CSCI 4250/5250 Take Home Test 2 (100 pts) **Name** Sean Smyth

Due: midnight, Sunday, Dec 3rd

You are required to type your answers for these questions.

Open Book & Notes – You are on the honor system and do this exam with no help from any other person. When you type your name below, you are indicating that you have adhered to these restrictions. Turn in this page with your answers to the questions. Turn in your typed answers in pdf file to the D2L Dropbox named "Test 2".

I, Sean Smyth - Sean Smyth**, worked all of the problems on this test completely on my own without any assistance from any other person. My only resources were the textbook, online course material, my notes, and Internet sources.**

1. **(10 pts) Fill in the blanks:**

   a. The ___view___ matrix transforms the object drawn from the world coordinate into the camera coordinate.

   b. The ___model view___ matrix is saved and restored using Push and Pop operations in WebGL during 2D and 3D drawing involving transformations.

   c. The ___depth___ buffer is used for hidden surface removal.

   d. In WebGL lighting model, light scattered so much it is difficult to tell the source is called A___mbient___ light. Light coming from one direction tending to bounce off a surface in mostly 1 direction is called S___pecular___ light. Light coming from one direction but is scattered in many directions is called D___iffused___ light.

   e. In texture mapping, the s and t values of the texel coordinates (s, t) is limited to the range between ___0___ and ___1___.

   f. In perspective projection, the vertices further away from the viewer appear to have smaller x and y coordinate values, this is the result of applying the step: P___erspective___ D___ivision___, in the graphics pipeline.

2. **(10 pts) Short answer questions:**

   a. Compute the angle between these two vectors: **v1**=(2, 4, 2) and **v2**=(3, 5, 2)?

Dot product $= 2\cdot3+4\cdot5+2\cdot2 = 6+20+4=30$

$|a| =$ magnitude $= \sqrt{2^2+4^2+2^2} = \sqrt{4+16+4} = \sqrt{24} = 4.8989$

$|b| = \sqrt{3^2+5^2+2^2} = \sqrt{9+25+4} = \sqrt{38} = 6.1644$

$\cos\theta = \frac{v1\cdot v2}{|a|\cdot|b|} = \frac{30}{30.1988} = 0.9934$

$\theta \boxed{= 84.3°}$

   b. Given the coordinates of two points, how does the graphics system figure out the location of all the points in between these two vertices? Explain with the example: given two points: **A**=(2, 4, 2) and **B**=(3, 5, 2), compute the point P on the line segment AB, where P is at 1/4 the way from A to B? (i.e., t=1/4)

Using the equation $P=(1-t)A+tB$ where t is between 0 and 1, 0=t being point A's position, and t=1 being point B.

$P=(1-0.25)A+0.25B$ for each corresponding coordinate in A, B.

$P_0=(0.75)2+0.25\cdot3=2.25$

$P_1=(0.75)4+0.25\cdot5=4.25$

$P_2=(0.75)2+0.25\cdot2=2$

$\boxed{P=(2.25, 4.25, 2)}$

   c. Given three points on a plane: A(3, 2, 1), B(3, 4, 2), and C(2, 4, 2), compute the normal vector to this plane, i.e., the vector perpendicular to the plane containing the triangle ABC?

planar normal $= V_{AB}\times V_{AC}$

$V_{AB} = B-A$

$V_{AC} = C-A$

$V_{AB} = (3,4,2)-(3,2,1) = (0,2,1)$

$V_{AC} = (2,4,2)-(3,2,1) = (-1,2,1)$

$$V_{AB}\times V_{AC} = \begin{bmatrix} i & j & k \\ 0 & 2 & 1 \\ -1 & 2 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} 2 & 1 \\ 2 & 1 \end{bmatrix} i - \begin{bmatrix} 0 & 2 \\ -1 & 2 \end{bmatrix} j + \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix} k = 0i - 2j + 1k$$

normal vector = [0, -2, 1]

3. (35 pts) Given:

- the vertices describing the 3D shape in the world coordinates,
- the camera location, look at position and up direction specified in the lookAt function, and
- the orthographic projection setup,

answer the following questions:

a. What is the view matrix generated at line 1 ? Show all steps in how the view matrix is derived.
b. What is the modelviewMatrix matrix generated from lines 2, 3, and 4 ?
c. What is the projection matrix generated at line 5? Show all the matrix is derived.
d. After the modelviewMatrix and projectionMatrix have been sent to the vertex shader, they are used to transform the individual vertices in each object into their final coordinates in the clip-coordinates. Compute the coordinates of vertices B in the clip coordinates.

```
modelviewMatrix = lookAt(0, 0, 8, 0, 0, 0, 0, 1, 0);      // line 1
t=translate(1, 0, 0);                                     // line 2
r=rotate(30, 0, 1, 0);                                    // line 3
modelviewMatrix = mult(mult(modelviewMatrix, r), t);      // line 4
projectionMatrix = ortho(-5, 5, -6, 6, 2, 10);            // line 5
```

eye at up

L R B T N F

(c) Projection =

$$\begin{bmatrix} \frac{2}{10} & 0 & 0 & 0 \\ 0 & \frac{2}{12} & 0 & 0 \\ 0 & 0 & \frac{-2}{8} & \frac{-12}{8} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} .2 & 0 & 0 & 0 \\ 0 & .1\overline{6} & 0 & 0 \\ 0 & 0 & -.25 & -1.5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(a) lookat (0,0,8,0,0,0,0,1,0) = modelviewMatrix

$n = eye - look$

$u = up \times n$

$v = n \times u$

$$V = \begin{bmatrix} U_x & U_y & U_z & d_x \\ V_x & V_y & V_z & d_y \\ n_x & n_y & n_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$d = (-e \cdot u; e \cdot v, -e \cdot n)$

$e = eye - O$ (origin = 0,0,0)

$n = (0,0,8) - (0,0,0) = (0,0,8)$

$$u = \begin{vmatrix} i & j & k \\ 0 & 1 & 0 \\ 0 & 0 & 8 \end{vmatrix} = (8,0,0)$$

$$v = \begin{vmatrix} i & j & k \\ 0 & 0 & 8 \\ 8 & 0 & 0 \end{vmatrix} = (0,0,64)$$

normalized u, v, n

$U_{normal} = (1,0,0)$

$V_{normal} = (0,0,1)$

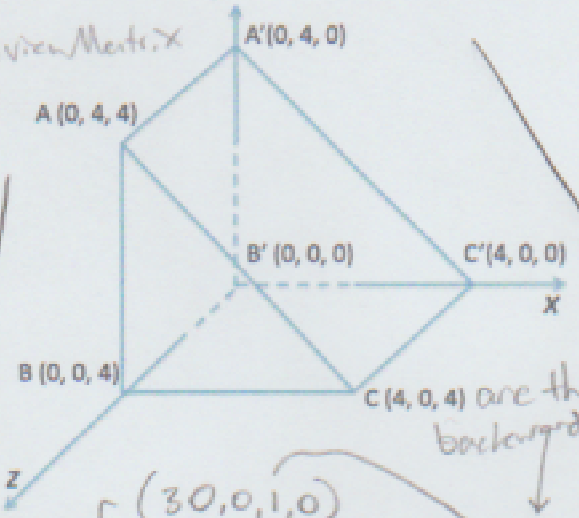
$n_{normal} = (0,0,1)$

$d_x = -eye.u = (0,0,-8).(1,0,0) = 0$

$d_y = -eye.v = (0,0,-8).(0,0,1) = -8$

$d_z = -eye.n = (0,0,-8).(0,0,1) = -8$

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -8 \\ 0 & 0 & 1 & -8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



are these 2 backwards?

(b) r (30,0,1,0)

$$= \begin{bmatrix} \cos 30 & -\sin 30 & 0 & 0 \\ \sin 30 & \cos 30 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos 1 & 0 & \sin 1 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin 1 & 0 & \cos 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

V

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -8 \\ 0 & 0 & 1 & -8 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} .8659 & -0.5 & .0151 & 0 \\ .4999 & .8660 & .0087 & 0 \\ -0.0175 & 0 & .9998 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow \begin{bmatrix} .8659 & -0.5 & .0151 & 0 \\ -0.0175 & 0 & .9998 & -8 \\ -0.0175 & 0 & .9998 & -8 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

???

$$= \begin{bmatrix} .8659 & -0.5 & .0151 & .8659 \\ -0.0175 & 0 & .9998 & .4999 \\ -0.0175 & 0 & .9998 & -.0175 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \boxed{mvM}$$

(d) I know this is wrong because shouldn't be 0, but this is all I can do with what I've calculated.

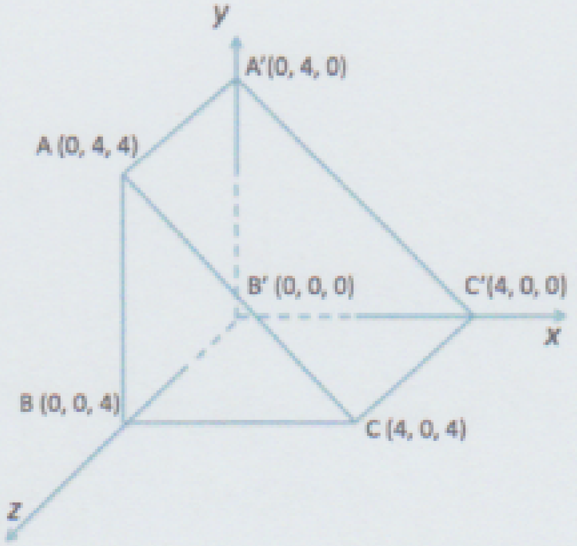
$$P \times mvM = \begin{bmatrix} .2 & 0 & 0 & 0 \\ 0 & .16 & 0 & 0 \\ 0 & 0 & -.25 & -1.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} .8659 & -.5 & .0151 & .8659 \\ -.0175 & 0 & .9998 & .4999 \\ -.0175 & 0 & .9998 & -.0175 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M = \begin{bmatrix} .1731 & -.1 & .0302 & .1731 \\ -.0028 & 0 & .16 & .08 \\ .0044 & 0 & -.25 & -1.4956 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$B' = (.1731, .08, -1.4956, 1)$

$B = (.2939, .72, -2.4956, 1)$

4. (10 pts) Assuming a perspective projection is used to view the object defined below:



modelviewMatrix = lookAt(0, 0, 8, 0, 0, 0, 0, 1, 0); *// the camera/eye is set up the same way // as in problem 3*

projectionMatrix = **frustum**(-6, 6, -6, 6, 2, 12);

L R B T N F

Compute the x and y coordinates of the points A and A' as they project on the near plane.

wasn't necessary, whoops

Aspect = (right − left)/(top − bot) = 12/12 = 1

View Angle = 2* arctan(½ *(top − bot)/N) = 2* arctan(.5(12)/2) = 2* tan⁻¹(3) = 143.13°

perspect(143.13, 1, 2, 12)

$(B')'_x = N \times B'_x / (-B'_z) = 2 \cdot 0/0 = 0$

$(B')'_y = N \times B'_y / -B'_z = 0$

$(B')' = (0,0)$

$(B)'_x = N \times B_x / -B_z = 2 \times 0/4 = 0$

$(B)'_y = N \times B_y / -B_z = 0$

$(B)' = (0,0)$

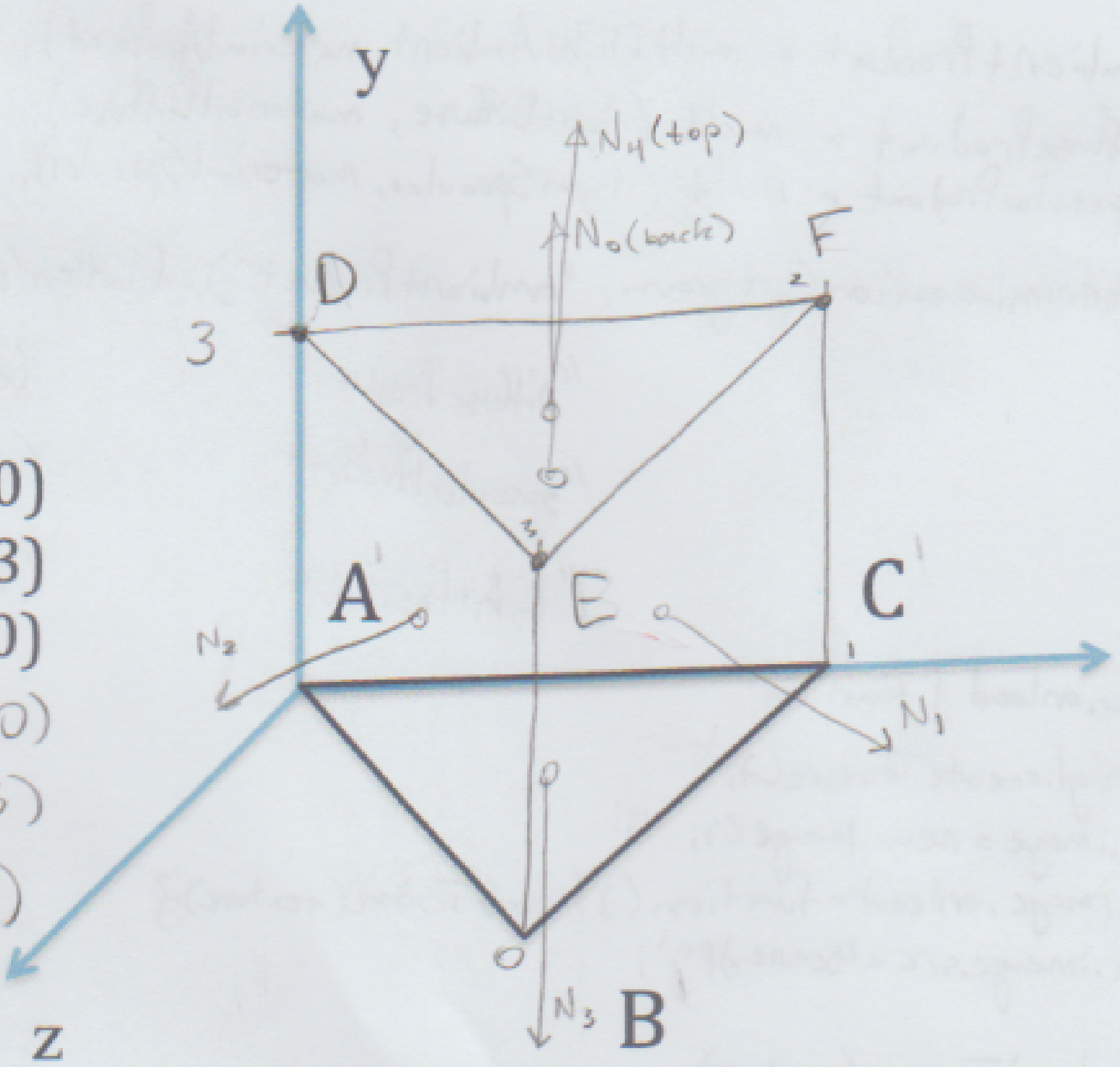
5. (35 points) An extruded shape is formed from the base triangle shown below. The height of the extruded shape is 3.

   a. Define the extruded shape in terms of the vertex list, normal list, and face list. When normal to a face is not readily computable, apply Newell's method for computation. Show each list in a table format as discussed in class.
   b. Suppose all the relevant data, e.g., the vertex positions, the faces, and the normals have all been stored in the appropriate arrays and pushed onto the vertex shader, show all the relevant WebGL code (in .js file) to setup proper lighting and object material properties to display a blue and shiny extruded triangle. Use a white directional light with light direction set to [4, 2, 4, 0].
   c. Show WebGL code needed in .js file to put an image "scene.jpg" (the size of the image is not a power of two) onto each side of the extruded shape as 2D texture.

a) vertex list

Vertex	x	y	z
0	0	0	0
1	2	0	3
2	4	0	0
3	0	3	0
4	2	3	3
5	4	3	0

0 A=(0, 0, 0)
1 B=(2, 0, 3)
2 C=(4, 0, 0)
3 D=(0,3,0)
4 E=(2,3,3)
5 F=(4,3,0)



normal list

normal	n_x	n_y	n_z
0	0	0	-1
1	9	0	-4
2	2	2	2
3	0	-1	0
4	0	1	0

guess

$n_i = (i+1) \bmod 4$

$$N1x = \sum_{i=0}^{3}(y_i - y_{n_i})(z_i + z_{n_i}) = \overset{B-E}{(0-3)(3+3)} + \overset{C-F}{(0-3)(3+0)} + \overset{F-C}{(3-0)(0+0)} + \overset{E-B}{(3-0)(3+3)} =$$
$$(-3)(6) + (-3)(3) + (3)(0) + (3)(6) =$$
$$-18 + -9 + 0 + 18 = -9$$

$$N1y = \sum_{0}^{3}(z_i - z_{n_i})(x_i + x_{n_i}) = \cancel{(3-3)(\ldots)} + \cancel{0} + \cancel{0} + \cancel{0} = 0$$

$$N1z = \sum_{0}^{3}(x_i - x_{n_i})(y_i + y_{n_i}) = \cancel{2-2} + \cancel{4-4} + \cancel{4-4} + \cancel{2-2} = 0$$

this is wrong, I don't know why, I followed the instructions exactly

Face list

Face	vertices	normal
0	(C,A,D,F)	(0,0,0,0)
1	(C,F,E,B)	(1,1,1,1)
2	(B,E,D,A)	(2,2,2,2)
3	(B,A,C)	(3,3,3)
4	(F,D,E)	(4,4,4)

5.

b)

```
var lightPosition = vec4(4, 2, 4, 0.0);
var lightAmbient = vec4(0.2, 0.2, 0.2, 1);
var lightDiffuse = vec4(1.0, 1.0, 1.0, 1.0);
var lightSpecular = vec4(1.0, 1.0, 1.0, 1.0);

var materialAmbient = vec4(0, 0, 1, 1.0);
var materialDiffuse = vec4(0, 0, 1, 1.0);
var materialSpecular = vec4(1.0, 1.0, 1.0, 1.0);
var materialShininess = 50.0;

var ambientProduct = mult(lightAmbient, materialAmbient);
var diffuseProduct = mult(lightDiffuse, materialDiffuse);
var specularProduct = mult(lightSpecular, materialSpecular);

gl.uniform4fv(gl.getUniformLocation(program, "ambientProduct"), flatten(ambientProduct));
                                              "diffuseProduct"            (diffuseProduct)
                                              "specularProduct"           (specularProduct)
                                              "lightPosition"             (lightPosition)
```

repeat this line changing the variables

c)

```
[in window.onload] function
  texture = gl.createTexture();
  texture.image = new Image();
  texture.image.onload = function() { loadTexture(texture); }
  texture.image.src = 'scene.jpg';

function loadTexture(texture)
{
    gl.pixelStorei(gl.UNPACK_FLIP_Y_WEBGL, true);
    gl.activeTexture(gl.TEXTURE0);
    gl.bindTexture(gl.TEXTURE_2D, texture);
    gl.texImage2D(gl.TEXTURE_2D, 0, gl.RGB, gl.RGB, gl.UNSIGNED_BYTE, texture.image);
    gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_T, gl.CLAMP_TO_EDGE);
    gl.texParameteri(gl.TEXTURE_2D, gl.TEXTURE_WRAP_S, gl.CLAMP_TO_EDGE);
    gl.uniform1i(gl.getUniformLocation(program, "texture"), 0);
}
```

also, add `texCoordsArray.push(texCoord[0]);` to `quad(...)`

1

2

; etc