Subject: Re: Polygons and polylines
Posted by Rick Towler on Wed, 26 Apr 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Struan Gray wrote:
>
> Steven Chetelat, chetelat@csee.usf.edu writes:
>
>> Does the polygon object allow
>> for transparency?
>
> Investigate the use of a 'texture map': an image object which
> gives colour to the polygon faces and overrides (well, interacts with)
> the colours given to the polygon vertices. If the image has a fourth,
> 'alpha', channel the values in that channel determine the degree of
> transparency. I think David F. has an example of how to do this on
> his website (www.dfanning.com).

I have struck out with a couple of my replies so take this as a suggestion only. I am trying.....

Struan is on the right path. I just had to figure this one out and here are some hints. I make no claims that they are the *best* hints but they'll get you somewhere:

create an image object with an alpha channel. Note that your image data can be created manually too.

```
;you can read one from an existing file
READ_JPEG,file ,idata, TRUE=3
imsize=SIZE(idata, /DIMENSIONS)
;add the alpha channel - a 4th "color" channel
;create the new larger image array
idata2=FLTARR(imsize(0),imsize(1),imsize(2)+1)
;copy the original image data to it
idata2(*,*,0:2)=idata
;set alpha to the value of my opacity slider (values from 0..255)
WIDGET_CONTROL, state.wOpacityS, GET_VALUE=alpha
idata2(*,*,3)=alpha

state.oPhoto = OBJ_NEW('IDLgrImage', idata2, INTERLEAVE=2, $
      BLEND_FUNCTION=[3,4], /INTERPOLATE)
```

Or create it manually. Here is a 50x50 pixel pure white solid (alpha channel set to 255) image object:

```
state.oImage = OBJ_NEW('IDLgrImage', REPLICATE(255,50,50,4),
INTERLEAVE=2, $
      BLEND_FUNCTION=[3,4], /INTERPOLATE)
```

Notice the BLEND_FUNCTION. Values of 3,4 do what your looking for but take a better look at the documentation on this. I personally didn't get the whole picture from the manual and I stopped trying when I got things to work the way I wanted them to.

Now create a polygon with the texture mapped image object.

```
state.oFBSurface = OBJ_NEW("IDLgrPolygon", x,y,z, $
      SHADING=1,  COLOR=body_color, POLY=fb_mesh, $
      TEXTURE_MAP=state.oImage, STYLE=2, $ /TEXTURE_INTERP, $
      TEXTURE_COORD=TRANSPOSE([[ynorm],[znorm]]), $
      REJECT=0, ZERO_OPACITY_SKIP=0)
```

You can see that I set TEXTURE_MAP to the image object I create above. You only need to set TEXTURE_INTERP if your going to be using detailed images and want the better image quality. TEXTURE_COORD is required and is set to an array of normailzed vertex values and is used to map the image object to the polygon object.

Lastly, order your objects from back to front so they are displayed properly.
something like:

```
oModelTop -> ADD, myPolyline
oModelTop -> ADD, mySurface
```

To change opacity of your polygon just change the alpha channel values of your image object. I have a slider widget and can pick up the new value of my alpha channel from that (as ev.value)

```
state.oImage->getProperty, Data=image
image(*,*,3)=ev.value
state.oImage->setProperty, Data=image
```

Hope this helps.

-Rick Towler

>
> You also need to pay attention to the order in which objects are
> drawn, with the frontmost, transparent object drawn last. You control
> the drawing order by changing the ordering of the objects in their
> enclosing model or view. I don't know if this works with two
> intersecting transparent surfaces, but I suspect that only one will be
> drawn properly.
>
> Struan
