University of California San Diego
Department of Computer Science
CSE167: Introduction to Computer Graphics
Midterm Examination #1
Thursday, October 24th, 2013
Instructor: Dr. Jürgen P. Schulze

Name:________________________________

Please put your name or initials at the top of every sheet before beginning the exam so that we don’t have to guess when we find loose sheets.

Your answers to the exam questions must include all steps of your derivations.

This is closed book exam. You may not use electronic devices, notes, textbooks or other written materials.

There are ten questions on 6 sheets for a total score of 100 points.

Good luck!

Do not write below this line

Exercise	Points
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

1. Geometric Transformations (10 Points)

We have discussed rotation, translation and scaling transformations in class. For this question, we will be using the following notation:

- R(theta, axis) denotes a rotation by theta degrees around the specified (x, y, z) axis. Use the right hand rule for the direction of the rotation.
- T(dx, dy, dz) denotes a translation by (dx, dy, dz) units.
- S(sx, sy, sz) denotes a scaling by (sx, sy, sz) units.

a) Start with a 2x2 square in the x/y plane, centered at the origin. Draw a picture that shows what happens to this square when you first apply rotation R(90 , z) followed by a scaling of S(1, 2 , 1). Draw each step. What happens if you reverse the order of operations? Is this true for all R,S? Explain. (4 points)

b) Draw a picture that shows what happens if you translate a 2x2 square by T(1 , 1 , 0) followed by a rotation of R(45 , z). What happens if you reverse the order of operations? Is this true for all R,T? Explain. (3 points)

c) Sometimes we want to scale an object about an arbitrary center point (cx, cy, cz) to simplify user interaction or to get different visual effects. Use diagrams and equations to show how to perform this composite geometric transformation. (3 points)

2. Geometric Models (10 Points)

As in problem #1, we will be using the following notation:

- R(theta , axis) denotes a rotation by theta degrees around the specified (x , y , z) axis.
- T(dx , dy , dz) denotes a translation by (dx , dy , dz) units
- S(sx , sy , sz) denotes a scaling by (sx , sy , sz) units.

Assume that we have the coordinates of all polygons in your geometric model stored in three arrays X[u][v] , Y[u][v] , Z[u][v] , and you would like to transform this object so it fits entirely into a cube going from [- 1..1] in X , Y , Z coordinates, without changing the shape of the object.

a) Describe the sequence of steps you would use to calculate the geometric transformation above. Be specific about which of the R, T, S operations you would perform, and how you would calculate the parameters to each of these functions. (6 points)

b) What is the formula for translating a point by (dx , dy , dz) units? Show how this can be implemented using a 4x4 matrix multiplication. (2 points)

c) What is the formula for uniformly scaling a point by a factor of s? Show how this can be implemented using a 4x4 matrix multiplication. (2 points)

3. Projection (10 Points)

a) A vertex (point) is drawn at the origin of the object coordinate system.
It is viewed through a camera whose inverse transformation matrix is:

$$\begin{bmatrix} 2 & -1 & 0 & -2 \\ 1 & 2 & 0 & -2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The object that the vertex is part of is drawn with transformation matrix:

$$\begin{bmatrix} 0 & -1 & 0 & 5 \\ 1 & 0 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

This simple projective transformation matrix is used:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Where does the point appear in screen coordinates? Give the x/y position. (6 points)

b) For each of the following statements, mark them with a P if it is true of a perspective projection, O if it is true of an orthographic projection, B if it is true of both orthographic and projective, and N if it is true of neither. (Use P or O only if the property is true of only one type of transformation). (4 points)

__ Straight lines are mapped to straight lines

__ Distances are preserved

__ Angles are preserved

__ Far away objects appear the same size as closer ones

4. View Volumes (10 Points)

Given the following typical GLUT reshape function:

```
void reshapeCallback(int w, int h)

{
  glViewport(0, 0, w, h);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(45.0, float(w)/float(h), 1.0, 100.0);
}
```

The main() function of the program includes a typical GLUT initialization sequence, and the following lines:

```
  glutInitWindowSize(512, 512);
  glMatrixMode(GL_PROJECTION);
  glLoadIdentity();
  gluPerspective(45.0, 512.0/512.0, 1.0, 100.0);
```

a) What does the reshape function do when you use the mouse to resize the window? (2 points)

b) What would happen if you removed the glViewport(0, 0, w, h) command from the reshape function, compiled the program, and used the mouse to resize the window? (2 points)

c) What would happen if you used the mouse to resize the window if the reshape function had glViewport(0, 0, w, h) as the only single statement in it? (2 points)

d) What would happen if you used the mouse to resize the window (at least twice) after removing the glLoadIdentity( ) statement from the reshape function? (2 points)

e) What would happen if you changed the 45.0 in the gluPerspective function to 60.0? (2 points)

5. Rendering Optimization (10 Points)

a) What is the purpose of triangle backface culling? On average, what percentage of the triangles of a scene can be culled this way? (2 points)

b) Given a triangle with vertices p0, p1, p2, give a mathematical description of how to determine if the triangle is back facing. (3 points)

c) List two ways the points of a triangle could be arranged in which would make it subject to degenerate culling. (2 points)

d) Describe what the general idea of view frustum culling for triangles is. (3 points)

6. Depth Buffer (10 Points)

Discuss the depth buffer (z-buffer) algorithm.

a) Describe the algorithm. Write down pseudo-code for it. (3 points)

b) What is its purpose? (1 point)

c) What requirements does it have on the graphics hardware? (2 points)

d) What are its advantages over competing algorithms? (2 points)

e) What are its disadvantages relative to competing algorithms? (2 points)
(Be specific about which competing algorithms you are discussing.)

7. Color (10 Points)

Sid wants to render an outdoor scene with a fog effect. He observes that fog has the following characteristics:

- it has a color of its own, usually white but sometimes other colors
- shapes are faded to the fog color with distance

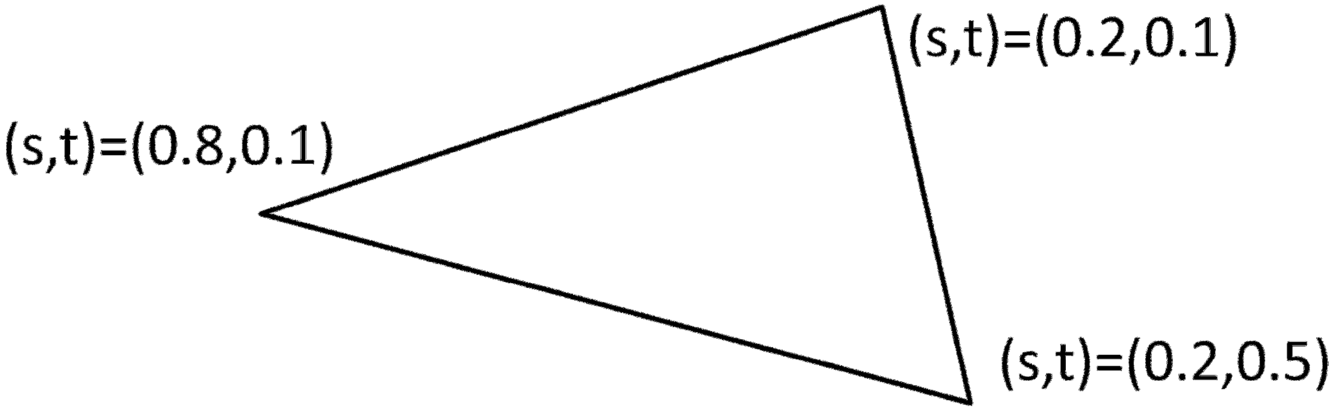
Help Sid write a **fog**(*d*, *src_color*, *fog_color*, *fog_density*) function that takes 4 parameters:

- *d*: the distance to an object
- *src_color*: the original (shaded) color of the object
- *fog_color*: the color of the fog
- *fog_density*: how quickly shapes fade to the fog color – greater density implies a quicker fade

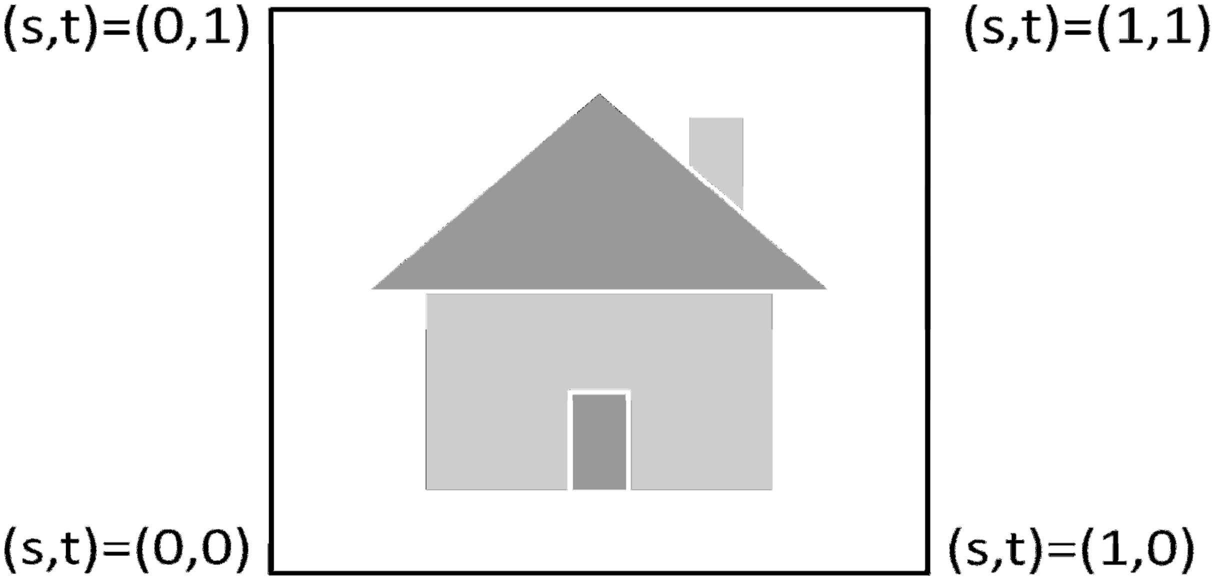
fog(...) returns the perceived color of the object seen through the fog. Write either a math formula or pseudocode for a plausible **fog**(...) function. (There is no single correct answer. Your solution should satisfy Sid's two criteria at least.)

8. Texture mapping (10 Points)

Given a triangle with associated texture coordinates (s,t) for its vertices

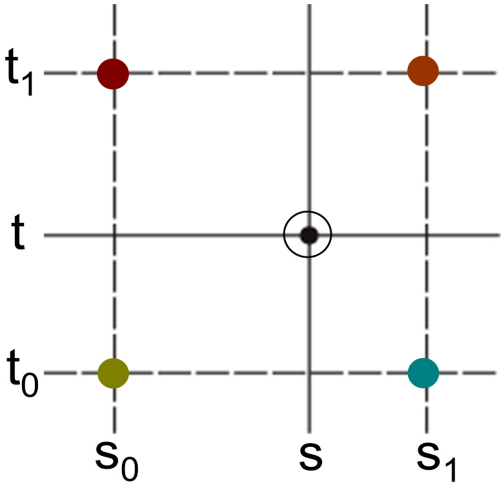


and a simple texture with its corner coordinates given:



a) Sketch into the triangle what it will look like after texture mapping. (4 points)

b) Given interpolated texture coordinates (s, t) at the current pixel, the closest four texels in texture space are at: (s_0,t_0), (s_1,t_0), (s_0,t_1), and (s_1,t_1), as illustrated in the picture below. The colors of the closest four texels are $tex(s_0,t_0)$, $tex(s_1,t_0)$, $tex(s_0,t_1)$, and $tex(s_1,t_1)$.



Derive a formula or pseudo-code for the color c of the pixel when nearest neighbor interpolation is used. (2 points)

Derive a formula or pseudo-code for the color of the pixel when bi-linear interpolation is used. (4 points)

9. Shading (10 Points)

a) Both Gouraud and Phong Shading interpolate along polygon edges to compute intensities. But the two shading models interpolate different things.

1) What does Gouraud Shading interpolate along edges? (1 point)

2) What does Phong Shading interpolate along edges? (1 point)

b) What is the difference between Phong Shading and the Phong Illumination Model? (2 points)

c) What two parameters does a Bidirectional Reflectance Distribution Function (BRDF) take in, and what does it calculate? (3 points)

d) Which three components does the simplified illumination model OpenGL uses consist of? (3 points)

10. Shader Programs (10 Points)

a) Assuming that both vertex and fragment programs have been loaded for rendering a triangle. How many times will the vertex program get executed, how many times the fragment program? (3 points)

b) Name two examples each for what vertex and fragment programs can be used for. (4 points)

c) What is the difference between the storage classifiers “uniform” and “varying” in GLSL? (3 points)