov 2005 18:10:07 GMT
View Forum Message <> Reply to Message

Thanks for the suggestion David. Yes there is a definite reason for
doing so. I could have mentioned this in my previous post, but for the
sake of brevity, I didn't. Well.. The three images im1, im2 and im3 are
created using object graphics (using the concept of alpha channels.)

Now, im1, im2 and im3 are supposed to be separately overlaid on a
background image (say imback - and imback, im1, im2 and im3 all have
the same dimensions.) There is also a colorbar graphics object which
finally needs to show up on the side of these 3 overlaid images.
So..... suggestions please ! Could you please guide me how !P.Multi
work in object graphics (maybe, specifically in my case)?

Thanks,
- Vaibhav

David Fanning wrote:
> Victor writes:
>
>
>> I have 3 images (im1,im2,im3) which I need to display on XObjView (or
>> any alternate object graphics viewer).
>
> Any particular reason these have to be displayed in object graphics?
> In direct graphics I would do this:
>
> !P.Multi=[0,3,1]
> TVImage, im1
> TVImage, im2

>  TVImage, im3
>  !P.Multi=0
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming: http://www.dfanning.com/

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Karl Schultz on Tue, 08 Nov 2005 18:13:37 GMT
View Forum Message <> Reply to Message

On Mon, 07 Nov 2005 17:07:45 -0800, Victor wrote:
<snip>
> Also,
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, [myModel_1, myModel_2, myModel_3], POSITION=[10,20,30]*B
>> XOBJVIEW,myModel,TITLE='All 3 images'
> Works, but doesn't give the expected results (just overlays the three
> images, Position keyword seems to have no effect whatso ever)

This approach is the closest you came to the correct way of doing it.

I'm not sure why, but it seems that you have developed the idea
that the POSITION keyword parameter has something to do with spatial
locations. POSITION for IDLgrModel and IDL_Container refers to the
ordering of objects in a container. For example, POSITION=5 means to
insert the object into the container in the 6th position (POSITION is
zero-based). That is, the object will have 5 objects that preceed it in
container order after it is inserted. POSITION is an index, not a spatial
coordinate.

There are a number of ways to change an image's displayed location. The
IDLgrImage LOCATION property is one way that was mentioned by another
poster.

Since you have a model for each image you can also do:

```
myModel = Obj_New('IDLgrModel')
myModel->Add, [myModel_1, myModel_2, myModel_3]
myModel_1->Translate, 10, 0, 0
myModel_2->Translate, 20, 0, 0
myModel_3->Translate, 30, 0, 0
XOBJVIEW, myModel, TITLE='All 3 images'
```

Substitute offsets that make sense for your image sizes.

Karl

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Dick Jackson on Tue, 08 Nov 2005 18:16:12 GMT
View Forum Message <> Reply to Message

Hi Victor,

"Victor" <vmiscellaneous@yahoo.com> wrote in message
news:1131412065.067552.155280@g49g2000cwa.googlegroups.com.. .
> I have 3 images (im1,im2,im3) which I need to display on XObjView (or
> any alternate object graphics viewer). When I tried to add more than 1
> image to the same graphics object as follows:
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, im1
>> myModel->Add, im2
>
> IDL complains:
> "IDLGRMODEL::ADD: Objects can only have one parent at a time:
> <ObjHeapVar9869(IDLGRCOLORBAR)>"
>
> Huh!

Huh, indeed! I don't see any reason this should happen from what you've said above. (there are no colorbars mentioned up there)

I see that Antonio and Karl have covered the Position/Location issue, here's another example that should work fine:

```
im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10])
im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)))
im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)), Location=[10,10])
myModel = Obj_New('IDLgrModel')
myModel->Add, [im1, im2, im3]
XObjView, myModel
```

Now, this gives the images rendered nicely at first but has the odd behaviour of IDLgrImages if you rotate the view. They don't rotate, they stretch and squish. If you really want images that can rotate, you'll need to make a polygon that contains each, something like this:

```
im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)))
im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)))
im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)))
```

```
poly1=Obj_New('IDLgrPolygon', [[-10,10],[0,10],[0,20],[-10,20]], $
        Color=[255,255,255], Texture_Map=im1, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
poly2=Obj_New('IDLgrPolygon', [[0,0],[10,0],[10,10],[0,10]], $
        Color=[255,255,255], Texture_Map=im2, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
poly3=Obj_New('IDLgrPolygon', [[10,10],[20,10],[20,20],[10,20]], $
        Color=[255,255,255], Texture_Map=im3, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
myModel = Obj_New('IDLgrModel')
myModel->Add, [poly1, poly2, poly3]
XObjView, myModel
```

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Rick Towler on Tue, 08 Nov 2005 18:38:08 GMT
View Forum Message <> Reply to Message

Victor wrote:
> I have 3 images (im1,im2,im3) which I need to display on XObjView (or
> any alternate object graphics viewer). When I tried to add more than 1
> image to the same graphics object as follows:
>
```
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, im1
>> myModel->Add, im2
```
>
>
> IDL complains:
> "IDLGRMODEL::ADD: Objects can only have one parent at a time:
> <ObjHeapVar9869(IDLGRCOLORBAR)>"

You have something else going on here. You *can* add multiple images to an instance of IDLgrModel. The error is complaining about a colorbar object. In your code below you add 3 images to a single model.

> These obviously plot the three images into 3 separate XObjView windows.
> I want them to be displayed in the same XObjView window. So I try to
> add them to a container and display the it in XObjView as follows:

> myContainer = OBJ_NEW('IDL_Container')
> myContainer->Add,myModel_1, POSITION=0.1
> myContainer->Add,myModel_2, POSITION=0.5
> myContainer->Add,myModel_3, POSITION=0.9
> XOBJVIEW,myContainer,TITLE='All 3 images'
>
> IDL complains (This time a nastier one):
> "XOBJVIEW: IDLGRMODEL::ADD: The ALIAS keyword is only allowed for
> IDLgrComponent class objects."
>
> Looks like XOBJVIEW doesn't even work with IDL_Container or IDLgrView
> objects either (I tried that)
>
> Also,
>
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, [myModel_1, myModel_2, myModel_3], POSITION=[10,20,30]*B
>> XOBJVIEW,myModel,TITLE='All 3 images'
>
> Works, but doesn't give the expected results (just overlays the three
> images, Position keyword seems to have no effect whatso ever)

No, XOBJVIEW only works with IDLgrModel objects and subclasses. You are also misunderstanding the POSITION keyword of IDL_Container and IDLgrModel. First, it is a long index value that specifies the "place in line" of the object you are adding. Think of a stack. The position is the location in the stack, starting at 0, from the top down. POSITION has nothing to do with locating the object in world space.

> I know the following might be useful, but I am not sure how to
> incorporate them in the above logic (which I'm not even sure is the
> best approach)
> 1) "!P.MULTI"
> 2) "Position"

!P.MULTI is a direct graphics system variable and has no meaning in object graphics.

As I mentioned above, the POSITION keyword isn't what you are looking for. As some Antonio suggested, check out the LOCATION keyword of IDLgrImage instead.

Another approach would be to transform your image data such that your images are lined up as you desire. For example, say I have 2 images that are 100x100 pixels. I first add each of them to a model:

mod1=OBJ_NEW('IDLgrModel')

mod1->add, img1

mod2=OBJ_NEW('IDLgrModel')
mod2->add, img2

Then I move the second image to the right (positive X direction) by
using the TRANSLATE method:

mod2->translate, 100,0,0

Now viewing the images in XOBJVIEW I have 2 images side by side:

XOBJVIEW, [mod1,mod2]

Don't get too crazy with this though.  For reasons we don't need to get
into here, rotating IDLgrImage objects that have been translated doesn't
always yield the desired result.  Remember view->full reset in XOBJVIEW.

-Rick

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Karl Schultz on Tue, 08 Nov 2005 19:05:58 GMT
View Forum Message <> Reply to Message

On Tue, 08 Nov 2005 18:16:12 +0000, Dick Jackson wrote:

```
>  Now, this gives the images rendered nicely at first but has the odd
>  behaviour of IDLgrImages if you rotate the view. They don't rotate, they
>  stretch and squish. If you really want images that can rotate, you'll need
>  to make a polygon that contains each, something like this:
>
>  im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)))
>  im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)))
>  im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)))
>  poly1=Obj_New('IDLgrPolygon', [[-10,10],[0,10],[0,20],[-10,20]], $
>              Color=[255,255,255], Texture_Map=im1, $
>              Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
>  poly2=Obj_New('IDLgrPolygon', [[0,0],[10,0],[10,10],[0,10]], $
>              Color=[255,255,255], Texture_Map=im2, $
>              Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
>  poly3=Obj_New('IDLgrPolygon', [[10,10],[20,10],[20,20],[10,20]], $
>              Color=[255,255,255], Texture_Map=im3, $
>              Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
>  myModel = Obj_New('IDLgrModel')
>  myModel->Add, [poly1, poly2, poly3]
>  XObjView, myModel
```

I should point out that starting with IDL 6.2, IDL renders images using texture-mapped polygons, doing some of the steps above for you. Further, there is a new TRANSFORM_MODE property that will treat the image as a polygon during transforms, instead of just transforming the opposite corners and making a new 2D box from the new corner locations.

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Dick Jackson on Tue, 08 Nov 2005 20:34:55 GMT
View Forum Message <> Reply to Message

"Karl Schultz" <k____schultz@rsinc.com> wrote in message news:pan.2005.11.08.19.05.56.672000@rsinc.com...

> I should point out that starting with IDL 6.2, IDL renders images using
> texture-mapped polygons, doing some of the steps above for you. Further,
> there is a new TRANSFORM_MODE property that will treat the image as a
> polygon during transforms, instead of just transforming the opposite
> corners and making a new 2D box from the new corner locations.

Well, how about that. Thanks for the tip, Karl. My new example would then be:

```
im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10], $
      Transform_Mode=1)
im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)), $
      Transform_Mode=1)
im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)), Location=[10,10],
$
      Transform_Mode=1)
myModel = Obj_New('IDLgrModel')
myModel->Add, [im1, im2, im3]
XObjView, myModel
```

Another advantage of this is that we get away from the IDLgrPolygon texture map's required use of image sizes that are a power of two (the images were resampled to 16x16 in my previous example)

I see now that this whole issue is covered well in online help:

Programmer's Guides:
  Object Programming:
    Working with Image Objects:
      Positioning Image Objects in a View

Cheers,
--

-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Dick Jackson on Tue, 08 Nov 2005 22:00:45 GMT
View Forum Message <> Reply to Message

"Dick Jackson" <dick@d-jackson.com> wrote in message
news:PZ7cf.441815$1i.267001@pd7tw2no...
> "Karl Schultz" <k____schultz@rsinc.com> wrote in message
> news:pan.2005.11.08.19.05.56.672000@rsinc.com...
>
>> I should point out that starting with IDL 6.2, IDL renders images using
>> texture-mapped polygons, doing some of the steps above for you. Further,
>> there is a new TRANSFORM_MODE property that will treat the image as a
>> polygon during transforms, instead of just transforming the opposite
>> corners and making a new 2D box from the new corner locations.
>
> Well, how about that. Thanks for the tip, Karl. My new example would then
> be:
>
> im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10], $
> Transform_Mode=1)
> im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)), $
> Transform_Mode=1)
> im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)),
> Location=[10,10], $
> Transform_Mode=1)
> myModel = Obj_New('IDLgrModel')
> myModel->Add, [im1, im2, im3]
> XObjView, myModel
>
> Another advantage of this is that we get away from the IDLgrPolygon
> texture map's required use of image sizes that are a power of two (the
> images were resampled to 16x16 in my previous example)

Someone else at RSI has kindly cautioned me that the IDLgrImage won't give exactly the same behaviour as a texture-mapped IDLgrPolygon. The Image objects don't really "live" in the same 3-D space as Polygon objects. Rather, they are just drawn into the view in sequence along with the other objects, depending on the order they were added.

This example (images all at Z=0, polygons at Z=-10, 0 and 10) shows the rather bewildering appearance that mixing these objects in one view can

give:

```
im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)))
im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)))
im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)))
poly1=Obj_New('IDLgrPolygon',
[[-10,10,-10],[0,10,-10],[0,20,-10],[-10,20,-10]], $
        Color=[255,255,255], Texture_Map=im1, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
poly2=Obj_New('IDLgrPolygon', [[0,0],[10,0],[10,10],[0,10]], $
        Color=[255,255,255], Texture_Map=im2, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
poly3=Obj_New('IDLgrPolygon', [[10,10,10],[20,10,10],[20,20,10],[10,20,10]],
$
        Color=[255,255,255], Texture_Map=im3, $
        Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
im4=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10], $
    Transform_Mode=1)
im5=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)), $
    Transform_Mode=1)
im6=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)), Location=[10,10],
$
    Transform_Mode=1)
myModel = Obj_New('IDLgrModel')
myModel->Add, [poly1, poly2, poly3, im4, im5, im6] ; Images show up in front
;myModel->Add, [im4, im5, im6, poly1, poly2, poly3] ; Polygons show up in
front
XObjView, myModel
```

So, if you want images to look as if they are really in the same 3-D space
as other geometric objects, use texture-mapped IDLgrPolygons (or
IDLgrSurfaces, I suppose).

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Karl Schultz on Tue, 08 Nov 2005 22:39:21 GMT
View Forum Message <> Reply to Message

On Tue, 08 Nov 2005 22:00:45 +0000, Dick Jackson wrote:

> "Dick Jackson" <dick@d-jackson.com> wrote in message
> news:PZ7cf.441815$1i.267001@pd7tw2no...
>> "Karl Schultz" <k____schultz@rsinc.com> wrote in message
>> news:pan.2005.11.08.19.05.56.672000@rsinc.com...
>>
>>> I should point out that starting with IDL 6.2, IDL renders images using
>>> texture-mapped polygons, doing some of the steps above for you. Further,
>>> there is a new TRANSFORM_MODE property that will treat the image as a
>>> polygon during transforms, instead of just transforming the opposite
>>> corners and making a new 2D box from the new corner locations.
>>
>> Well, how about that. Thanks for the tip, Karl. My new example would then
>> be:
>>
>> im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10], $
>> Transform_Mode=1)
>> im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)), $
>> Transform_Mode=1)
>> im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)),
>> Location=[10,10], $
>> Transform_Mode=1)
>> myModel = Obj_New('IDLgrModel')
>> myModel->Add, [im1, im2, im3]
>> XObjView, myModel
>>
>> Another advantage of this is that we get away from the IDLgrPolygon
>> texture map's required use of image sizes that are a power of two (the
>> images were resampled to 16x16 in my previous example)
>
> Someone else at RSI has kindly cautioned me that the IDLgrImage won't give
> exactly the same behaviour as a texture-mapped IDLgrPolygon. The Image
> objects don't really "live" in the same 3-D space as Polygon objects.
> Rather, they are just drawn into the view in sequence along with the other
> objects, depending on the order they were added.
>
> This example (images all at Z=0, polygons at Z=-10, 0 and 10) shows the
> rather bewildering appearance that mixing these objects in one view can
> give:
>
> im1=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)))
> im2=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)))
> im3=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)))
> poly1=Obj_New('IDLgrPolygon',
> [[-10,10,-10],[0,10,-10],[0,20,-10],[-10,20,-10]], $
> Color=[255,255,255], Texture_Map=im1, $
> Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
> poly2=Obj_New('IDLgrPolygon', [[0,0],[10,0],[10,10],[0,10]], $
> Color=[255,255,255], Texture_Map=im2, $

```
>             Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
> poly3=Obj_New('IDLgrPolygon', [[10,10,10],[20,10,10],[20,20,10],[10,20,10]],
> $
>             Color=[255,255,255], Texture_Map=im3, $
>             Texture_Coord=[[0,0],[1,0],[1,1],[0,1]])
> im4=Obj_New('IDLgrImage', BytScl(BIndGen(3, 10, 10)), Location=[-10,10], $
>       Transform_Mode=1)
> im5=Obj_New('IDLgrImage', BytScl(RandomU(seed, 3, 10, 10)), $
>       Transform_Mode=1)
> im6=Obj_New('IDLgrImage', 255B-BytScl(BIndGen(3, 10, 10)), Location=[10,10],
> $
>       Transform_Mode=1)
> myModel = Obj_New('IDLgrModel')
> myModel->Add, [poly1, poly2, poly3, im4, im5, im6] ; Images show up in front
> ;myModel->Add, [im4, im5, im6, poly1, poly2, poly3] ; Polygons show up in
> front
> XObjView, myModel
>
> So, if you want images to look as if they are really in the same 3-D space
> as other geometric objects, use texture-mapped IDLgrPolygons (or
> IDLgrSurfaces, I suppose).
```

You could change the DEPTH_TEST_DISABLE property to make your images behave the same as other objects.

In the description for this property for IDLgrImage, we note:

"To preserve backward compatibility with previous versions of IDL the default value of this property is different from that of other graphic objects using this property."

For other objects, the default is OFF; for IDLgrImage it is ON, meaning that depth testing is disabled.

When we changed the image object to render as a texture-mapped polygon, we had to maintain the transform and depth-related behaviors with respect to IDL 6.1 for backwards compatibility.

That is, the "squishy" rotation and "painter's" depth buffer behavior had to be preserved, like them or not. So, the IDL 6.2 implementation turns off depth buffer processing by default when rendering an IDLgrImage, so that it "paints" the same way it used to, as a depth-agnostic pixel primitive. (Image primitives are often considered as 'pixel primitives' in graphics systems and are treated differently than geometric primitives.)

But if you want to go ahead and treat the image as a geometric primitive, set DEPTH_TEST_DISABLE to 0 to turn on depth testing and set TRANSFORM_MODE to 1 so that it transforms like a polygon.

And the LOCATION property can take 3 elements, so you can position the image in Z and have it sort itself out against other geometry with depth testing.

So, you should still be able to get what you want without resorting to texture mapping onto a polygon or surface yourself.

Karl

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Victor[1] on Tue, 08 Nov 2005 23:46:16 GMT
View Forum Message <> Reply to Message

Thanks alot to you all (David, Antonio, Karl, Dick, Rick) who took their time out explaining the concepts and giving the right pointers. Looks like we had a nice discussion that followed my initial inquiry (which by the way got very well addressed).

This message board is awesome, and there is all the expert help out here than you can ever ask for ! Looking forward to staying tuned. Thanks again to all.

- Victor

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by David Fanning on Wed, 09 Nov 2005 04:02:40 GMT
View Forum Message <> Reply to Message

Karl Schultz writes:

> I'm not sure why, but it seems that you have developed the idea
> that the POSITION keyword parameter has something to do with spatial
> locations.

Yes, when you start learning object graphics it really helps if you forget everything you ever thought you knew about direct graphics, including the definitions of all words. It just gets confusing otherwise. :-)

Cheers,

David

--

David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/

---

Subject: Re: Displaying three images simultaneously (using Object Graphics)
Posted by Dick Jackson on Wed, 09 Nov 2005 06:29:55 GMT
View Forum Message <> Reply to Message

"Karl Schultz" <k____schultz@rsinc.com> wrote in message
news:pan.2005.11.08.22.39.11.656000@rsinc.com...

> [...]
>
> So, you should still be able to get what you want without resorting to
> texture mapping onto a polygon or surface yourself.

Thanks for the thorough and patient explanation, Karl.

Cheers,
--
-Dick

Dick Jackson / dick@d-jackson.com
D-Jackson Software Consulting / http://www.d-jackson.com
Calgary, Alberta, Canada / +1-403-242-7398 / Fax: 241-7392

---