Subject: Re: boxgrid isurface
Posted by Jim Pendleton, ITT Vi on Tue, 19 Feb 2008 21:55:50 GMT
View Forum Message <> Reply to Message

"David Fanning" <news@dfanning.com> wrote in message
news:MPG.2224e1ce7d1d19dd9896bd@news.frii.com...
> jochem.verelst@gmail.com writes:
>
>> I was wondering, is there a way to realize in ISURFACE a box gridding
>> representing the x-xis,y-axis and z-axis for those 'walls' of the data
>> space that are located behind the shown data. So the grid would
>> typically cover the two back walls and the bottom of the data space.
>> For instance similar to how it is done in Matlab (e.g. see:
>> http://www.agnld.uni-potsdam.de/~marwan/matlab-tutorials/htm l/basics_16.png)
>
> Not an iSurface plot, but you could try Plot_3DBox
> if you get desperate. :-)
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming (www.dfanning.com)
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

There are a couple ways to accomplish this, more or less.

First you can change the properties of the Axes objects in the iSurface Visualization Browser. Set the Line style and major and minor tick lengths, for example, to create a grid.

However, that may not be enough control for a power user.

If you're up for an interesting challenge, you can always add your own model trees into an iTools hierarchy. It just depends on how willing you are to accept the fact that iTools doesn't actually "know" about them! For many purposes, hacking into iTools' graphics makes sense, for example to produce final printed output.

In the example of a surface, we can determine through OBJ_VALID() and OBJ_ISA() you can find the reference to the IDLgrSurface object.

```
IDL> isurface, dist(30)
IDL> allobjects = obj_valid(/cast)
IDL> osurface = (allobjects[where(obj_isa(allobjects, 'idlgrsurface'))])[0]
```

Knowing that, you can find the parent object, which will be a model.

IDL> osurface->getproperty, parent = oparent

Now you can add your own graphics hierarchy, for example

```
IDL> omodel = obj_new('idlgrmodel')
IDL> oparent->add, omodel
IDL> opolygon = obj_new('idlgrpolygon', [0, 1, 1, 0]*29, [0, 0, 1, 1]*29, $
IDL>    replicate(1.e-3, 4), style = 2, color = [255, 0, 0])
IDL> omodel->add, opolygon
```

The next time the iTool is updated (such as after an expose event), you'll have a
red rectangle in the Z ~= 0 plane. Making a grid as shown with the MATLAB output wouldn't be too much more work with a few IDLgrPolyline objects, or perhaps the addition of more IDLgrAxis objects set up the exact way you want them.

Of course to make your added visualization objects accessible "properly" through the "Visualization Browser" window, you'd need to add these items "the iTools way", but that's left as an exercise for the reader.

Jim P.
ITT VIS
