
Subject: Re: Surface mesh

Posted by [rtowler](#) on Thu, 10 Apr 2008 16:53:23 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Apr 9, 4:26 am, twhaw <wooihaw....@gmail.com> wrote:

> Good day.

>

> I have created a 3D model consisting of points sampled on the outline
> (surface) of a 3D object. The 3D object is initially translated so
> that the origin is at its center of mass. The points are then sampled
> in the spherical coordinates and converted back to the rectangular
> coordinated and stored as a (3, n) array. The outline of the model is
> sampled at a fixed interval longitudinally (every10 deg).

<snip>

> Now, I would like to create the surface mesh of the 3D model from the
> sample points. I have used MESH_OBJ (type 0) but the resulting mesh
> does not correspond to the surface of the 3D model. I have also
> triangulated the points first to get the polygons and passed them to
> IDLgrPolygon, but could not get the desired surface mesh.

This problem is similar to an earlier thread titled "how to compute the merged volume of two 3d objects" although you absolutely need to find the equilibrium surface since your object is concave. Unlike the poster in that thread, your task is much easier. Your vertices are ordered by the slice and you know that the verts in one slice will "connect" to the verts in the next. Easier still since the number of samples per slice is fixed. It gets even easier if your verts from each slice are sampled in the same order. Given this, you should be able to generate a polygon mesh comprised of quad strips that create bands connecting one slice to the next.

Since the planes pass thru the center of mass of your object (and I would assume the axis of rotation for your cutting planes is fixed) each slice should share 2 common vertices. Given your example object, the only trick I can see is that these common verts will be transition points where you will have to reverse the order of the polygon winding since the "top" and "bottom" row of verts will flip-flop. This wouldn't affect the wire-frame view, but if you don't change the winding the normals will be pointed inward which would affect lighting/shading.

You'll have a couple of for loops. The outer loops nSlices-1 (17) times. The inner will loop k times. In the inner you'll fill in your polygon array, starting at the common point on slice n, and connecting it's counterclockwise neighbor, then the neighbor's compliment on

slice $n+1$, then the commons's compliment on $n+1$. Repeat for each point on that slice, keeping in mind that when you get to the transition point again you'll need to change the order that you connect the verts. The last slice is then meshed to the first after your nested for loops in the same manner. For more complicated objects this gets harder but I think this should work for your example object.

You'll want to read the IDL docs on "About Polygon and Polyline Objects" and "Polygon Optimization" for more info on creating polygon arrays.

Have fun.

-Rick
