Assignment 6: Rendering

15-462 Graphics I
Spring 2003
Frank Pfenning

Out March 20
Due April 3 before lecture
50 points

- The work must be all your own.
- The assignment is due **before lecture** on Thursday, April 3.
- Be explicit, define your symbols, and explain your steps.
  This will make it a lot easier for us to assign partial credit.

1 Blending (20 pts)

When drawing translucent objects in OpenGL a simple use of the z-buffer is no longer sufficient: unless we know about special properties of the scene, we need to order the objects by their distance to the viewer and draw them in a specific order. For this question, assume the objects do not interpenetrate and can be put into a total order based on the distance to the viewer. We do not consider refraction or similar effects.

1. Explain in detail how to draw the objects in OpenGL using blending, once they have been ordered. Take care to describe the commands to achieve the effect of translucency and the treatment of the z-buffer (if you are using it). You do not need to give an implementation, but you should complement your explanation with the appropriate calls to the OpenGL library.

2. Illustrate by means of a counterexample why the objects need to be ordered.

2 Intersections (30 pts)

Now assume we are just drawing flat, simple, convex polygons. Sometimes they overlap or interpenetrate so that they cannot be put into a total order and we have to calculate intersections to generate smaller polygons that can then be ordered.

1. Describe in detail an algorithm to determine if two flat, simple, convex polygons given by points $\mathbf{p}_0, \ldots, \mathbf{p}_{n-1}$ and $\mathbf{q}_0, \ldots, \mathbf{q}_{k-1}$ intersect. Be careful to distinguish and treat correctly special cases that may arise.

2. What is the worst-case complexity of your algorithm in terms of the number of vertices n and k?

3. Describe how to extend your algorithm above to subdivide the polygons if necessary. These subdivided polygons might intersect but they should no longer interpenetrate which means that they can be ordered.

4. How many polygons can result in your algorithm from the two given ones? Will they also always be flat, simple, and convex?