Questions

1. In OpenGL, back face culling is done at the rasterization stage. What needs to be computed to cull a polygon at this stage? How would you compute that?

2. The depth buffer partitions the near to far range of depth into intervals using fixed precision (typically 24 bits, hence 2^{24} intervals). The mapping from continuous to discrete depth happens late in the pipeline, in particular, it maps from the normalized device depth coordinate to a (24 coding) of the range [0,1].

   The depth "bins" do *not* correspond to uniform sized depth bins in the original scene space, however. Why not? Are the close bins bigger or smaller than the far bins, when measured in the original scene space ?

3. Suppose we have a scene with n polygons. The Painter's Algorithm requires that we sort the polygons by the depth of the furthest vertex. This takes time $O(n \log n)$. I did not provide full details of the Painter's algorithm. However, given the details that I did provide, what would you say is the $O(\ )$ time complexity ?

1. We need to know the sign of the z value of the surface normal, specifically, the surface normal of the polygon *in the normalized device coordiantes*. To compute the surface normal, we could take the cross product of two vectors – say $(\mathbf{p_1} - \mathbf{p}_0) \times (\mathbf{p_2} - \mathbf{p}_1)$, where the $\mathbf{p}$'s are successive vertices of the polygon. However, we only need to know the z component of this cross product, so that's all we need to compute.

2. This question is about the perspective mapping. From bottom of lecture 5 p.2, we see that z is mapped to $f_0 + f_1 - \frac{f_0 f_1}{z}$. There is a normalization that occurs after this, which translates and scales. But let's not deal with that yet.

   The main effect is the z to $-\frac{1}{z}$ map. A small interval $[z, z + dz]$ gets mapped to interval

   $$[f_0 + f_1 - \frac{f_0 f_1}{z} \ , f_0 + f_1 - \frac{f_0 f_1}{z + dz} \ ].$$

   From Calculus 1,

   $$\frac{1}{z} - \frac{1}{z + dz} \approx \frac{dz}{z^2}$$

   the size of this interval is approximately $\frac{f_0 f_1}{z^2} dz$. Thus, the size of the mapped interval would decreases as z increases.

   The question asks something slightly different. The bins are uniform in the mapped (projective) space. So we need to apply the above argument backwards. Depth buffer bins of a uniform depth range in projective space map backwards to non-uniform bins in the original scene space. I will spare you the math, but you should be able to convince yourself that bins that represent points far from the camera have a larger depth range in the original scene than bins that represent point that are close to the camera. So the far depth points get compressed in depth, just as they get compressed in x and y.

   We are not out of the woods yet. You translate the volume so the near plane gets mapped to $z = 0$ and then you scale so that the far plane is mapped to 1. (Note this is not the normalized device coordinates here. Rather, I am jumping all the way to window coordinates where the pixels are, and depth goes from 0 to 1). This scaling has some effect, but it does *not* affect whether one interval is bigger than another. Intuitively, you can't make one object bigger than another by changing the size units from cm to mm.

3. For each polygon P and Q we need to ensure that P can indeed be drawn before Q. There are $O(n^2)$ pairs of polygons.

   This seems like a large cost. However, in practice it is not so bad. Consider a polygon Q and let's say its furthest vertex is at depth z_Q. Do we need to compare it with all polygons that come before it in the list? No, we don't. Suppose that Q's closest (to the camera) vertex has depth $z_{Q,min}$. Then we only need to examine polygons P whose further vertex has depth greater than $z_{Q,min}$.