CS 491/591 ECE 495/595 Spring 2012

Ed Angel

Assignment1: Tessellation and Twisting

Almost all graphics systems are best at displaying triangles because, as long as the three vertices are not collinear, such polygons are simple, convex, and flat. Other more complex polygons and curved surfaces are displayed by breaking them up or *tessellating* them into triangles. Although the OpenGL 4.1 provides a polygon tessellator, it is more common for tessellation to be left to the application program. This exercise should teach you something about tessellation and let you practice your OpenGL.

Suppose that we start with an arbitrary triangle in the plane $z = 0$ that is centered approximately at the origin. We can subdivide this triangle into smaller triangles by a variety of methods. One is to bisect each side and connect the bisectors, thus forming four triangles from the original, a process that can be continued recursively. See the text for a discussion of this and other methods. The text also contains code for triangle subdivision.

Suppose that after some number of subdivisions, we rotate each triangle's vertices about the origin by θ degrees and then redisplay all the triangles. The new coordinates of a vertex (x, y) are given by

$$x' = x \cos\theta - y \sin\theta$$

$$y' = x \sin\theta + y \cos\theta$$

It should be easy to convince yourself that regardless of how many subdivisions we carry out, we get the same display when we either fill all triangles with the same color or display only the outer edges. In other words, we see the same display as if we had not done any subdivision steps but had simply rotated the original three vertices and displayed a single triangle. Now suppose that after some number of subdivisions, we rotate each vertex by an amount that depends on how far from the origin that vertex is located. Thus we replace θ in our rotation formula by $d * \theta$, where

$$d = \sqrt{x^2 + y^2}$$

This type of rotation is sometimes called *twisting*. Now the amount of tessellation—the number of subdivision steps—matters.

You are to write a program using OpenGL that will allow you to demonstrate twisting and tessellation. The user should be able to enter both the level of tessellation—the number of subdivisions—and the angle θ. You can have the original vertices determined inside your program or entered by the user. You can display your results as filled triangles or by their edges. Note that if you allow for large values of x and y when you specify the initial triangle, the values of θ should be very small. You might have your program scale the d so as to achieve a degree of twist that is about the same for a given θ.

A good program will adjust the window and/or viewport to best show the results and have an appropriate reshape callback. A really good program will allow for more complex initial polygons than triangles.

This assignment should give you an opportunity to test the difference between carrying out operations such as rotation in the application versus doing the same operation in the application. With small modifications, you can use this program to test how many triangles per second you can render on your system.