Subject: Re: MESH_DECIMATE
Posted by Karl Schultz on Thu, 28 Mar 2002 01:02:29 GMT
View Forum Message <> Reply to Message

"Reimar Bauer" <r.bauer@fz-juelich.de> wrote in message
news:3CA21E59.1A63F0F6@fz-juelich.de...
> Hi,
>
> did I miss something I like to use a function like mesh_decimate
> but for vectors and not only for 3D Arrays.
>
> Did someone have already such a routines.
>
> Reimar
>

I'm not really sure what you are asking for, but the rest of this posting
assumes that you want to decimate polylines, as opposed to polygons.

MESH_DECIMATE is good for decimating things like regularly sampled height
fields. The decimator removes vertices that are not as important as others
in describing the height field. It will remove vertices in flat areas, for
example. See the DECIMATE example that ships with IDL.

The trouble is, there's no real easy way to do the same thing with
polylines. Suppose that you had a highly detailed polyline (lots of
vertices) that describes a coastline. You may want to simplify the line by
reducing the number of vertices at the expense of some unwanted detail.
There are various algorithms and approaches for the problem, but I don't
think there's anything to directly do this in IDL.

So, here's one of the tackiest things I've ever done with IDL. You can
convert your polyline into a polygon extrusion, decimate that, and then take
a border of your remaining extrusion as your decimated polyline. It is
overkill, but is pretty cool nonetheless. Enjoy.

Karl

```
PRO coastline

   filename = FILEPATH('states2.sav',
SUBDIRECTORY=['examples','demo','demodata'])
   RESTORE, filename

   ; pick a state and get its outline data
   n = 57
   PRINT, "State is ",  states[n].state
   outline = *states[n].poutline
   ; free stuff we do not need
   PTR_FREE, states.poutline

   ; build vertex array of outline, plus another copy of the outline
stacked on top in Z
   nPoints = N_ELEMENTS(outline[0,*])
   pts = FLTARR(3,nPoints*2)
   pts[0,0:nPoints-1] = outline[0,0:nPoints-1]
   pts[1,0:nPoints-1] = outline[1,0:nPoints-1]
   pts[2,0:nPoints-1] = 0
   pts[0,nPoints:2*nPoints-1] = outline[0,0:nPoints-1]
   pts[1,nPoints:2*nPoints-1] = outline[1,0:nPoints-1]
   pts[2,nPoints:2*nPoints-1] = 10

   ; build connectivity array to make quads between the two outlines.
   ; this will look like an extrusion of the outline
   conn = LONARR(5 * nPoints)
   conn[LINDGEN(nPoints)*5] = 4
   conn[LINDGEN(nPoints)*5+1] = LINDGEN(nPoints)
   conn[LINDGEN(nPoints)*5+2] = LINDGEN(nPoints) + 1
   conn[LINDGEN(nPoints)*5+3] = LINDGEN(nPoints) + nPoints + 1
   conn[LINDGEN(nPoints)*5+4] = LINDGEN(nPoints) + nPoints
   conn[5 * nPoints - 3] = nPoints
   conn[5 * nPoints - 2] = 0

   ; look at the original extrusion
   oPolygon1 = OBJ_NEW('IDLgrPolygon', pts, POLYGON=conn, COLOR=[255,0,0])
   xobjview, oPolygon1

   ; decimate and look at the decimated extrusion
   n = MESH_DECIMATE(pts, conn, new_conn, PERCENT_VERTICES=50)
   oPolygon2 = OBJ_NEW('IDLgrPolygon', pts, POLYGON=new_conn,
COLOR=[0,255,0])
   xobjview, oPolygon2

   ; Now pull out the vertices that remain after the decimation

   ; First, filter out the 3's from the conn list
   ; Replace the 3's with a "big" value that we'll filter out later
   i = LINDGEN(N_ELEMENTS(new_conn)/4)*4
   line_conn = new_conn
   line_conn[i] = nPoints

   ; Now keep only the vert indicies from the decimated list that are
   ; smaller than nPoints.  This gets rid of all the verts from the top
   ; of the extruded outline.
   i = WHERE(line_conn LT nPoints)
   line_conn = line_conn[i]
```

```
    ; Now sort and uniq the list, so that we only get one of each vertex,
    ; and in the right order.
    ; Otherwise, we'd have duplicate verts introduced by the triangles.
    line_conn = line_conn[UNIQ(line_conn, SORT(line_conn))]

    PRINT, nPoints, ' points in the original (red) outline.'
    PRINT, N_ELEMENTS(line_conn), ' points in the decimated (green)
outline.'

    oPolyline1 = OBJ_NEW('IDLgrPolyline', pts[*, 0:nPoints-1],
COLOR=[255,0,0])
    oPolyline2 = OBJ_NEW('IDLgrPolyline', pts[*,line_conn], COLOR=[0,255,0])
    xobjview, [oPolyline1, oPolyline2], /BLOCK
    OBJ_DESTROY, [oPolygon1, oPolygon2, oPolyline1, oPolyline2]
END
```