Instructor: Zoran Popović

Computer Graphics

CSE 457, Autumn 2002

Homework #2

Hidden Surfaces, Shading, Ray Tracing, Texture Mapping, Parametric Curves

Assigned: Friday, November 8th
Due: Wednesday, November 27th, **at the beginning of class**

Directions: Please provide short written answers to the questions in the space provided. If you require extra space, you may staple additional pages to the back of your assignment. Feel free to discuss the problems with classmates, but please ***answer the questions on your own*.**

Name:_____

Problem 1. TRUE/FALSE Justify each answer. (10 points)

a. By increasing the number of polygons, you can make the difference between Gouraud interpolation and Phong interpolation to be arbitrarily small—you can make the polygons small enough that there is no perceivable difference.

b. Every third order Bezier curve can be broken up into two other third-order Bezier.

c. Every C^2 continuous spline is also C^1 continuous.

d. Gouraud interpolation cannot produce specular highlights.

e. In the Phong model, specular reflection does not depend on the view angle.

Problem 2. Phong Shading (15 points)

The Phong shading model can be summarized by the following equation:

$$I_{phong} = k_e + k_a I_a + \sum_i \left[I_{l_i} \left[k_d \left(\mathbf{N} \cdot \mathbf{L}_i \right)_+ + k_s \left(\mathbf{V} \cdot \mathbf{R}_i \right)_+^{n_s} \right] \min\left\{ 1, \frac{1}{a_0 + a_1 d_i + a_2 d_i^2} \right\} \right]$$

where the summation i is taken over all light sources. The variables used in the Phong shading equation are summarized below:

I a_0 a_1 a_2 d_i k_e k_a k_d k_s n_s I_a I_{li} $\mathbf{L}_i$ $\mathbf{R}_i$ $\mathbf{N}$ $\mathbf{V}$

a. Which of the quantities above are affected if the viewing direction changes?

Now imagine a scene consisting of a sphere illuminated by white global ambient light and a single white directional light. Assume the directional light is pointing in the same direction as the viewer. Describe the effect of the following conditions on the shading of the object. At each incremental step, assume that all the preceding steps have been applied first.

b. Now increase the specular reflection coefficient k_s of the material to be greater than zero. What happens?

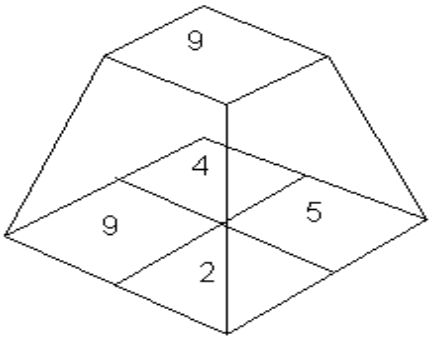
c. Now increase the specular exponent n_s. What happens?

d. Now rotate the sphere about its own origin and then translate it straight towards the viewer. What happens to the shading of the sphere?

e. The directional light is off. How does the shading vary over the surface of the object?

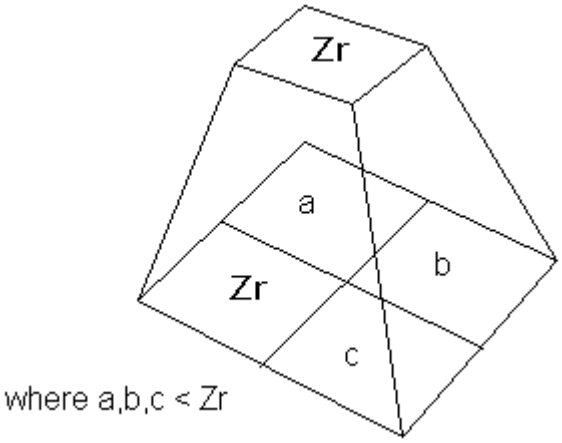
Problem 3. Z-buffer (20 points)

The Z-buffer algorithm can be improved by using an image space "Z-pyramid." The basic idea of the Z-pyramid is to use the original Z-buffer as the finest level in the pyramid, and then combine four Z-values at each level into one Z-value at the next coarser level by choosing the farthest (largest) Z from the observer. Every entry in the pyramid therefore represents the farthest (largest) Z for a square area of the Z-buffer. A Z-pyramid for a single 2x2 image is shown below:



a. At the coarsest level of the pyramid there is just a single Z value. What does that Z value represent?

Suppose we wish to test the visibility of a polygon **P**. Let **Zp** be the nearest (smallest) Z value of polygon **P**. **R** is a region on the screen that encloses polygon **P**, and is the smallest region of the Z-pyramid that does so. And let **Zr** be the Z value that is associated with region **R** in the Z-pyramid.



b. What can we conclude if **Zr** < **Zp**?

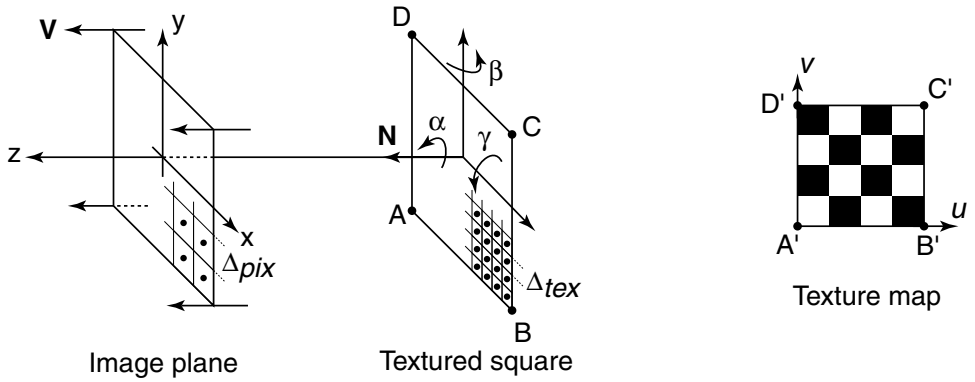
c. What can we conclude if **Zp** < **Zr**?

If the visibility test is inconclusive, then the algorithm applies the same test recursively: it goes to the next finer level of the pyramid, where the region **R** is divided into four quadrants, and attempts to prove that polygon **P** is hidden in each of the quadrants **R** of that **P** intersects. Since it is expensive to compute the closest Z value of **P** within each quadrant, the algorithm just uses the same **Zp** (the nearest Z of the *entire* polygon) in making the comparison in every quadrant. If at the bottom of the pyramid the test is still inconclusive, the algorithm resorts to ordinary Z-buffered scan conversion to resolve visibility.

d. Suppose that, instead of using the above algorithm, we decided to go to the expense of computing the closest Z value of **P** within each quadrant. Would it then be possible to always make a definitive conclusion about the visibility **P** of within each pixel, without resorting to scan conversion? Why or why not?

Problem 4. Texture Filtering (20 points)

In class, we discussed how brute force sampling, mip maps, and summed area tables can be employed to anti-alias textures. The latter two techniques average over a region of the texture image very quickly with varying degrees of accuracy, which we consider further in this problem. Consider the scene below: an ***orthographic viewer*** looking down the –z-axis views a textured square. The image size and square size are the same and they are initially aligned to one another as shown. The pixel spacing on the image plane and the texel spacing on the square are Δ_{pix} and Δ_{tex}, respectively.



a. Assuming $\Delta_{pix} > \Delta_{tex}$, how must these sample spacings be related in order for mip mapping to yield the correct values without interpolating among mip map levels?

Problem 4 (cont'd)

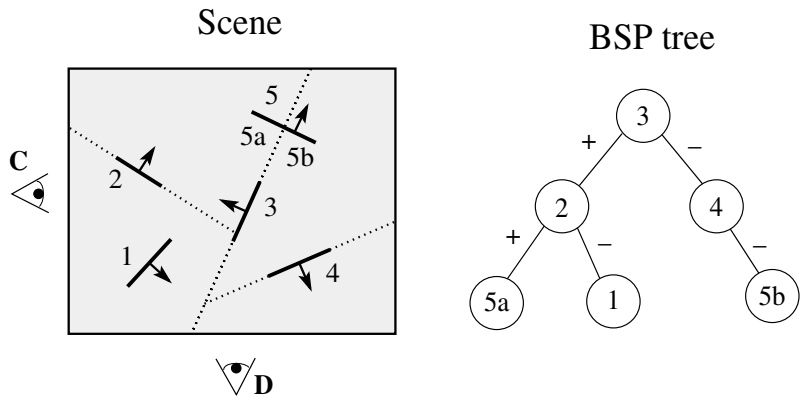
b. Consider the coordinate system of the square shown in terms of the normal **N** and the two axes aligned with the x and y axes in the figure. Assume that we have the freedom to rotate the square about the x-axis, the y-axis, or the N vector, as indicated by rotation angles α, β, and γ. What restriction do we have on rotation about any one of these axes in order for mip mapping to return the correct average texture values? [For example, you could decide that α, β, and γ must all be zero degrees, or you could decide that some of them can vary freely, or you can decide that some can take on a set of specific values. Do not focus on rotations that cause the square to be back-facing.]

c. Now assume we start again with the unrotated geometry and that we're using summed area tables. If linear interpolation within the summed area table causes no significant degradation, what restriction, if any, should we place on the relative pixel and texel spacings to get correct texture averaging?

d. As in (b), what restriction must we place on rotation about any one of the given axes in order for summed area tables to return the correct average texture values?

Problem 5. BSP tree (15 points)

Recall that a BSP tree breaks the world up into tree of positive and negative half-spaces that can be traversed to render a scene from an arbitrary viewpoint. Below is the **exact same figure used in class** to illustrate the principle of BSP trees:



Recall that each arrow in the scene tells us which way the normal to a polygon is facing and that the normal points to the positive half-space of a polygon.

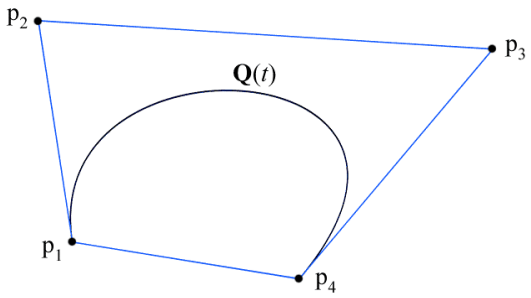
a. Given viewpoint **C**, list the polygons in the order in which they would be drawn after traversing the BSP tree to give a back to front ordering.

b. Given viewpoint **D**, list the polygons in the order in which they would be drawn after traversing the BSP tree to give a back to front ordering.

c. If we move a polygon in the scene, will we always have to recompute the BSP tree? Justify your answer.

Problem 6. Parametric Curves (15 points)

A nice property of Bezier curves is that the curve itself will always remain within the *convex hull* of its control points. The convex hull of a set of points is defined as the smallest convex polygon containing all those points. Intuitively, you might imagine the convex hull of a set of points in two dimensional space to be the polygon defined by wrapping a rubber band around those points. In three dimensional space, imaging using a rubber sheet instead.



An intuitively true property about convex hulls is as follows. Suppose we are given *n* points; call these $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$. Now suppose we are given *n* real numbers, $w_1, w_2, \mathrm{K}\ , w_n$. If $0 \le w_i \le 1$ for all $1 \le i \le n$ and $w_1 + w_2 + \ldots + w_n = 1$, then $\mathbf{q} = w_1\mathbf{p}_1 + w_2\mathbf{p}_2 + \ldots + w_n\mathbf{p}_n$ lies within the convex hull of the points $\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_n$. In other words, taking a weighted average of a set of points necessarily gives a point within the convex hull of those points.

a. A point on a cubic Bezier curve can be defined by the function

$$\mathbf{Q}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \\ \mathbf{p}_4 \end{bmatrix}$$

where $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$ are the control points of the curve and $0 \le t \le 1$. Write out the Bezier basis functions $f_1(t),\ f_2(t),\ f_3(t),\ f_4(t)$ such that

$$\mathbf{Q}(t) = f_1(t)\mathbf{p}_1 + f_2(t)\mathbf{p}_2 + f_3(t)\mathbf{p}_3 + f_4(t)\mathbf{p}_4 .$$

Problem 6 (cont'd)

b. Show that $f_1(t) \geq 0,\ f_2(t) \geq 0,\ f_3(t) \geq 0,\ f_4(t) \geq 0$ for all $0 \leq t \leq 1$.

c. Show that $f_1(t) + f_2(t) + f_3(t) + f_4(t) = 1$ for all $0 \leq t \leq 1$.

d. Using the property about convex hulls stated previously, argue that any Bezier curve must lie within the convex hull of its control points. (Make sure you use the convex hull property exactly as it is stated)

e. Give an example of a situation in which the convex hull property of Bezier curves might be useful.