Practice Final
CMSC 427
Distributed Tuesday, December 11, 2007
Review Session, Monday, December 17, 5:00pm, 4424 AV Williams
Final: 10:30 AM Wednesday, December 19, 2007

General Guidelines: The final will focus on topics that have been discussed in class. I will not ask questions about material from the text book that has not also been discussed in class or in a problem set. However, there will be no questions about OpenGL or programming problems. You will be expected to know some basic equations, but nothing very obscure. I'll try to indicate below the key equations you should know.

The final will be closed book, with no notes or calculators. This means that problems should not require the use of a calculator; if you feel you need one when you are doing a problem this probably means there is an easier way to do the problem.

The final will be comprehensive. It will focus more (about 3/4 emphasis) on the material covered since the midterm, but you are still responsible for knowing material from the first half of the semester.

Topics and things to know about them:

1. Geometry (needed for problems on the next two topics)
    a. How to take an inner product, and what this does.
    b. How to take a cross product, and what this does.
    c. What a unit vector is, and how to find the magnitude of a vector.
    d. How to write the equations for a 3D line, or for a plane in 3D, or for a sphere.
2. Transformations
    a. How to write rotation about the *x, y,* or *z* axis as a matrix.
    b. How to write translation as a matrix.
    c. How to combine these. For example, rotate about a point not at the origin.
    d. How to rotate or translate a line.
    e. How to relate the rows of a rotation matrix to the axes of a new coordinate system
       This includes writing the coordinates of points relative to a new coordinate system.
       Given a position, lookat vector, an upvector, you should be able to write a matrix that transforms the world so that the viewer is at the origin, with the lookat direction in the *z* direction, and the upvector in the *y* direction.
3. Projection
    a. Perspective projection
        i. How to find the image coordinates of a 3D point
        ii. Some consequences of that, such as properties of vanishing points, knowing when a line in 3D does not project to a line in the image, …..
    b. Orthographic projection
        i. How to find the image coordinates of a 3D point.

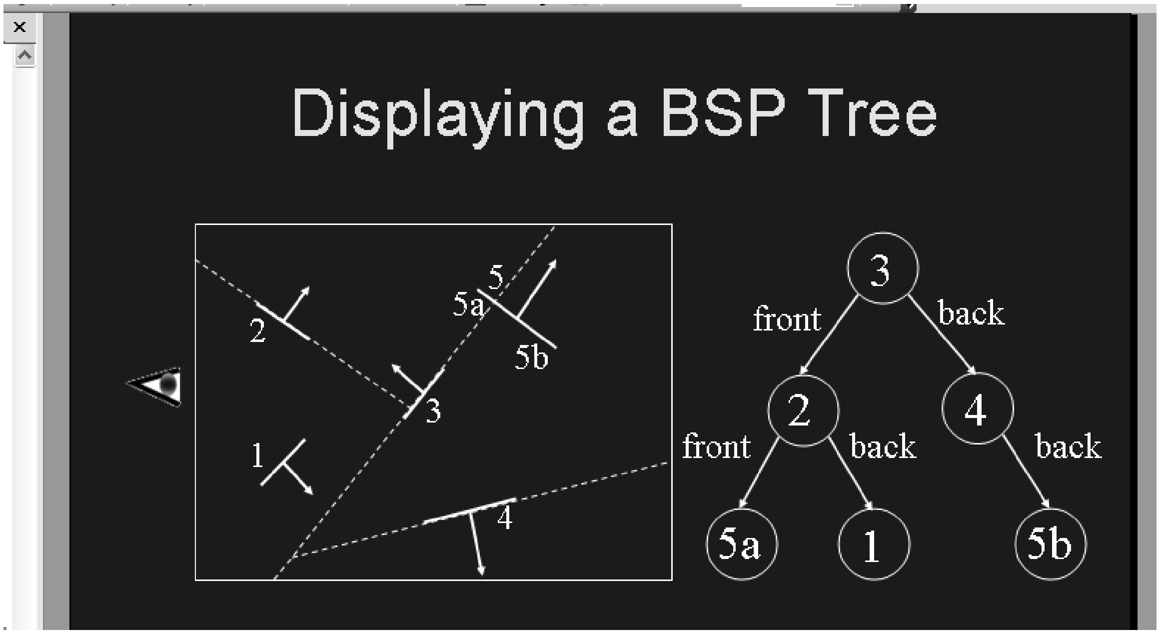
    ii. Consequences of orthographic projection, such as that it preserves parallelism and ratios of areas.

4. Intersections and Relations between lines and planes and points.
    a. How to use the equation of a line or plane to tell which side of the line/plane a point is on.
       Example: Given a plane and two points, are the points separated by the plane, or on the same side of it?
    b. How to find the intersections of spheres, axial rectanguloids, triangles.
    c. How to intersect a line with a plane, triangle, or sphere
5. Discretization
    a. How to represent a line with pixels.
    b. How to fill a polygon.
        i. How to tell if a point is inside a polygon.
            1. Implicit function test (testing that it is on the appropriate side of each line).
            2. Crossing number test.
        ii. Flood fill algorithm.
6. Sampling and Aliasing
    a. Creating an image by sampling
    b. Aliasing – what it means, what causes it.
        i. Spatial
        ii. Temporal
    c. Anti-aliasing
        i. Smoothing
        ii. Supersampling and smoothing
7. Interpolation
    a. Linear, bilinear
    b. Cubic
    c. Bezier curves -- Given three control points, you should know how to construct points on a Bezier curve, and know some of its basic properties.
8. Texture
    a. Perlin noise algorithm – General idea
9. Morphing and interpolation
    a. Morphing algorithm that you implemented.
    b. Describing points in a new coordinate system.
10. Color
    a. Superposition
    b. Color Spaces – CIE, RGB, CMYK, HSV.
11. Visibility
    a. Culling – Basic idea
    b. Painter's algorithm
        i. For example, how to tell if a triangle 1 is in front of, or behind, the plane that contains triangle 2, and how knowing this might help you in the painter's algorithm.
    c. Z buffer.

        i. Given the vertices of a 3D triangle, you should be able to figure out where they appear in an image (perspective projection), which 2D points are inside the triangle, and what their depth is.
    d. BSP Trees – The basic idea of the algorithm. See the problem below.
12. Lighting – Optics and Reflectance
    a. Ambient – What does this mean?
    b. Lambertian -- Know the equation for Lambertian reflectance.
    c. Specular –
        i. Know the equation for Phong reflectance.
        ii. Know how to figure out the direction of mirror reflection.
13. Shading – Know what the following mean.
    a. Gouraud
    b. Phong
14. Ray Tracing
    a. Shadows
    b. Finding rays from focal point to pixels.
    c. Given the intersection point between a ray and an object, know in which directions you need to cast additional rays.
15. Shadows
    a. Z-buffer algorithm.
    b. Casting a shadow with a directional or point source onto a plane.
16. Radiosity and photon mapping – Basic idea
17. Modeling
    a. Implicit shapes – Don't memorize which shapes have which equations (except you should know the equation for a sphere).
    b. Constructive solid geometry -- You need to anyway know how to tell whether a point is in a simple geometric object (eg., cube, sphere), and where a ray intersects it. You should understand how to use this to tell whether a point is inside, or a ray intersects, a more complex object made up of simple ones.
    c. Fractals -- Just the basic idea; no math (eg., don't worry about complex numbers).

Sample Problems (Note, some of these problems may be a bit more involved than ones I'd ask on a time-limited exam).

1. Suppose the camera's focal point is at (1,2,3), you are looking in the direction (1,1,1), and the focal length is 1. Give a ray from the focal point through the center of the image.
2. Suppose the camera has a focal point at (0,0,0), and you cast a ray through a pixel whose center is at (1/2, 0, 1). Does this ray intersect a sphere centered at (3, 2, 12) with a radius of 3?
3. Suppose the camera has a focal point at (0,0,0), and you cast a ray through a pixel whose center is at (1/2, 0, 1). This ray definitely intersects a sphere centered at (3, 2, 12) with a radius of 5. What are the coordinates and surface normal of the first point that the ray intersects?
4. Consider the BSP tree discussed in class:



a. Suppose we look at this scene from the right side, instead of the left. Use the BSP tree to determine the order in which the triangles should be rendered.
b. Construct a valid BSP tree starting with triangle 1 instead of triangle 3.

5. Suppose we construct a model by taking a sphere centered at (10,0,10) with a radius of 2, and subtracting a cylinder. The cylinder is centered also at (10,0,10). It has a circular cross-section with a radius 1 when we slice it with a plane that has a constant z value (eg., if we intersect the cylinder with the z = 10 plane, we'll have a circle centered at (10,0,10) with a radius of 1). The cylinder has a length of 5. Suppose we cast a ray from (0,0,0) in the direction (5,0,4). Will it intersect the model? If so, where?
6. Suppose we construct a Bezier curve, using the control points (0,0), (1,1), (2,0).
   a. Give four points that lie on the curve.
   b. What is the tangent of the curve at its beginning and ending points?
7. Suppose we construct a curve in the following way. We create a half circle starting at (0,0), passing through (1,1) and ending at (2.0), with a radius of 1. Then we connect this to a half-circle starting at (2,0), passing through (3,-3), and ending at (4,0), with a radius of 1. How smooth is this curve?
8. Suppose you want to model the reflectance properties of a blue car.
   a. What reflectance model might you choose, and what parameters might you use? Why? You might want to limit yourself to using the models we've discussed in class (eg., Phong, Lambertian).
   b. Suppose you have a flat piece of blue car hood. You shine a light on the car from an angle of pi/4. Then you take a few pictures of the car from different angles. How might you use these images to determine what parameters you should use to model its reflectance? How many images do you think you would need?

9. Create a simple scene with a viewer looking in a specific direction, one or two simple objects, and a point source of light. Then work out how to render or ray trace this scene. Figuring out what kind of scene will create a reasonable set of problems will also be helpful to your studying. This is an Uber-problem. If you can do this, you have studied much of what is needed.

a. Transformations
    i. Create a matrix that will transform the world so that the viewer is at the origin and the viewing direction is in the z direction.
    ii. Apply this matrix to an object in the scene to determine its coordinates after this transformation.
    iii. Apply the equations of perspective projection to determine the image location of the object.

b. Ray Tracing.
    i. Find the equation for the image plane.
    ii. Determine a ray that goes from the viewer to a vertex in the scene, and figure out where it intersects the image plane.
    iii. Determine a ray that goes through the center of the image, and figure out which object in the scene it first intersects.
    iv. Suppose a ray strikes a surface that is a bit shiny (that is, it reflects light with some Phong and some Lambertian reflectance), but it is not at all transparent. If we are allowing for multiple bounces and shadows, what rays will we cast from the point where the original rays hits the surface. Give the direction of each ray.

c. Lighting
    i. Find a ray from the viewer that intersects an object in the scene. Suppose the object is Lambertian and white. Determine the intensity it will produce in the image.
    ii. Now do the same, assuming the object has Phong reflectance.

d. Shading
    i. The surface normal of a vertex isn’t really well-defined. One way to define it is to take the average of the normals of the sides of the object that form the corner. Using this approach, find the normals for the vertices of a polygon in the scene.
    ii. As in (c), find the intensity that each of these vertices will produce in the image. Then determine the intensity that you will get in the middle of the polygon using Gouraud or Phong shading.