EXERCISES FOR INF3320

TRIANGLE MESHES

3/10/2010

1. A tetrahedron is a pyramid where all four faces are triangles.
   (a) What is the Euler-characteristic and the genus?
   (b) Write down an indexed triangle set describing this shape.
2. A cube is consists of six quadrilaterals.
   (a) What is the Euler-characteristic and the genus?
   (b) Write down an indexed face set describing this shape.

   Then, assume we have tesselated the cube into triangles If we move one of the corners away from the origin, how does our choice of tessellation affect the surface?

   Further, what can we do to improve the shape? Give an outline to an algorithm.
3. Several file formats have been presented in the lecture. Summarize some of the strength and weaknesses of these. Search the web and see if you find more file formats, and find the strength and weaknesses on these as well.
4. An usual measure on the efficiency on a rendering sequence is the number of vertices processed by the number of triangles produced versus the number of vertices processed by the GPU. The GPU has a post transform vertex-cache that holds around 12 vertices.
   (a) Assume we can specify the entire mesh as a single long triangle strip, in general, what is the efficiency of using this rendering sequence?
   (b) Is there a better way?
5. Mesh decimation (simplification) usually gives some degradation. In which situation does mesh decimation *not* degrade the mesh?
6. Using the ear clipping algorithm of Real-time Rendering page 442 in 2nd edition/536 in 3rd edition, find a concave polygon with a simple border where this algorithm fails.
7. Implement the function `earClipTriangluate` which tessellates polygons with a simple border (both convex and non-convex) into triangles.

   The algorithm presented in the book fails for some polygons, an algorithm that works is the following:

   Given the polygon $P = \{\mathbf{p}_0, \ldots, \mathbf{p}_{N-1}\}$:

(a) If the number of points of P is 3, add this triangle and terminate.

(b) Find a protruding point $\mathbf{p}_i$ in P by traversing the edge of the polygon in a counter clockwise order. Create the triangle $[\mathbf{p}_{i-1}, \mathbf{p}_i, \mathbf{p}_{i+1}]$ and remove $\mathbf{p}_i$ from P.

(c) Go to step (a).

A protruding point is a point $\mathbf{p}_i$ that satisfies:

- The point $\mathbf{p}_i$ is *outside* the half-plane defined by the line $[\mathbf{p}_{i-1}, \mathbf{p}_{i+1}]$.
- All other points are outside all the three halfplanes $[\mathbf{p}_{i-1}, \mathbf{p}_i]$, $[\mathbf{p}_i, \mathbf{p}_{i+1}]$, og $[\mathbf{p}_{i+1}, \mathbf{p}_{i-1}]$ (that is, all points are outside the triangle $[\mathbf{p}_{i-1}, \mathbf{p}_i, \mathbf{p}_{i+1}]$).

The notion of a point $\mathbf{p}$ being inside or outside a halfplane by the line $[\mathbf{a}, \mathbf{b}]$ is as follows.

We say $\mathbf{p} = (p_1, p_2)$ is **inside** a half-plane defined by $[\mathbf{a}, \mathbf{b}]$, where $\mathbf{a} = (a_1, a_2)$ and $\mathbf{b} = (b_1, b_2)$ if and only if

$$0 < cross2d(b - a, p - a) = \begin{vmatrix} b_1 - a_1 & p_1 - a_1 \\ b_2 - a_2 & p_2 - a_2 \end{vmatrix}$$

We say $\mathbf{p} = (p_1, p_2)$ is **outside** a half-plane defined by $[\mathbf{a}, \mathbf{b}]$, where $\mathbf{a} = (a_1, a_2)$ and $\mathbf{b} = (b_1, b_2)$ if and only if

$$0 > cross2d(b - a, p - a) = \begin{vmatrix} b_1 - a_1 & p_1 - a_1 \\ b_2 - a_2 & p_2 - a_2 \end{vmatrix}$$

We say $\mathbf{p}$ is **colinear** with $\mathbf{a}$ and $\mathbf{b}$ if it is neither inside or outside a half-plane defined by $[\mathbf{a}, \mathbf{b}]$.

Notice that

$$cross2d(b - a, p - a) = cross2d(b - p, p - a) + cross2d(p - a, p - a) =$$

$$-cross2d(p - a, b - p) + 0 = cross2d(a - p, b - p) = orient2d(a, b, p)$$

So, to check if $\mathbf{p}$ is inside or outside the halfplane defined by the line $[\mathbf{a}, \mathbf{b}]$, we inspect the sign of the signed area of the triangle $[\mathbf{a}, \mathbf{b}, \mathbf{p}]$.

Intuitively we could say, a point $\mathbf{p}$ is **inside** a half-plane defined by $[\mathbf{a}, \mathbf{b}]$ if when "walkingfrom $\mathbf{a}$ to $\mathbf{b}$ the point $\mathbf{p}$ is on the left.

You can start using `ex6-7_triangulatepoly.cpp.template`.