CPSC 352 Final Exam Information

- Date: **Wednesday, May 20, 9:00 AM**
- Exam is **cumulative.** There will be more emphasis on the last 1/3.
- Sources: Text, lectures chapters 1-9; PowerPoint lecture notes
- Format: Similar to tests 1 and 2. Short answer (e.g. "Explain the Gouraud Shading algorithm"); some more concrete problems with respect to linear algebra (e.g. show how to find the angle between two vectors [given] using the dot product).
- Use the review sheets from tests 1 and 2 (linked on the syllabus web page)
- Additional topics from chapters 7, 8, and 9:

Text, chapter 7: discrete techniques

1. Texture mapping
   a. Use of texture coordinates
   b. Render modes: decal, replace, modulate, blend
   c. Need for interpolation. Nearest vs. linear interpolation
   d. Mipmaps
2. Environment mapping
3. Bump mapping
4. Displacement mapping
5. OpenGL's buffers: Color, depth, stencil, accumulation. What each is used for. How (conceptually) you would do things like fog, anti-aliasing, etc. with buffers.
6. Screen-door transparency
7. Picking using pseudo-colored image
8. Compositing
   a. Example uses
   b. Billboarding

Chapter 8: The Rendering Pipeline

1. Front-end vs. back-end processing: what is done in each?
2. Parametric representation of lines
3. Clipping
   a. How to compute the intersection of a line (x1,y1)-(x2,y2) with a line, e.g. y=y1
   b. Cohen-Sutherland clipping algorithm
   c. Liang-Barksy clipping algorithm
   d. Sutherland-Hodgeman clipping
   e. Pipelined clipping
4. Scissoring
5. Rasterizing a line
   a. DDA Algorithm
   b. Bresenham's algorithm
6. Rasterizing polygons
   a. Inside-outside testing
   b. Winding number, inside-outside rules
   c. Flood fill
   d. Scan-line approach
   e. Handling singularities (e.g. a scanline passing through a vertex
7. Anti-aliasing
8. Halftoning
9. Dithering

    a. Floyd-Steinberg error-diffusion dither algorithm
10. Color Quantization
    a. Heckbert median-cut color quantization algorithm

Chapter 9: Hierarchical graphics, modeling, and animation

1. Primitives such as points, lines, polygons, ellipses, NURBS, particles, meshes, skin and bones, subdivision surfaces, algorithmic primitives
2. Mesh deformations: e.g. modeling landscapes, cloth
3. Modeling tools: sweep, revolution, extrusion, stitching, blending
4. Hierarchical modeling.
    a. Hierarchical object as DAG, with transformations on edges
    b. Algorithm for rendering such models
5. Kinematics
6. Inverse kinematics
7. Keyframe animation
8. Morphing
9. Tree-based data structures for representing objects
    a. CSG Tree
    b. BSP-tree and its use for hidden surface removal
    c. Algorithms for building, traversing BSP-trees
    d. Quadtrees, octrees
10. Visibility preprocessing algorithms

Sample Questions

1. What is texture mapping? Why do we have to give texture coordinates when using texture mapping in OpenGL? Explain the methods that OpenGL uses to deal with aliasing.
2. Explain how billboarding might be used to render a tree with only two polygons. How does this technique use the stencil buffer?
3. Explain bump mapping.
4. Why are hierarchical DAG models used to represent complex objects? What is the purpose of the transformation associated with each edge of the DAG? What is the algorithm for rendering such models? What is the difference between kinematics and inverse kinematics? How would inverse kinematics be combined with keyframing for animating the model?
5. How could you use keyframing to animate a 3D morph of an object defined with curved surfaces such as NURBS?
6. Explain the BSP-tree insertion and traversal algorithms. How is it used for hidden surface removal in the absence of a depth buffer?
7. Given a line segment from (x1,y1) to (x2,y2), how would you compute the intersection parameter with the y=y0 line?
8. What are the Cohen-Sutherland “outcodes” and how are they used in their clipping algorithm?
9. What is “scissoring” when would you want to (or not want to) use it?
10. What is color quantization? Why is it necessary? Explain the Heckbert median-cut algorithm for color quantization. Explain how it can be combined with the Floyd-Steinberg error-diffusion dither algorithm.