IN3005: Computer Graphics using OpenGL

Instructor: Dr. Eddie Edwards
Email: Philip.Edwards@city.ac.uk

Interim Coursework

Due 5 pm Sunday 28th February 2021

Submission Details:

You must upload the finished document to Moodle by **5 pm Sunday 28th February 2021**. Standard lateness penalties and extensions policy applies. This assignment is scored out of 100 and scaled to **10% of the total marks for this module**.

You should submit a document (doc/pdf) that describes that provides written answers to the following questions. The five parts carry equal marks. You are only expected to provide written answers. No functioning code needs to be written or submitted.

This submission will be via TurnitIn and will be fully checked for plagiarism. Please complete the task individually as evidence of copying will result in a zero mark and may result in disciplinary action.

Context:

As part of a game that you are making, you intend to render a house. The house will have four walls and an angled roof. The remaining questions will examine how you would go about designing and rendering this house using OpenGL.

Question 1 – Geometry and vertex attributes:

The house should be positioned so that the Y-axis is up, the X-axis is left-to-right looking at the front of the house and the Z-axis is pointing outwards from the front wall. Is this a right-handed coordinate system? Explain how you arrive at your answer.

Choosing your own dimensions for the house and angle for the roof, draw a diagram showing how many vertices are required in the model and list the coordinates for all of the vertices. Choose a suitable origin and explain your choice.

Accurate rendering of the object will require correct normals for the walls and the roof. Calculate these normals for your design, describing your working and describe any changes that need to be made to the previous list of vertices to achieve this (hint: you may want to duplicate some vertices to allow each face to have consistent normals).

The house is to be rendered without any windows or doors, but with a different colour for the walls and the roof. Provide a possible OBJ file format description of your house. This does not have to be a working OBJ file that can be loaded, but needs to contain the positions, normals and faces. The vertex positions and normals should be listed first, then the faces. Assuming you have two materials defined in a separate file, matWall and matRoof, you can colour the faces (not the vertices) as follows:

```
usemtl matWall

f 1//1 2//2 3//1

      ⋮

usemtl matRoof

f 12//12 14//12 13//12

      ⋮
```

Where each face provides the index of the vertex position, v, and the vertex normal, vn, for each corner of the triangle in the format v//vn. It is fine to have a vertex normal defined for each vertex position in which case the format will be f n//n m//m p//p, but it is also fine to list fewer normals if you prefer.

When describing the faces you should order the vertices in each triangle so that the face points outwards.

Question 2 – Vertex Buffer Object (VBO) design and transformations

You now have to design a VBO to contain the vertex attributes for your house. Choose a suitable layout for your VBO and list the calls to OpenGL from your client program required to transfer the VBO to the GPU. These should be native OpenGL commands. Do not assume that you have the helper classes from the template. Ensure that you give the correct values for the position and stride for each of the vertex attributes in VBO.

Your house is to be translated so it is centred at a new position on the ground plane, rotated by an angle that is not a multiple of 90º in such a way that the house remains correctly positioned on the ground and scaled to some multiple of its size. Choose your own rotation, translation and scale and provide the calls to glm using a glUtil matrix stack that will achieve the desired transformation. Also provide the code to transfer this information to the GPU.

Question 3 - Camera positioning

You now want to place a camera so that it is looking at the house from a reasonable distance and such that the house fills most of the screen. Begin by explaining the how terms *fovy, zNear, zFar* and *aspect* affect the viewing frustum when calling the function glm::perspective(). Then list the function calls in the client program required to achieve the desired view, including moving the camera to the correct location and orientation.

You now want the camera to move in a path such that it moves in a circle around the house, with the house always being at the centre of the screen. Describe how you would achieve this and what convenience functions you might use for this purpose. You don't need to provide full code, just a description of the method.

Question 4 – Shader implementation

What are the minimum shaders that are required to achieve rendering of the house? Provide the necessary GLSL code listings for these shaders to render the house at the correct location in the scene. Include the transfer of any relevant information from the client program via uniform variables and the VBO.

Question 5 - Texture mapping

Instead of a colour for the walls and the roof, you now want to include a brick texture for the walls and a tile texture for the roof. Explain how this can be achieved with the use of a diagram and provide appropriate texture coordinates for the vertices calculated in question 1.

Show how the VBO design must be amended to include this information and provide updated client C++ and GLSL code to achieve textured rendering.

The front of the house is to be rendered with a door and four windows. Explain how this could be done with a change of texture alone. Then draw the front of the house and show how many new vertices and triangles would be needed to achieve this with geometry rather than texture.