Mid Term Examination

Due Mar 7 at 6:30p.m. **Points** 100 **Questions** 15
Available Mar 7 at 5p.m. - Mar 7 at 6:30p.m. 1 hour and 30 minutes **Time Limit** None

Instructions

CSCI 3090 Mid Term Examination

March 7, 2023 17:00-18:30

This is an open book examination. You are free to use any source of information, but all answers must be in your own words. You are not allowed to copy and paste from other sources. This will result in a zero(0) on the question. You are required to answer all of the questions. Questions will be presented one at a time. There are 15 questions in total.

This quiz was locked Mar 7 at 6:30p.m..

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	86 minutes	95 out of 100

Correct answers are hidden.

Score for this quiz: **95** out of 100
Submitted Mar 7 at 6:26p.m.
This attempt took 86 minutes.

Question 1

6 / 6 pts

Can a 3x3 matrix be used to perform a 3D translation?

Your Answer:

A 3x3 matrix cannot be used to perform 3D translations. You would need to use a 4x4 transformation matrix to perform a 3D translation

Question 2

6 / 6 pts

Can back face culling be used as a hidden surface algorithm?

Your Answer:

Back face culling cannot be used as a hidden surface algorithm since it would not solve the problem of having one object in front of another, drawing this, we would end up drawing the polygon in the back, then the polygon in front would be drawn since back face culling draws from the farthest back z value to the nearest. However if there is just one object, or there are no overlaps the scene will be rendered as normal.

Question 3

6 / 6 pts

Given the matrix M for transforming vertices, how can the matrix for transforming normal vectors be constructed?

Your Answer:

The matrix for transforming normal vectors can be constructed by taking the inverse transpose of the transformation matrix M.

so we get, $M' = (M^{-1})^T$ where M' is the transformation matrix for the normal vectors.

Question 4

6 / 6 pts

What is the main difference between C^n and G^n continuity?

Your Answer:

For a curve to be C^n continuous the n^{th} derivative of the curve must also be continuous, which means its shape and speed. meanwhile for a curve to be G^n continuous, they have to have proportional n^{th} derivates, meaning same direction but the speed can be different. If a curve us C^n continuous it is also G^n continuous in most cases.

Question 5

10 / 10 pts

What data structure is used to store a polygonal mesh?

Your Answer:

To store a polygonal mesh we use a Vertex Table and a Face table. The vertex table will hold information such as the vertex coordinates, normals, colors, etc. And the face table will just contain the indices of the polygons vertices for each row in the table.

Question 6

3 / 6 pts

How can the normal vector for a polygon be computed?

Your Answer:

The normal vector for a polygon can be computed computed constructing two vectors vec1, vec2, using its vertices then taking the cross product of the two vectors

So for example if our polygon is a triangle we would have 3 vertices, A, B, and C.
we would then construct the two vectors vec1 and vec2

finally we get the normal vector, n = vec1 dot vec2 (dot product between vec1 and vec2)

Question 7

10 / 10 pts

Briefly explain the z-buffer hidden surface algorithm. What problem can occur when this algorithm is used with the perspective projection?

Your Answer:

The z-buffer algorithm works by creating a buffer which has a size equal to the resolution of the viewing window and all points to the farthest point, f (the far plane)
next to write a pixel we must first check the current z-buffer value and compare it against the the z' value. if the z' is less than the value in the z-buffer, we write the pixel and update the buffer otherwise we ignore.

The problem with using the z-buffer hidden surface algorithm with the perspective projection is that the size of the bins increase as we get farther and father away from the eye. This would result in z-fighting

Question 8

6 / 6 pts

Briefly describe the three components of the Phong lighting model.

Your Answer:

The Phong illumination model consists of 3 components. Ambient light, Diffuse Light, and Specular Light.

Ambient light is the illumination of the scene from the background light of the environment and is modeled by a constant C_a which is a vector of rgb values representing the light level. Can think of this as the base scene lighting

The amount of ambient reflection of an object is modeled by the relationship $I_a = C_a C_r$. (This is component wise multiplication between the reflection vector of the object and C_a.

Diffuse light is the light from diffuse reflections, it is uniform in all directions and the amount of diffuse light is proportional to the amount of light that is hitting the object. We can approximate diffuse lighting using lamberts cosine law giving us the following relationship. $I_d = C_l C_r \cos(\theta) = C_l C_r (L \cdot N)$. we can also normalize the L and N vectors so we get max intensity at $\theta = 0°$ and 0 intensity at $\theta = 90°$. in the equation, C_l is the rgb color of the diffuse light. If we get a negative dot product that means that the normal is pointing away from the light source, to get around this we can make use of the max function to make sure that the diffuse light is 0 in that case. Our equation for diffuse light then becomes $I_d = C_l C_r \max(0, (L \cdot N))$

the final component would be Specular light, this is the light from mirror-like reflections. just using pure reflection would produce really small highlights, we can fix this by enhancing the peak of the function by raising it to a power and use max like we did above to avoid negative values. for our specular equation we get:

$I_s = C_l(\max(V \cdot R))^n$

Question 9

6 / 6 pts

A graphics programmer wants to change the size of an object in his program. They apply a scale matrix to the object and notices that its size changes, but it also changes position. They don't want the position change. What should the programmer do?

Your Answer:

To fix this issue the programmer would need to first translate the object to the origin, perform the scale, then translate the object back to the desired position. Directly scaling would work if the object was already at the origin, but since the object is being translated it means it probably is not, therefore you would need to first translate the object to the origin, scale the object, then translate back.

Question 10

6 / 6 pts

What is the main reason for using uniform blocks?

Your Answer:

The main reason we would want to use a uniform block is if we want to to send multiple uniform variables to the shader, instead of sending each uniform value individually we can send the entire block at once. This has the effect of saving calls to OpenGL and makes it easier to modify the uniforms.

Question 11

6 / 6 pts

How are texture coordinates sent to the GPU?

Your Answer:

We send texture coordinates to the GPU similarly to how we send vertex information. We package the texture coordinates with the vertex buffer. this can be done using the following steps:

1. Call glGenBuffers to generate a vertex buffer

2. Bind the vertex buffer we just generated using glBindBuffer()

3. define the size of the buffer and how it will be drawn using glBufferData() note, the size parameter for this function should be the size

of the vertices + the size of the texture coordinates since we are sending both those pieces of information.

4. Call glBufferSubData() on our buffer to populate it with the vertices

5. Call glBufferSubData() on our buffer to populate it with the texture coordinates. note, the starting position for this function call must be the sizeof(vertices) since we want to place the texture coordinates right after the vertices.

Finally we can access and set the normals in our shader program as we would usually do.

When displaying this texture we would just bind it using glBindTexture(), then we can issue the draw call as normal.

Question 12

6 / 8 pts

What is a mipmap and what problem does it solve?

Your Answer:

a mipmap is when you take a texture from its original size and dividing it by 4, you do this until the resulting size is 1x1. We use mipmap's to prevent texture aliasing by sampling the texture at a far away distance from the camera. this will give a slightly blurred image but it is good enough.

Question 13

6 / 6 pts

In a rush to complete an assignment a student produced the following vertex shader. The student noticed that the lighting was all wrong. What are the problems with this shader?

```
in vec4 vPosition;
in vec3 vNormal;
```

```
uniform mat4 modelViewProjection;
out vec3 normal;

void main() {
    gl_Position = modelViewProjection * vPosition;
    normal = modelViewProjection * vNormal;
}
```

Your Answer:

the problem with this code is that the normal vector is being calculated incorrectly. it is calculating the normal by using the modelViewProjection matrix which is incorrect, instead it needs to use a modelView matrix. to do this we first need to define a uniform for the normal matrix (normalMat) that will be passed to the shader by OpenGL the normalMat will be calculated using the inverse transpose of the view matrix. we can then calculate the normal in our shader by doing normal = normalMat * vNormal.

Question 14

6 / 6 pts

What is the main difference between an attribute and a uniform variable?

Your Answer:

A uniform variable stays constant and holds the same value across vertices and fragments however an attribute variable gets a new value for each vertex.

Question 15

6 / 6 pts

In the case of a curve or surface why is local control important?

Your Answer:

Local control is important because we do not want a small change on the control points to have a large impact on the curve, this lets us work on the curve one part at a time.

Quiz Score: **95** out of 100