Subject: Re: access to vercities & connectivity data
Posted by Rick Towler on Wed, 12 Nov 2003 18:01:51 GMT

"Neil" wrote...
> "Rick Towler" wrote ...
>> "Neil" wrote...
>>> I would like to read in a DXF file and get access to the numerical
>>> values of the vertices and the connectivity. That's all i wish to do,
>>> which IDL routines should i use for this task. From the IDL help it is
>>> not immediately obvious how this can be done.

>> There are a couple of ways:
>>
>> You can use the IDLffDXF object. There is an example close to what you
are
>> looking for in in the docs under IDLffDXF::Read. Also of interest will
be
>> IDLffDXF::GetEntity which describes how to pull out the different entity
>> types. The types are described in the IDLffDXF::GetContents docs.

> Thank you Rick for you comments. Well i decided on option 1 of yours
> from above.

> This seems to be getting part of the way, the information looks the
> right stuff, but i dont know if all the routine calls are necessary.
>
> However, i do not yet seem to be able to access the real numberical
> values of the "verticies" and "connectivities". I can see that
> tissues.verticies and tissue.connectivity are pointers, but to what?
> How do i get the one dimensional array with all the connectivity and
> the 3 by n array of verticies? Which variable holds the data?

Neil,

You are sooooo close. You probably have already figured this out but try:

```
print, *tissue.connectivity
print, *tissue.vertices
```

As you discovered, the GetEntity method returns a structure which contains 3
pointers. Without really getting into it, pointers in IDL are lightweight
tokens that "point to" some data. The pointer variable only contains a
reference to that data which is what you have discovered. What you need to

do is to dereference your pointer using the "*" character.

Pointers are persistent heap variables. That means that they will exist until you explicitly destroy them or exit IDL. What may not be obvious is that when using the IDLffDXF object the pointers that are returned to you are your responsibility. You must keep track of them and free them using PTR_FREE() when you are finished. Something as simple as the following would work:

```
tissue = oTest -> GetEntity(TestTypes[1])

verts = *tissue.vertices
conn = *tissue.connectivity

PTR_FREE, [tissue.vertices, tissue.connectivity, $
   tissue.vertex_colors]
```

-Rick