Subject: Re: texture_coord
Posted by David Fanning on Thu, 01 Nov 2001 15:18:56 GMT
View Forum Message <> Reply to Message

Harald von der Osten-Woldenburg (hvdosten@lb.netic.de) writes:

> maybe it's easier than I feel, but I don't came along with it:
>
> I would like to map a jpeg-file onto a small part of a 3D-surface. It
> works fine if I consider the entire surface. But this is not what I want
> to have. The problem seems to be the array texture_coord.
>
> If the surface is of an arry of - lets say - 1000 x 1200, how could I
> map a jpeg-file onto this surface with the subsurface-grid-coordinates
> [100, 50], [200,50], [200,300], [100,300]? And: I hope that the
> jpeg-file can have a higher resolution than [100 x 250] pixels for this
> example?
>
> Whatever I try - I get the error-message "number of vertices, normals,
> and texture coordinates do not match". Concerning to the online-help
> "TEXTURE_COORD property defines how individual data points within the
> image data are mapped...". I don't hope that each pixel in the jpeg-file
> must be referenced by corresponding coordinates....
>
> Thanks for each encouraging hint...

Oh, goody. Another simple problem. :-)

I don't know, Harald. I can't even add a texture
map to the surface at all without getting this
error:

```
  thisImage = Obj_New('IDLgrImage', image)
  thisSurface = OBJ_NEW('IDLgrSurface', data, x, y, Style=2, $
    Color=[255,255,255], Texture_Map=thisImage, $
      Texture_Coord=[[0,0], [1,0], [1,1], [0,1], [0,0]]

  IDL>Simple_Surface, Image=myimage
   % IDLGRSRCDEST::DRAW: Error, numbers of vertices, normals, and
     texture coordinates do not match.
```

Although the image *does* show up on the surface, oddly enough. But
changing the values of the texture coordinates does absolutely
nothing with respect to mapping the image on the surface. It always
covers the entire surface. I'm using IDL 5.5.

I've got work to do. I think you are going to have to handle
this one yourself. :-)

Cheers,

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

Subject: Re: texture_coord
Posted by Karl Schultz on Thu, 01 Nov 2001 16:11:32 GMT
View Forum Message <> Reply to Message

"Harald von der Osten-Woldenburg" <hvdosten@lb.netic.de> wrote in message news:3BE13E28.B452973E@lb.netic.de...
> Hi,
>
> maybe it's easier than I feel, but I don't came along with it:

It isn't that bad, but it is sort of hard to explain. Code helps - see the bottom of this post.

> I would like to map a jpeg-file onto a small part of a 3D-surface. It
> works fine if I consider the entire surface. But this is not what I want
> to have. The problem seems to be the array texture_coord.

Yes, using TEXTURE_COORD is how you do this.

> If the surface is of an arry of - lets say - 1000 x 1200, how could I
> map a jpeg-file onto this surface with the subsurface-grid-coordinates
> [100, 50], [200,50], [200,300], [100,300]? And: I hope that the
> jpeg-file can have a higher resolution than [100 x 250] pixels for this
> example?

The image (jpeg file) can be any resolution you want. That is part of the utility of texture mapping.

> Whatever I try - I get the error-message "number of vertices, normals,
> and texture coordinates do not match". Concerning to the online-help
> "TEXTURE_COORD property defines how individual data points within the
> image data are mapped...". I don't hope that each pixel in the jpeg-file
> must be referenced by corresponding coordinates....

Yes, you do need to specify a texture coordinate for each vertex in your polygon or surface. Otherwise, IDL won't know how to map your image to the

geometry. I know that it "seems" obvious in a lot of cases, especially when dealing with a specialzation of a polygon such as the surface. It seems like you'd just want to linearly interpolate the image across the surface, and that is what we do for the default case. But you need to be fully explict with anything other than that.

Here is a real simple program you can try that maps an IDL logo onto a subrect in the middle of a surface. Loop-haters can probably figure out how to get rid of the loop, but I hope that what I have below is clear enough to understand easily.

Hope that this helps,

Karl

```
pro texture

filename = FILEPATH('examples.tif', SUBDIRECTORY=['examples','data'])
imageData = READ_TIFF(filename, R, G, B)
imageData = REVERSE(imageData,2)
oPalette = OBJ_NEW('IDLgrPalette', R, G, B)
oImage = OBJ_NEW('IDLgrImage', imageData, PALETTE=oPalette)
texCoords = FLTARR(2,100,120)
; subrect is [20,30] to [80,90]
FOR y=30, 90 DO BEGIN
   texCoords[0, 20:80, y] = FINDGEN(61)/ 60
   texCoords[1, 20:80, y] = (y-30) / 60.0
ENDFOR
oSurface = OBJ_NEW('IDLgrSurface', dist(100,120), $
   COLOR=[255,255,255], STYLE=2, TEXTURE_MAP=oImage,
TEXTURE_COORD=texCoords)

xobjview, oSurface
end
```

Subject: Re: texture_coord
Posted by David Fanning on Thu, 01 Nov 2001 17:53:23 GMT
View Forum Message <> Reply to Message

Karl Schultz writes:

> It isn't that bad, but it is sort of hard to explain. Code helps - see the
> bottom of this post.

OK, thanks, Karl. That certainly wasn't what the
documentation implied, but what else is new. :-)

Here is the deal. To map an image onto the *entire*
surface, this code works. (Note that Scale_Vector
is a function found on my web page. You can scale
the array however you like. It must be scaled
into the range of 0 to 1.)

```
  ss = Size(surfaceData, /Dimensions)
  texcoords = FltArr(2, ss[0], ss[1])
  texcoords[0,*,*] = Scale_Vector(Findgen(ss[0]) # $
     Replicate(1,ss[1]), 0.0, 1.0)
  texcoords[1,*,*] = Scale_Vector(Replicate(1,ss[1]) #
     Findgen(ss[0]), 0.0, 1.0)
```

To place the image on a portion of the surface,
you have to modify the values appropriately. Here my
surface is an array of (200,100) in size and I wish
to place the image with the lower-left corner at (40,30)
and the upper-right corner at (129,59). Thus the image
will occupy a 90x30 "chunk" of my surface.

```
  texcoords = FltArr(2, ss[0], ss[1])
  texcoords[0,40:129,30:59] = Scale_Vector(Findgen(90) # $
     Replicate(1,30), 0.0, 1.0)
  texcoords[1,40:129,30:59] = Scale_Vector(Replicate(1,30) # $
    Findgen(90), 0.0, 1.0)
```

In either case, the surface is created like this:

```
  thisSurface = OBJ_NEW('IDLgrSurface', surfaceData, Style=2, $
    Color=[255,255,255], Texture_Map=thisImage, $
    Texture_Coord=texcoords
```

I'll write up an article about this later today.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: texture_coord
Posted by David Fanning on Thu, 01 Nov 2001 17:57:58 GMT

David Fanning (david@dfanning.com) writes:

> Here is the deal.

By the way, if it is resolution you are after you
will have to increase the resolution of your surface,
NOT your image! What you are mapping is little tiny
polygons. The more polygons you have, the better the
image looks on the surface. It will look chunky if
the surface is 40x40. It will look nice if the
surface is 400x400.

(I suppose you can increase the resolution in
your texture coordinates, too. But you get the
idea.)

Cheers,

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

Subject: Re: texture_coord
Posted by Harald von der Osten-[1] on Thu, 01 Nov 2001 18:48:45 GMT

Hi,

thanks a lot to David and Karl. Now it works!!!!

Harald

--
Harald von der Osten-Woldenburg
Geophysical Prospection of Archaeological Sites
Landesdenkmalamt Baden-Wuerttemberg

Silberburgstrasse 193
D-70178 Stuttgart
Fax Office: +49-(0)711-1694-707
Fax Private: +49-(0)180 50 52 55 22 10 05
http://www.lb.netic.de/hvdosten : Geomagnetics, Geoelectrics, Radar, EMI

Subject: Re: texture_coord
Posted by David Fanning on Fri, 02 Nov 2001 00:34:40 GMT
View Forum Message <> Reply to Message

Harald von der Osten-Woldenburg (hvdosten@lb.netic.de) writes:

> Thanks a lot to David and Karl. Now it works!!!!

So we all remember how to do it, I've added an article to my web page:

http://www.dfanning.com/ographics_tips/imgtex.html

I've also cobbled together an example program:

http://www.dfanning.com/programs/texture_surface.pro

To see how it works:

IDL> Texture_Surface

To call it with your own surface data and image, do something like this:

IDL> Texture_Surface, surfaceData, Image=myimage

I wrote the program so you could position the image at some coordinate location on the surface. So, for example, if your surface is a 41 by 41 array and you wish to put the image with its lower-left corner at (5,10) and its upper-right corner at (25,18) with respect to this surface. Then you can call the program like this:

IDL> Texture_Surface, Position=[5, 10, 25, 18]

Note that the article talks about a couple of unresolved issues. First, when I position the image as above, I don't seem to have control over what color the *rest* of the surface is. Second, the positioned image seems to have problems around its edges. I suspect both of these problems

may be related, but so far I have made no progress resolving
them. I'm open to any and all ideas.

Oh, by the way, I think I was wrong about the resolution
of the surface. Making the surface bigger does not seem
to affect the resolution of the image on the surface
at all.

Cheers,

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

Subject: Re: texture_coord
Posted by Mark Hadfield on Fri, 02 Nov 2001 01:07:11 GMT
View Forum Message <> Reply to Message

From: "David Fanning" <david@dfanning.com>
> So we all remember how to do it, I've added an article to
> my web page:
>
> http://www.dfanning.com/ographics_tips/imgtex.html
>
> I've also cobbled together an example program:
>
> http://www.dfanning.com/programs/texture_surface.pro

Great work, David.

> Note that the article talks about a couple of unresolved
> issues. First, when I position the image as above, I don't
> seem to have control over what color the *rest* of the
> surface is.

I see that in your screenshots it is yellow. On my system it is dull green.
Curious!

> Second, the positioned image seems to have
> problems around its edges.

Your test image ('rose.jpg') is 227 x 149 pixels. Texture-map image
dimensions are supposed to be a power of 2 (or so I was informed some time

ago when I complained to RSI about misalignment problems). As I understand
it, this is an OpenGL restriction.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

--
Posted from clam.niwa.cri.nz [202.36.29.1]
via Mailgate.ORG Server - http://www.Mailgate.ORG

Subject: Re: texture_coord
Posted by David Fanning on Fri, 02 Nov 2001 01:31:28 GMT
View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.cri.nz) writes:

> I see that in your screenshots it is yellow. On my system it is dull green.
> Curious!

The color of the rest of the surface is somehow
related to the colors used in the image, although
in a way I haven't discovered as yet. I've tried
to tweak corner pixels, etc. But I haven't found
the magic pixel yet. (Any Diablo players out there?)

>> Second, the positioned image seems to have
>> problems around its edges.
>
> Your test image ('rose.jpg') is 227 x 149 pixels. Texture-map image
> dimensions are supposed to be a power of 2 (or so I was informed some time
> ago when I complained to RSI about misalignment problems). As I understand
> it, this is an OpenGL restriction.

What does that mean? I am suppose to resize
my image before I put it on the surface? Or
that I should only put it on surfaces that have
a power of 2 size? (This actually would make
more sense to me, but I don't think it's what
you mean.) Resizing images might have serious
aspect ratio repercussions. :-)

Cheers,

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

Subject: Re: texture_coord
Posted by Mark Hadfield on Fri, 02 Nov 2001 02:48:49 GMT
View Forum Message <> Reply to Message

From: "David Fanning" <david@dfanning.com>
> Mark Hadfield (m.hadfield@niwa.cri.nz) writes:
>
>> Your test image ('rose.jpg') is 227 x 149 pixels. Texture-map image
>> dimensions are supposed to be a power of 2 (or so I was informed some
time
>> ago when I complained to RSI about misalignment problems). As I
understand
>> it, this is an OpenGL restriction.
>
> What does that mean? I am suppose to resize
> my image before I put it on the surface?

Yep.

Also note that the maximum size that an OpenGL implementation is required to support is only 64 x 64. (Though most support more; the limit for any destination object can be found using the GetDeviceInfo method, keyword MAX_TEXTURE_DIMENSIONS.) The new TEXTURE_HIGHRES keyword in 5.5 lets you exceed MAX_TEXTURE_DIMENSIONS, but it is incompatible with TEXTURE_COORD. Aaaaaaarrrrrgggghhh!!

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

--

Subject: Re: texture_coord
Posted by David Fanning on Fri, 02 Nov 2001 15:09:45 GMT
View Forum Message <> Reply to Message

Mark Hadfield (m.hadfield@niwa.cri.nz) writes:

> Your test image ('rose.jpg') is 227 x 149 pixels. Texture-map image
> dimensions are supposed to be a power of 2 (or so I was informed some time
> ago when I complained to RSI about misalignment problems). As I understand
> it, this is an OpenGL restriction.

I hear from the usual reliable source that IDL will
automatically resize your input texture image
to a power of two, so you do not need to do this
yourself. It will also down-sample the image to
the maximum supported texture resolution on your
hardware automatically.

I thought something like this was happening,
because it really does do a fairly nice job
when you texture the entire surface.

Cheers,

David

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

Subject: Re: texture_coord
Posted by Karl Schultz on Fri, 02 Nov 2001 16:39:18 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.164b980a9deb6a55989744@news.frii.com...
> Note that the article talks about a couple of unresolved
> issues. First, when I position the image as above, I don't
> seem to have control over what color the *rest* of the
> surface is.

Yes, that's a little hard in this context. In fact, my difficulty in
explaining this may suggest that we need to add something to IDL to make
this easier. Some discussion may help.

At least with the code posted in this thread, we were just letting the "unused" texture coords stay (0,0), which means that the color on the rest of the surface would take on the color of the (0,0) texel, whatever that is. See next topic.

> Second, the positioned image seems to have
> problems around its edges. I suspect both of these problems
> may be related, but so far I have made no progress resolving
> them. I'm open to any and all ideas.

Right. I noticed this too. There's some interpolation going on along the edge of the surface. One thing to keep in mind is that there isn't always an one-to-one relationship between texels and pixels. It may take several texels to decide what color to make a pixel, as is the case where the texture has a higher sampling than the screen. Or, a single texel may be used to determine the color of many pixels in the opposite case. You might be seeing texel (0,0) and the texel from an edge of the texture being combined to determine the color of a pixel along the edge. This can lead to somewhat random-looking results.

In fact, the texel interpolation is a whole lot worse than this. Suppose that you are mapping a texture onto a sub-surface with corners [20,20] and [30,30]. The texture coordinate at [20,20] is [0,0]. The texture coordinate at [20,19] is also [0,0], so there is no real problem there. But if you look at the texture coordinate at [28,20] and at [28,19] we see that two adjacent texture coords are something like [0.9,0.0] and [0.0,0.0]. This is really bad because OpenGL will take all the texels (from the image) between (normallized) [0.9,0.0] and [0,0], average them together, and then use that color value as one of the colors used to decide the color of the pixel in that area. It gets a lot worse if you go over to [30,30] where we would average all the image texels in a diagonal line across the image. Yuk. OpenGL is doing what we tell it to, but not what we want.

OpenGL has a LOT of facilities in its texture mapping support to control border issues, which indicates to me that it is not a simple problem. IDL doesn't expose all these controls.

One step in attacking the problem is to pre-process your image to put a border around it. Make the color whatever you'd like the "rest" of the surface to look like. And you might try it with one-pixel borders, and perhaps two or three. I extended your program (texture_surface) to do this and I got a nice black background since I made my borders black.

But this didn't completely solve the problem. The left and the bottom borders look ok - black. But the top and right edges have smudged up colors where the black border should be. This is caused by the texel interpolation across the image I mentioned above. How to fix this? More work. The

texture coords of the vertices ADJACENT to the area where the texture is mapped need to be something other than [0,0]. Using the above example again, the texture coordinates at [28,20] should be [0.9,0.0] and the texture coordinates at [28,19] need to be [0.9,0.0] as well. This will cause the texels accessed from the image to be the same (the new border texels in this case) and we should get black. In fact, it may make sense to set the texture coords at [28,0:19] to [0.9,0.0]. This avoids the [0.9,0.0]->[0.0,0.0] texel interpolation across the image.

I also tried this with your program and got pretty encouraging results, although I didn't implement a full, general solution. Maybe I'll work on it.

The bottom-line is that we were getting lazy by not setting the texture coordinates of the vertices of the "rest" of the surface to reasonable values. This caused a major discontinuity in the texture interpolation.

>
> Oh, by the way, I think I was wrong about the resolution
> of the surface. Making the surface bigger does not seem
> to affect the resolution of the image on the surface
> at all.

I almost posted about this topic yesterday. Right, the number of polygons (facets) in the surface won't have an effect on the appearance of the image. The texture image is interpolated across each facet, using the texture coords of the facet to determine what part of the image is used. If you generate more facets, you will have smaller steps in the texture coordinates across the facets and you end up with the same thing. If you wanted to do something other than a linear texture mapping, like some sort of morphing, then you might want more facets to give you more control.

You may see some difference in a more geometric sense. For example, if you have an implicit surface (generated by some function) that is pretty curvy, you'll get a better looking and more accurate surface as you increase the number of facets, texture or no texture. For simpler surfaces, fewer facets suffice, textured or not. People often map textures onto 4-vertex planar polygons or surfaces so that they can display an image in a more flexible way. You can't really manipluate an IDLgrImage very well with all the model transforms, so if you wanted to display an image with an arbitrary transform, you can map it onto a simple polygon and orient it however you want. You also gain a lot of functionality in the areas of stretching and transparency. Anyway, one facet is enough if all you want is a flat surface. The image texels are interpolated across the single facet.

Subject: Re: texture_coord

Posted by Karl Schultz on Fri, 02 Nov 2001 16:46:50 GMT
View Forum Message <> Reply to Message

"Mark Hadfield" <m.hadfield@niwa.cri.nz> wrote in message
news:001301c16348$e4736f10$d938a8c0@Hadfield...
> From: "David Fanning" <david@dfanning.com>
>> Mark Hadfield (m.hadfield@niwa.cri.nz) writes:
>>
>>> Your test image ('rose.jpg') is 227 x 149 pixels. Texture-map image
>>> dimensions are supposed to be a power of 2 (or so I was informed some
> time
>>> ago when I complained to RSI about misalignment problems). As I
> understand
>>> it, this is an OpenGL restriction.
>>
>> What does that mean? I am suppose to resize
>> my image before I put it on the surface?
>
> Yep.
>

IDL will scale it if you don't. But that scaling may not have been what you
wanted, and may have caused the misalignment problems?

Subject: Re: texture_coord
Posted by David Fanning on Fri, 02 Nov 2001 17:04:22 GMT
View Forum Message <> Reply to Message

Karl Schultz (kschultz@researchsystems.com) writes:

> Some discussion may help.

I really appreciate this information, Karl. Thank you.
I look forward to seeing the "general" solution. :-)

I'd like to work on it some more, too. But I
really have to get back to work. My customers
think I have completely wigged out. :-(

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438, E-mail: david@dfanning.com

Subject: Re: texture_coord
Posted by Pavel A. Romashkin on Fri, 02 Nov 2001 17:16:04 GMT
View Forum Message <> Reply to Message

As this thread became more and more technical, I am beginning to wonder:
isn't it easier to just create a temporary blank image that matches the
dimensions of the entire surface, then insert the sub-image into it,
then map it over the whole surface? Or does this just take all the fun
out of it?

Pavel

Harald von der Osten-Woldenburg wrote:

> I would like to map a jpeg-file onto a small part of a 3D-surface. It
> works fine if I consider the entire surface. But this is not what I want
> to have. The problem seems to be the array texture_coord.c

Subject: Re: texture_coord
Posted by David Fanning on Fri, 02 Nov 2001 17:30:07 GMT
View Forum Message <> Reply to Message

Pavel A. Romashkin (pavel.romashkin@noaa.gov) writes:

> As this thread became more and more technical, I am beginning to wonder:
> isn't it easier to just create a temporary blank image that matches the
> dimensions of the entire surface, then insert the sub-image into it,
> then map it over the whole surface? Or does this just take all the fun
> out of it?

While this might appear to be an easier solution,
it not only takes all the fun out of a hyper-technical
programming operation, it introduces a whole host of
new problems that have to be solved. For one, reducing
your image to the resolution of the surface will make
that image awfully ugly. (I can't really think of a
way to solve this limitation, to tell you the truth.)

Another problem would be matching the image to a
particular location on the surface. This, presumably,
is the whole point of putting the image on there in
the first place. Getting the correspondence correct

would be an awfully tedious process, it seems to me,
and could only really be accomplished easily if the
surface and image data had the same aspect ratio.
Pretty rare in practice, I think.

Cheers,

David

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

Subject: Re: texture_coord
Posted by Karl Schultz on Fri, 02 Nov 2001 17:49:10 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.164c860d201596a9989748@news.frii.com...
> Pavel A. Romashkin (pavel.romashkin@noaa.gov) writes:
>
>> As this thread became more and more technical, I am beginning to wonder:
>> isn't it easier to just create a temporary blank image that matches the
>> dimensions of the entire surface, then insert the sub-image into it,
>> then map it over the whole surface? Or does this just take all the fun
>> out of it?
>
> While this might appear to be an easier solution,
> it not only takes all the fun out of a hyper-technical
> programming operation, it introduces a whole host of
> new problems that have to be solved. For one, reducing
> your image to the resolution of the surface will make
> that image awfully ugly. (I can't really think of a
> way to solve this limitation, to tell you the truth.)
>
> Another problem would be matching the image to a
> particular location on the surface. This, presumably,
> is the whole point of putting the image on there in
> the first place. Getting the correspondence correct
> would be an awfully tedious process, it seems to me,
> and could only really be accomplished easily if the
> surface and image data had the same aspect ratio.
> Pretty rare in practice, I think.

Another thought I had was to put the sub-image on its own fully-textured polygon and then draw that polygon on top of the polygon that you want the image to be a sub-image of. Then you run into z-buffer stitching problems, which can be partly or fully addressed by the DEPTH_OFFSET property. But that's no fun either. I sent David an updated program that fixes the problem by fully specifying the texture coordinates. It wasn't so bad..

Subject: Re: texture_coord
Posted by David Fanning on Sun, 04 Nov 2001 16:53:15 GMT
View Forum Message <> Reply to Message

Karl Schultz (kschultz@researchsystems.com) writes:

> I sent David an updated program that fixes the
> problem by fully specifying the texture coordinates.
> It wasn't so bad.

Indeed. I made a couple of modifications to the program to make it more general. And I found that a one-pixel border (instead of two) around the image was enough to be able to specify a surface color. (I worry whether this might be OpenGL-implementation specific, but it definitely works on my machine in every instance I've tried it.)

You can find the final program here:

http://www.dfanning.com/programs/texture_surface.pro

Try this:

```
IDL> Texture_Surface
```

Or, to position the image on the surface and color the rest of the surface a light gray color:

```
IDL> Texture_Surface, Position=[10, 5, 35, 30], $
       BorderColor=[185, 185, 185]
```

An article that explains the details of this can be found here:

http://www.dfanning.com/ographics_tips/imgtex.html

Cheers,

David

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

Subject: Re: texture_coord
Posted by Mark Hadfield on Sun, 04 Nov 2001 21:23:40 GMT
View Forum Message <> Reply to Message

From: "Karl Schultz" <kschultz@researchsystems.com>
>>> Mark Hadfield (m.hadfield@niwa.cri.nz) writes:
>>>> Your test image ('rose.jpg') is 227 x 149 pixels.
>>>> Texture-map image dimensions are supposed to be a
>>>> power of 2 (or so I was informed some time ago
>>>> when I complained to RSI about misalignment problems).
> IDL will scale it if you don't. But that scaling may not have been
> what you wanted, and may have caused the misalignment problems?

Yes. I realise now this has nothing to do with the interesting effects encountered by David. But IDL's automatic re-sizing of images when it needs to use them as a texture map is unsatisfactory, in my opinion. Below is a test program to illustrate this. It creates & displays an image on its own and then as a texture-map. If it is run with argument "num" set to any number that is not a power of 2 the texture-mapped image is shifted up & to the right.

Actually, I've found IDL's image-oriented mathematical operations (interpolation, ROIs) often have pretty dodgy geometry when used with a small number of pixels.

Mark Hadfield
m.hadfield@niwa.cri.nz http://katipo.niwa.cri.nz/~hadfield
National Institute for Water and Atmospheric Research

```
pro mgh_test_image2, NUM=num

   compile_opt IDL2

   if n_elements(num) eq 0 then num = 16

   if n_elements(option) eq 0 then option = 0

   ; Create a symmetrical 2D array
```

```
    thedata = bytarr(num,num)
    thedata[0:num-2,0:num-2] = bytscl(dist(num-1))
    thedata[num-1,*] = thedata[num-2,*]
    thedata[*,num-1] = thedata[*,num-2]

    ; Load the data into an IDLgrImage

    theimage = obj_new('IDLgrImage', thedata, LOCATION=[0,0] $
          , DIMENSIONS=[1,1])

    ; Display the image alone

    xobjview, theimage

    ; Display the image mapped onto an IDLgrPolygon

    xobjview, obj_new('IDLgrPolygon', [0,1,1,0], [0,0,1,1] $
          , COLOR=[255,255,255], TEXTURE_MAP=theimage $
          , TEXTURE_COORD=[[0,0],[1,0],[1,1],[0,1]])

end
```