Notes on midterm exam which is on TUESDAY OCTOBER 29

This will be a 75 minute written test covering everything we’ve done up to but not including texture mapping. In the textbook, the relevant sections are:

- Chapters 1-5 (except 5.4 on hierarchy)
- Chapter 6 and Appendix A (code examples)
- Chapter 10 – sections 10.1, 10.2 only
- Chapter 11 (except 11.2.1; also don't worry about the derivation of Frustum in 11.3)
- Chapter 12 (except 12.3.1 on texture viewport)
- Chapter 14 (except 14.4 on anisotropy)

Some topics and review questions are listed below. The written homework from the beginning of the course provides some good examples for exam questions on 2D transformations. Some other ideas for review or exam questions are shown in italics.

You may need to write short bits of C code but you do not have to memorize specific OpenGL functions (though you should have a basic idea how we use them). You should understand the LookAt(), Ortho(), and Perspective() functions. You’ll need to be able to write shader code at roughly the level of detail of the examples we’ve done.

The graphics pipeline:

(Model -> )
  Vertex processing -> Primitive assembly -> Rasterization -> Fragment processing
  (-> framebuffer)

Role of shaders in the pipeline

- Could you write a vertex shader that uses three adjacent vertices to calculate a normal vector? Explain briefly.
- Could you write a shader to modify the way clipping is done?

Linear interpolation

Basic color representation

Clip coordinates, clipping volume, depth testing, left handedness

- Which point is closer to the viewer in clip space, (0.0, 0.0, -0.5) or (0.0, 0.0, 0.5)?
- Why does clipping potentially create new polygons?

Vertex attributes, “in” and “out” variables

- What is the role of an “in” variable in a vertex shader?
- What is the role of an “in” variable in a fragment shader?
- Write a fragment shader that colors each pixel with a greyscale value obtained from the average of the red, green, and blue values of an in variable called fColor.
Write a complete shader program (vertex shader and fragment shader) that will color pixels according to their world space height (y coordinate). Assume that the world space vertex position is an “in” variable in the vertex shader and that there are uniform variables representing a minimum and maximum possible y. Values at or below the minimum should be dark grey, values at or above the maximum should be white. You can also assume that there is a uniform variable 'transform' representing an appropriate view/projection.

OpenGL primitives
- Given a set of vertices, show what is rendered using glDrawArrays(GL_LINES, ...), glDrawArrays(GL_TRIANGLE_FAN, ...), etc.
- Indexed rendering
  - Given a bunch of vertices, write down the indices that would be used to render it using glDrawElements(GL_TRIANGLES, ...), or b) as a wireframe using glDrawElements(GL_LINES, ...),

Using buffers and VAOs in OpenGL
Linear transformations as matrices $M = [f(e1) \ f(e2) \ f(e3)]$, where $f$ is the transformation and $e1$, $e2$, $e3$ are basis vectors.

Using homogeneous coordinates to represent translations; points vs vectors
Affine transformation = linear transformation followed by a translation
- Prove that the matrix for an affine transformation always has (0 0 0 1) in bottom row.

2D rotation, translation, scaling
- Given a 2D triangle with vertices (0, 0), (2, 0), (2, 1), sketch the triangle after
  a) translation by (4, 5) followed by rotation by 90 degrees
  b) rotation by 90 degrees followed by translation by (4, 5)
  c) translation by (4, 5) followed by scaling by 2 in the y direction
  (show the approximate new coordinates in each case, e.g., on graph paper)
- For each of a, b, c write down the transformation matrix

Frame = basis vectors, plus an origin
Standard matrices: Translate(), RotateX(), RotateY(), RotateZ(), Scale()
- Suppose an object is centered at (x, y, z) and we want to rotate it through angle r about its z axis. Write down the necessary matrix in terms of the standard matrices (e.g. Translate(), etc.) (Don’t multiply them out!)
- Suppose an object is centered at (x, y, z) and we want to rotate it through angle r about a line through its center, where the line is directed at angle theta from the positive y-axis and angle phi from the positive x-axis, as in spherical coordinates. Write down the necessary matrix in terms of the standard matrices.

Intrinsic vs extrinsic transformations
Inverting a product of standard transformations
- An affine transformation consists of scaling by \((1, 2, 1)\), then a translation to \((1, 2, 3)\). Give the matrix for this transformation, and calculate its inverse. (You don’t have to multiply out the result.)

Suppose \(M\) is an affine matrix whose right column is \([x \ y \ z \ 1]^T\), where \(x, y, \text{ and } z\) are nonzero. Under what conditions is it possible that the right column of \(M^{-1}\) is exactly \([-x \ -y \ -z \ 1]^T\) ?

Changing frames; transformation \(M\) vs. change of frames \(M^{-1}\)
- Suppose you are working in 2D and your new frame has origin \((5, 10)\) and that its \(x\)-axis is given by the vector \([0, 1]\) and its \(y\)-axis is given by \([-1, 0]\).
  a) What are the coordinates of the point \((1, 2)\) with respect to this new frame? Draw a picture.
  b) Find the "change-of-frame" matrix (that is, the matrix \(S\) such that if \(c\) is a coordinate vector for a particular point w.r.t. the original frame, then \(Sc\) is a coordinate vector for the same point w.r.t. the new frame.)

- An object is centered at \((2, 3, 4)\). What transformation will make the object twice as big without moving its center?
- Suppose some frame \(F\), is obtained from world frame by a matrix \(A\). What transformation will rotate an object 30 degrees counterclockwise about the z-axis of frame \(F\)?
- Suppose you are given coordinates for three orthonormal vectors \(x = [x_1 \ x_2 \ x_3]^T\), \(y = [y_1 \ y_2 \ y_3]^T\), and \(z = [z_1 \ z_2 \ z_3]^T\), plus a point \(p = [p_1 \ p_2 \ p_3 \ 1]^T\) that together describe a frame that you would like to use as your eye coordinate system. Write down the view matrix.
- Write the C function \texttt{LookAt(eye, at, up)}, assume you have a function for computing cross products and basic functions such as \texttt{RotateX()}, \texttt{Translate()}, etc, as needed

Scale and bias
- Create a 2D affine transformation \(M\) such that given a vector \(c = [x, y, 1]^T\), \(Mc = [x', y', 1]^T\), where \(x'\) is \(x\) converted from Celsius to Fahrenheit and \(y'\) is \(y\) converted from Fahrenheit to Celsius. (Write \(M\) as the product of a scale matrix followed by a translation matrix.)

Orthographic projections, view volume
- Describe what the C function \texttt{Ortho(lef, right, bottom, top, near, far)} does?
- Why do we have to specify near and far clipping planes?
- Derive this matrix and write the function.

Perspective transformation in OpenGL (perspective normalization)
- What’s a frustum?
- What happens to the z-values in an OpenGL perspective transformation? How is this related to z-fighting?

Perspective division
- What is it? Where does it occur in the pipeline?
Roles of the model, view, and projection matrices
- Describe an example showing why we might want to keep them separate

Coordinate systems used in OpenGL: model, world, eye/view, clip, NDC, window, viewport/screen
- What coordinate system are we working in during fragment processing? What is the range of x, y, and z values?
- Where does homogeneous division occur in the pipeline?
- What operation transforms from eye coordinates to clip coordinates?

Normal vectors
How to calculate face normals from vertex coordinates
Suppose you have a class representing a model with methods GetVertices() that returns the vertices as an array of vec3, and GetNumVertices() that returns the length of this array. Assume the vertices are appropriately ordered to render the model using GL_TRIANGLES. Write a C function GetNormals(vec3 * array, int numVertices) that returns an array of normal vectors (face normals) corresponding to the vertices.

Lambert shading
- Why do we have to normalize L and N in our lighting calculation?

Phong (ADS) lighting model
- Suppose at a given vertex, L points to the light and N is the normal vector and V points to the view point Assume all three are normalized. Sketch a picture. Write down the three components of the lighting model as accurately as you can.

Directional vs point lighting
The normal matrix \((M_{3x3}^{-1})^T\) (where \(M_{3x3}\) represents the upper-left 3x3 submatrix of M)
- Give a concrete example showing that a just using \(M_{3x3}\) might not do the right thing to normal vectors.
- Under what conditions does \(M_{3x3}\) correctly transform normal vectors?

Gouraud and Phong shading
- Briefly explain the difference between Gouraud and Phong shading
- Explain the nature of the artifacts that occur using Gouraud shading with a specular component
- Why do we have to normalize after interpolating vectors? Give an example showing that interpolating between two vectors of length 1 does not necessarily give you a vector of length 1.
- We normally do lighting calculations in eye coordinates. Could you do it just as well in world coordinates? Explain what might be easier/harder/different.
- Write a complete shader pair for per-fragment shading using a point light. Assume that the light position is given in world coordinates as a uniform variable, and that the model, view, and projection matrices are given as uniform variables.

The viewport transformation
The z-buffer algorithm