Subject: Re: closed surface
Posted by James Kuyper on Thu, 19 Sep 2002 18:21:59 GMT
View Forum Message <> Reply to Message

Chunlei Liu wrote:
...
> Triangulate, vertexlist[0, *], vertexlist[1, *], $
> tri,CONNECTIVITY=connect,/degrees

The online help says that the DEGREES keyword is only effective if the SPHERE keyword is also used.

...
> --> Triangulate, vertexlist[0, *], vertexlist[1, *], $
> tri,CONNECTIVITY=connect,/degrees,FVALUE=myF,SPHERE=myS
> ^^^^^^^^^^

I think that what you want for myF is a copy of vertexlist[2,*]. It's an input, as well as an output.

The online help doesn't mention the fact that when you use the FVALUE keyword, TRIANGLULATE rearranges not only the FVALUE, but also the corresponding 'x' and 'y' arguments. Therefore, 'x', 'y', and 'myF' must be passed by reference; which means that they have to be the names of variables, not subcript expressions.

Subject: Re: closed surface
Posted by Chunlei Liu on Thu, 19 Sep 2002 20:44:38 GMT
View Forum Message <> Reply to Message

James,
Thank you very much for the infomation. Following your suggestions, I got the triangulate work, however the sphere still does not look right.

Maybe my method does not work anyway. Does anybody has any suggetions about constructing a closed surface from some points sampled on the surface?

Chunlei

On Thu, 19 Sep 2002, James Kuyper wrote:

> Chunlei Liu wrote:
> ...
>> Triangulate, vertexlist[0, *], vertexlist[1, *], $
>> tri,CONNECTIVITY=connect,/degrees
>

> The online help says that the DEGREES keyword is only effective if the
> SPHERE keyword is also used.
>
> ...
>> --> Triangulate, vertexlist[0, *], vertexlist[1, *], $
>> tri,CONNECTIVITY=connect,/degrees,FVALUE=myF,SPHERE=myS
>> ^^^^^^^^^^
>
> I think that what you want for myF is a copy of vertexlist[2,*]. It's an
> input, as well as an output.
>
> The online help doesn't mention the fact that when you use the FVALUE
> keyword, TRIANGLULATE rearranges not only the FVALUE, but also the
> corresponding 'x' and 'y' arguments. Therefore, 'x', 'y', and 'myF' must
> be passed by reference; which means that they have to be the names of
> variables, not subcript expressions.
>

Subject: Re: closed surface
Posted by James Kuyper on Fri, 20 Sep 2002 23:04:23 GMT
View Forum Message <> Reply to Message

I've looked into this a little farther. If you choose the SPHERE option of TRIANGULATE, then the CONNECTIVITY option is ignored. The connectivity information is, instead, stored in the myS.iadj array. In any event, you don't really want the connectivity list, you want the triangles list.

There's another problem. The triangles list from TRIANGULATE is a long array of index triplets, each of which represents one triangle. However, the POLYGONS argument of idlgrPolygon::init() requires an argument which can contain multiple polygons, each with an arbitrary and different number of sides. Therefore, it needs some way of knowing the how many sides each one has. It does this by expecting the first element of each polygon to be a number telling how many sides that polygon has. Therefore, to create a polygon list suitable for idlgrPolygon::init() from the triangles list generated by TRIANGULATE, you'll have to insert a 3 just before each triangle.

If you're just trying to make a sphere, you can ignore the CV_COORD stuff, and just use myS.XYZ for your vertex list.

One additional point: your points are not evenly distributed over the surface of the sphere. They have a density proportional to 1.0/cos(latitude). To get a more evenly distributed set of points, use:

latitude = asin(2.0*RANDOMU(200)-1.0)*!RADEG

So, here's my revised version of your code:

```
longitude = RANDOMU(seed, 200) * 360. - 180.
latitude = ASIN(2.0*RANDOMU(seed, 200)-1.0)*!RADEG
radius = REPLICATE(300.0,200)

TRIANGULATE, longitude, latitude, triangles, $
 FVALUE=radius,/DEGREES, SPHERE=myS

; I'll bet that IDL experts have a better way of doing this, but
; this works
ntri = SIZE(triangles,/DIMENSIONS)
ntri = ntri[1]
connect = LONARR(4,ntri)
connect[0,*] = 3
connect[1:3,*] = triangles

oSurf = OBJ_NEW('IDLgrPolygon', TRANSPOSE(myS.XYZ), POLYGON=connect, $
 STYLE=2, SHADING=1, COLOR=[0,20,255])
oGroup = OBJ_NEW('IDLgrModel')
oGroup->ADD, oSurf
XOBJVIEW,oGroup
```

It still looks lumpy, but it's a lot better than before.
