CS450/550 Midterm Exam Practice Problems

Note: These problems are designed to encourage you to think about the topics that I will put on the exam. They do not necessarily represent the format of the questions I’ll put on an exam. For example, I will typically not have you define terms, but may have matching or multiple choice questions that test your knowledge of the important terms.

1. Define the following terms. In some cases I’ve listed multiple, related terms on a single line.
   a. Raster
   b. Framebuffer, depth, resolution
   c. Rasterization
   d. Pixel
   e. Fragment
   f. Rods and Cones
   g. View Volume, Image Plane, Center of Projection
   h. Window, viewport
   i. Immediate and retained mode graphics
   j. Object, World and Eye Space (related to #2 below)
   k. Orthographic, Perspective views

2. The pipeline approach to image generation is non-physical. What are the main advantages and disadvantages of such a nonphysical approach?

3. For the following scene, draw a picture of the resulting image. The scene consists of...
   a. Unit cube centered at the origin as ‘lines’
   b. Camera with the following LookAt and view volume specification:

   ```
   point4 eye(0.0, 0.0, 2.0,1.0);
   point4 at( 0.0, 0.0, 0.0, 1.0 );
   vec4 up( 0.0, 1.0, 0.0, 0.0 );
   mat4 p = Perspective(45 , 1.0, 0.5, 2.0);
   ```

4. Draw the graphics pipeline and label the components as well as the input/output types for each component. Label the ‘space’ of the vertex after each component.

5. Derive the projection of a point P (x,y) onto the projection plane using a pinhole camera model.

6. Explain how the Z-Buffer Algorithm works

7. Demonstrate the BSP solution for hidden surface removal by creating a simple 2-D scene, building the tree, and traversing to show the resulting render order.

8. Describe the difference between an attribute variable and a uniform variable in a glsl shader.
9. Which of the following is not an affine transformation?
   a. Rotation
   b. Scale
   c. Projection
   d. Reflection

10. Which of the following sequence of commands produces the scene shown below?
   a. Rotate(45, 0, 0, 1)
     Translate(a, b, 0)
     Scale(0.5, 0.5, 0.5);
   b. Rotate(45, 0, 0, 1)
     Scale(0.5, 0.5, 0.5)
     Translate(a, b, 0);
   c. Translate(a, b, 0)
     Rotate(45, 0, 0, 1)
     Scale(0.5, 0.5, 0.5)

11. a,b (4,3)
a) Give the sequence of transformation to move the object from it’s position and orientation in A to it’s position and orientation in B.

b) Write the corresponding equation in terms of a matrix multiplication. Represent a translation with T, rotation with R, etc. Label them with subscripts if there is more than one translation, rotation, etc.

c) How did you visualize the transformation : in a grand fixed coordinate system or a moving coordinate system

12. The human visual system detects light with sensors called:
   a. Cylinders and cubes
   b. Rods and cones
   c. Photocells
   d. Lens cells

13. Which of the following is not part of the Rendering Pipeline
   a. Modeling Transformations
   b. Colorizer
   c. Projection and Assembly
   d. Rasterizer

14. Given a frame buffer with the following characteristics
    1024x768 addressable pixels
    24 bit color (8 red, 8 green, 8 blue)
    a. How much memory is required to store the framebuffer?
**Answers**

1. See book
2. Pro: polygon independence therefore you can pipeline the vertices/primitives for efficiency. Con: no global effects (shadows, reflections, etc.)
3. 

![Diagram](image)

4. See pipeline.pdf
5. See Chapter 2
6. See board example that I did in class for a demonstration. In general, the depth of fragments is maintained and used to compare against depth values stored in the Zbuffer (initialized to \(-\text{bigNum}\)). If the current fragment’s depth is larger (ie. closer to eye) than that in the z-buffer, the framebuffer color at that fragment location is overwritten with the color of the current fragment, otherwise it is not.
7. See example from class
8. Attribute variables can change from one vertex to the next while uniform variables are constant for an entire primitive.
9. C. projection
10. C
11. `glTranslatef(4, 3, 0);`
    `glRotatef(90, 0, 0, 1);`
12. b
13. b
14. 1024 x 768 x 24