Subject: Re: 3D triangulation of x,y,z vertices
Posted by Dick Jackson on Mon, 02 Oct 2006 08:35:51 GMT
View Forum Message <> Reply to Message

Hi Thomas,

I think the terminology used in the docs is a bit confusing. QHull's output "Connectivity list" *sounds* like something you'd want for Tetra_Surface's Connin "Tetrahedral connectivity array"... but it's not. What you want to use is QHull's output "Tr" which is a 4-by-nTetra array of indices for each tetrahedron.

=====

```
PRO QHullTetra

oldverts = RandomU(seed, 3, 20)
Qhull, oldverts, tetrahedra, /delaunay
newconn=tetra_surface(oldverts, tetrahedra)
oPts = Obj_New('IDLgrPolygon', oldverts, Style=0, Thick=3)
oSurf = Obj_New('IDLgrPolygon', oldverts, Polygons=newconn, Color=[255,0,0])
XObjView, [oPts, oSurf]

END
```

=====

To see any points hiding inside, choose menu item View:Drag Quality:Low, then press and drag!

In IDL, a "connectivity LIST" is a description of a general polygon mesh: [nPts0, <set of "nPts0" indices>, nPts1, <set of "nPts1" indices>, ...]

But, as described in Tetra_Clip's doc (but not in all of the tetra-routines' docs, alas):

=====
A tetrahedral connectivity array consists of groups of four vertex index values. Each set of four index values specifies four vertices which define a single tetrahedron.
=====

Hope this helps!

Cheers,
-Dick

--

Dick Jackson Software Consulting http://www.d-jackson.com
Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com

"Thomas Launey" <t_launey@brain.riken.jp> wrote in message
news:1159708839.580643.297020@h48g2000cwc.googlegroups.com.. .
> Hello,
>
> I have a 3D object defined by its surface triangles. The connectivity
> is a bit messed up with some faces normals pointing inward (toward the
> inside of the object). Since I have all the points describing the
> surface, I thougth that it would be easy to re-triangulate the x,y,x
> vertices but apparently, I am missing the obvious...
> What I did is:
> Qhull, oldverts, tetrahedra, /delaunay, connectivity=connectivity
> newconn = TETRA_SURFACE (oldvert, connectivity)
>
> It fails however because the connectivity list returned by the Delaunay
> triangulation is not recognized as a proper connectivity list for
> tetrahedra. The IDL doc actually describe the 'connectivity' returned
> by Qhull as an adjacency list. I tried to reformat it but without
> success.
> Any help or pointer would be very much appreciated .
> Thanks,
> Thomas
>

Subject: Re: 3D triangulation of x,y,z vertices
Posted by Thomas Launey on Mon, 02 Oct 2006 13:44:56 GMT
View Forum Message <> Reply to Message

Thanks dick,
I still have a problem though because "tr" returns the tetrahedra of
the convex Hull but what I failed to make clear in my post is that I do
not want the convex hull, only the Delaunay triangulated vertices.
According to the Qhull doc, when /Delaunay is set, Qhull 'Performs a
Delaunay triangulation and returns the vertex indices of the resulting
polyhedra; otherwise, the convex hull of the data are returned. '
Its seems that even with the /Delaunay switch set, the tr polyhedra
still represent the convex hull.
Any clarification or and alternative to re-triangulate 3D vertices
would be most welcome :-)
Thanks,
Thomas

Dick Jackson wrote:
> Hi Thomas,

>
> I think the terminology used in the docs is a bit confusing. QHull's output
> "Connectivity list" *sounds* like something you'd want for Tetra_Surface's
> Connin "Tetrahedral connectivity array"... but it's not. What you want to
> use is QHull's output "Tr" which is a 4-by-nTetra array of indices for each
> tetrahedron.
>
> =====
>
> PRO QHullTetra
>
> oldverts = RandomU(seed, 3, 20)
> Qhull, oldverts, tetrahedra, /delaunay
> newconn=tetra_surface(oldverts, tetrahedra)
> oPts = Obj_New('IDLgrPolygon', oldverts, Style=0, Thick=3)
> oSurf = Obj_New('IDLgrPolygon', oldverts, Polygons=newconn, Color=[255,0,0])
> XObjView, [oPts, oSurf]
>
> END
>
> =====
>
> To see any points hiding inside, choose menu item View:Drag Quality:Low,
> then press and drag!
>
> In IDL, a "connectivity LIST" is a description of a general polygon mesh:
> [nPts0, <set of "nPts0" indices>, nPts1, <set of "nPts1" indices>, ...]
>
> But, as described in Tetra_Clip's doc (but not in all of the tetra-routines'
> docs, alas):
>
> =====
> A tetrahedral connectivity array consists of groups of four vertex index
> values. Each set of four index values specifies four vertices which define a
> single tetrahedron.
> =====
>
> Hope this helps!
>
> Cheers,
> -Dick
>
> --
> Dick Jackson Software Consulting http://www.d-jackson.com
> Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com
>

Subject: Re: 3D triangulation of x,y,z vertices
Posted by Dick Jackson on Tue, 03 Oct 2006 06:09:16 GMT
View Forum Message <> Reply to Message

Hi Thomas,

Another terminology confusion is that QHull refers to Delaunay triangulation while, in the case of 3D vertices, it is actually (deep breath, and...) tetrahedralization. The result is not the triangles of a surface, but the tetrahedra of the volume within the convex hull of the points.

Your original post:
> I have a 3D object defined by its surface triangles. The connectivity
> is a bit messed up with some faces normals pointing inward (toward the
> inside of the object). Since I have all the points describing the
> surface, I thougth that it would be easy to re-triangulate the x,y,x
> vertices

I see you want neither these tetrahedra nor the convex hull, as you have the desired *adjacency* info for the triangles of the desired surface and want the connectivity list for a good polygon mesh with consistent triangle facing.

If you know that your triangles describe a good mesh (except for inconsistent facing), you should be able to do this by ensuring that no segment (between two vertices) is used in the same direction more than once. If a maximum of two triangles share a segment (as they should in a valid mesh), one triangle should traverse the vertices as A->B, and the other as B->A. I put some time into a solution for this, but it's not quite finished and I have to put it aside right now. Write me directly if you are interested in what I've done.

Cheers,
-Dick

"Thomas Launey" <t_launey@brain.riken.jp> wrote in message news:1159796696.413229.92510@m7g2000cwm.googlegroups.com...
> Thanks dick,
> I still have a problem though because "tr" returns the tetrahedra of
> the convex Hull but what I failed to make clear in my post is that I do
> not want the convex hull, only the Delaunay triangulated vertices.
> According to the Qhull doc, when /Delaunay is set, Qhull 'Performs a
> Delaunay triangulation and returns the vertex indices of the resulting
> polyhedra; otherwise, the convex hull of the data are returned. '
> Its seems that even with the /Delaunay switch set, the tr polyhedra
> still represent the convex hull.
> Any clarification or and alternative to re-triangulate 3D vertices
> would be most welcome :-)

> Thanks,
> Thomas
>
> Dick Jackson wrote:
>> Hi Thomas,
>>
>> I think the terminology used in the docs is a bit confusing. QHull's
>> output
>> "Connectivity list" *sounds* like something you'd want for
>> Tetra_Surface's
>> Connin "Tetrahedral connectivity array"... but it's not. What you want to
>> use is QHull's output "Tr" which is a 4-by-nTetra array of indices for
>> each
>> tetrahedron.
>>
>> =====
>>
>> PRO QHullTetra
>>
>> oldverts = RandomU(seed, 3, 20)
>> Qhull, oldverts, tetrahedra, /delaunay
>> newconn=tetra_surface(oldverts, tetrahedra)
>> oPts = Obj_New('IDLgrPolygon', oldverts, Style=0, Thick=3)
>> oSurf = Obj_New('IDLgrPolygon', oldverts, Polygons=newconn,
>> Color=[255,0,0])
>> XObjView, [oPts, oSurf]
>>
>> END
>>
>> =====
>>
>> To see any points hiding inside, choose menu item View:Drag Quality:Low,
>> then press and drag!
>>
>> In IDL, a "connectivity LIST" is a description of a general polygon mesh:
>> [nPts0, <set of "nPts0" indices>, nPts1, <set of "nPts1" indices>, ...]
>>
>> But, as described in Tetra_Clip's doc (but not in all of the
>> tetra-routines'
>> docs, alas):
>>
>> =====
>> A tetrahedral connectivity array consists of groups of four vertex index
>> values. Each set of four index values specifies four vertices which
>> define a
>> single tetrahedron.
>> =====
>>

>> Hope this helps!
>>
>> Cheers,
>> -Dick
>>
>> --
>> Dick Jackson Software Consulting http://www.d-jackson.com
>> Victoria, BC, Canada +1-250-220-6117 dick@d-jackson.com
>>
>