Subject: DXF and Face3D
Posted by Thomas Launey on Wed, 30 Aug 2006 14:03:42 GMT
View Forum Message <> Reply to Message

Hello,
I am trying to smooth all the mesh (Face3d) objects in a DXF file using
the code below.
It fails however when I try to put the modified vertices back into the
DXF object with the message:
IDL> test_IDLffDXF
% Loaded DLM: DXF.
% IDLFFDXF::PUTENTITY: DXF error: FACE3D connectivity list must contain
only 4-vertex polygons.
% Execution halted at: TEST_IDLFFDXF 18 H:\test_idlffdxf.pro

```
Pro test_IDLffDXF
DXF_file=dialog_pickfile(filter="*.DXF", /must_exist)
 If N_elements(DXF_file) LT 1 then return
oDXF = OBJ_NEW('IDLffDXF')
status = oDXF->Read(DXF_file)
DXFTypes = oDXF->GetContents()
;*** get all face3D entities
DXFsurface = oDXF->GetEntity(DXFTypes[where(DXFTypes EQ 10)])
vertices = DXFsurface.vertices
connectivity = DXFsurface.connectivity
N_object=n_elements(vertices)
For i=0, N_object-1 do Begin
 vert=MESH_SMOOTH (*vertices[i], *connectivity[i])
 *((DXFsurface.vertices)[i])=vert
EndFor
oDXF->PutEntity,DXFsurface
End
```

It seems that all I am doing is to move around the vertices, without
changing the number of faces. A post by K. Schultz from June 2004
suggests to change the type to 9 (polygon) but also warn of potential
problems with running vertices through the tessellator.
Any help would be greatly appreciated :-)
Thanks,
Thomas

Subject: Re: DXF and Face3D
Posted by Karl Schultz on Thu, 31 Aug 2006 14:56:59 GMT
View Forum Message <> Reply to Message

If your original input data was in the form of an implicit quad mesh, then
you might be able to write it to DXF as a quad mesh instead of a polygon

with a connectivity list.

MESH_SMOOTH just moves the vertices around and returns the modified vertex list without changing the connectivity. (I was a bit wrong about this in my last posting) So, you should be able to submit the new vertex list to DXF, since the shape of the mesh hasn't changed and you don't need a connectivity list. Use the POLYGON type and specify the MESH_DIMS instead of the connectivity list. Again, this only works if your original data can be specified as an implicit quad mesh.

If you are dealing with general polygons with connectivity data, then you might just want to try it with the POLYGON enitity with connectivity list.

My reading of the problem description is that IDL ran the polygon through the tesselator when it did not need to. This resulted in storing a polygon in the DXF file that had a different vertex list and connectivity list, as compared to the input. But, depending on the input, the mesh stored in the DXF file could be equivalent to the input mesh. If the input mesh had things in it like overlapping faces, self-intersections, or holes, the mesh stored in the DXF file might not be topologically equivalent to the input mesh.

In other words, if your mesh is simple or regular, you might get away with it.

Otherwise, you are probably out of luck if you do not upgrade to 6.2. The fix for 6.2 was pretty involved and I don't think that there are any other sneaky workarounds.

Karl

On Wed, 30 Aug 2006 22:24:25 -0700, Thomas Launey wrote:

> Dear Karl,
> Thank you very much for the update on 6.2. Unfortunately I am still on
> 6.1, with no prospect of upgrade. Any suggestion for fixing the problem
> on 6.1?
> Thanks
> Thomas
>
> Karl Schultz wrote:
>>
>> You should probably go ahead and use the type 9 entity. I fixed the
>> problem that you mentioned that I mentioned in IDL 6.2.
>>
>> I think that the MESH_SMOOTH algorithm ends up treating your mesh as a
>> triangle mesh, and so outputs it as such.

>>
>> If you supply a vertex list and a connectivity list to a type 9 DXF
>> entity, IDL will examine the connectivity list and NOT call the tessellator
>> if all polygons in the mesh are triangles (as of IDL 6.2).
>>
>> Karl

---

Subject: Re: DXF and Face3D
Posted by Thomas Launey on Tue, 05 Sep 2006 00:52:32 GMT
View Forum Message <> Reply to Message

Dear Karl,

I really needed to keep the Face3D entities because of some
post-processing in a rendering software so I ended up building a DXF
text output by reformating the vertice/connectivity arrays (i.e.:
bypassing the Write method of the DXF object). Performance would
probably be better by calling the object's method but entities would
have to be converted to polygon anyway (?) so I abandonned this route.
Thank you very much for the clear explanations about the DXF object and
about why the Tesselator was messing my mesh.
Regards,
Thomas

Karl Schultz wrote:
> If your original input data was in the form of an implicit quad mesh, then
> you might be able to write it to DXF as a quad mesh instead of a polygon
> with a connectivity list.
>
> MESH_SMOOTH just moves the vertices around and returns the modified vertex
> list without changing the connectivity. (I was a bit wrong about this
> in my last posting) So, you should be able to submit the new vertex list to
> DXF, since the shape of the mesh hasn't changed and you don't need a
> connectivity list. Use the POLYGON type and specify the MESH_DIMS instead
> of the connectivity list. Again, this only works if your original data can
> be specified as an implicit quad mesh.
>
> If you are dealing with general polygons with connectivity data, then you
> might just want to try it with the POLYGON enitity with connectivity
> list.
>
> My reading of the problem description is that IDL ran the polygon
> through the tesselator when it did not need to. This resulted in storing a
> polygon in the DXF file that had a different vertex list and connectivity
> list, as compared to the input. But, depending on the input, the mesh
> stored in the DXF file could be equivalent to the input mesh. If the

> input mesh had things in it like overlapping faces, self-intersections, or
> holes, the mesh stored in the DXF file might not be topologically
> equivalent to the input mesh.
>
> In other words, if your mesh is simple or regular, you might get away with
> it.
>
> Otherwise, you are probably out of luck if you do not upgrade to 6.2. The
> fix for 6.2 was pretty involved and I don't think that there are any other
> sneaky workarounds.
>
> Karl
>

---