COMP612 Computer Graphics Programming

Project

This is an individual assignment. All work you submit must be entirely your own.

Where your implementation idea or inspiration has been taken or adapted from other published sources those sources should be acknowledged appropriately in the class header and detailed in your developer's logbook.

You must develop using the Eclipse version and jogamp libraries specified on Blackboard. It is your responsibility to submit an Eclipse project that runs on a COMP612 classroom machine.

It is expected that you will work consistently on this project, from briefing session to due date. Time will be allocated in weeks 10-12 to work on your project. Please be aware that this is not an assignment that can be completed at the last minute.

Proposal Due: Monday 6 May 2019 10:00 am

Project Due Date: Friday 31 May 2019 3:00 pm

Submission:

Proposal

You must submit a proposal to receive a mark for the project. An example proposal is available on Blackboard. Submit your proposal as a PDF file using the Blackboard link provided and bring a printed copy to class on Monday 6th May. Your proposal will be discussed with you individually in this class. Where necessary a revised proposal may be required to be submitted.

Project

An electronic submission must be uploaded to Blackboard using the link provided.

Your submission should be a single zip file containing the following:

- Eclipse project including any additional files required to run your project, if any, such as image files used for texture mapping
- Any additional external libraries (*.jar) used
- Your logbook as a PDF file

Late assignments, without an approved extension, will be subject to a deduction of 5% (one grade e.g. from C+ to C) of the total mark available for each 24-hour period, or part thereof, up to a maximum of five calendar days. Assignments over five days late will not normally be accepted or marked and students will receive a DNC (Did Not Complete) for that assessment.

From the Office of Jacqueline Whalley, Associate Professor
<jwhalley@aut.ac.nz> | <+64 9 9219999 ext. 5203>

Marking:

This assignment is worth 50% of the overall mark for the paper and will be graded out of 100. The project will be graded on correctness, efficiency, programming style, user interaction and elegance of the user interface. In addition, creativity and superior rendering effects will be rewarded with marks.

Plagiarism:

Please be aware that any piece of your assessment might be tested with the plagiarism prevention software to which AUT subscribes.

Assignment objectives:

This assignment assesses your ability to:

- Conceptualise, specify, and develop a significant OpenGL animation application.
- Create a working 3D real time animation.
- Effectively use lighting, materials, texturing, display lists, and fog.
- Implement smooth user interaction using mouse and key controls.

Notes:

- Your assignment will be re-compiled before running, so make sure it works on the AUT lab computers, not just on your home computer.

Introduction:

This project is designed to give you the freedom to explore and create a new 3D CG environment. Your environment may be set anywhere or at any time and your CG environment and code must demonstrate your ability to use some key graphics programming techniques.

Proposal:

Prepare a proposal for your project. It should include full details of your environment's theme and contain illustrative drawings. At this stage you need only detail the base requirements.

Details should include:

- Theme for the environment.
- A storyboard, or similar hand drawn illustrations, of the navigation through the scene.
- The various objects that will be included to complete the scene.
- How your animated spotlight will be used.
- Images of the textures you intend to use (make sure you adhere to copyright)
- Identify the scale of your objects and environment and mapping of real world units to OpenGL units. e.g.: if moving 1.0 units in the application what does this map to in the "real" world? 1 metre, 1 cm? Thinking about this now will make the positioning and animation of your objects (including the camera) in the environment and the projection parameters easier to implement.

From the Office of Jacqueline Whalley, Associate Professor
<jwhalley@aut.ac.nz> | <+64 9 9219999 ext. 5203>

This proposal gives you the opportunity to discuss your ideas and receive feedback and advice about the implementation of your project.

Base requirements for your CG Environment:

The following criteria must be met in your project:

Projection

Your scene must use a perspective projection.

Tracking Camera

You must implement a camera which tracks an object that is moving through your environment (you may modify the TrackballCamera provided for Assignment 2 to meet this criterion). To associate your object with the camera place the object just below and in front of the camera at an appropriate scale and move the camera and object relative to each other. This object could be a simple as a sphere but should make sense in your environment.

Display Lists

Place objects, where appropriate, into display lists in order to speed your animation.

Terrain/Floor

Construct a basic flat floor or terrain by implementing a `Grid` class which draws a grid of GL_QUADS on the X-Z plane centred at the origin.

Use vertex by vertex normal vectors for lighting to work. For each vertex you should set a unit normal using the `glNormal3d(0, 1, 0);` method. Remember that the winding of the quads is important – the points that make up each quad should be drawn in an anticlockwise direction.

Lighting

You should make full use of the OpenGL fixed pipeline lighting functions and include at least 2 different lights including:

- An animated spotlight
- One directional light source, with appropriate ambient, specular and diffuse levels

Set the light position(s) after setting up your camera.

Shading

Where you are not employing texture mapping you should use materials. Set the appropriate shininess, ambient, specular and diffuse material properties for each object.

Texture Mapping

You must use texture mapping to texture map at least two objects (one could be your floor/terrain). For example you might texture map your sky, a hedge or a pond.

- You can use any image textures you like (subject to copyright). There are some links to free texture libraries available on Blackboard.

- Turn texture mapping on *only* for texture mapped objects. If the objects are not texture mapped they should be drawn as shaded objects (using Materials).
- Use texture mipmapping.

Fog

Use fog in some effective way within your environment. Think carefully about your fog colour. Fog is typically used to do fading out/blend the scene into the distance and hide edges. You might want to alter fog colour and density based on the camera position in order to get the best effect for your scene.

A Complete Scene

Complete your environment by adding appropriate objects to the scene. For example a forest, buildings, roads, lakes, etc. The objects you add will be dependent on your choice of scene and should be consistent with the aesthetic of your scene. Enough objects and different types of objects should be added to make the exploration of your environment interesting.

Logbook

You must hand in your logbook as it forms part of your proof of authorship. Remember, the onus is on you to prove you are the creator of your product; thorough record keeping is essential to this process.

Your logbook should record dates, time spent, a record of bugs and fixes, design details, extension ideas and details as to how you realised the extension. It must also contain a brief description, that should be no more than 2 pages (one sided) or 1 page (double sided) excluding images and drawings, that explains how you implemented the project and describes the more challenging features and how you implemented them.

Finally you must provide a reflective statement in the last entry that critically evaluates what you did well, what you might do differently next time and identifies the shortcomings of your application.

One Extension - ADVANCED TECHNIQUES

This section is designed for those of you working towards an A grade.

To attempt this section you should have completed the base requirements. It is recommended that you prioritise completing the base requirements to a high level before working on this part of the assignment.

Because this is the section which distinguishes the projects that achieve an A grade from the others this part is more difficult. A project which has the base requirements implemented well will do better than one that has base requirements and an extension completed less well.

Remember that the advanced technique not only needs to be implemented well but also needs to make sense and be visually appealing in the context of your environment.

If you decide to proceed to this section you should choose and implement successfully in your environment **one** of the following advanced techniques. Don't use a technique you used in Assignment 2.

A mountainous or hilly terrain

Implement a terrain that is not flat. For example, this could simply be a mountainous area of land, or it could be an underwater terrain with raised islands. There are several ways you could do this including:

- Write a fractal terrain generation algorithm.
- Write code to read in a height map file (the height map may be an image or a text file) and create a triangle or quad mesh. This must be your own file parsing code.
- Calculate and render a parametric surface such as a Nurbs surface or a Bezier surface.

Make sure your vertex normal vectors are perpendicular to the surrounding faces.

Simple Collision Detection and Reaction

Implement a simple collision detection system (such as Bounding Sphere or Axially-Aligned Bounding Box) to prevent the animated object navigating your environment from moving through other objects. React to the collision in some way, for example you can make your object crash, or simply stop it from running into things by changing its heading.

An L-System

Use an L-System to generate a self-similar fractal (such as a tree or seaweed).

A 3D Particle System

Expand what you learnt in Assignment 1 and create a 3D particle system, for example bubbles or rain.

Swarm intelligence

Add some form of artificial intelligence to your environment. For example, a school of fish or a flock of birds.

Mirrors or shadows

Include a mirror or shadow(s) in your scene. You only need to use a relatively simple squish matrix or stencil buffer approach (slides on this topic can be found on Blackboard). Alternatively, you could explore the use of Shaders to get these types of effects.

A day/night cycle

This extension involves making effective use of the techniques learnt to implement a gradual transition between day and night. You will need to consider things such as lighting, texturing, and shading (materials) to get a good effect.

Billboarding

A billboard is a textured flat (2D) object, often a quad, which faces the camera. The billboard's direction changes constantly as the object and camera move, so it always faces the camera direction. Billboards use texturing and transparency to represent phenomena that do not have a smooth solid surface, for example: grass, a tree, fire or smoke. It allows for faster rendering because you are not dealing with a volumetric object.

Enclose your environment

Use a dome or skybox to enclose your environment. The textures you use will depend on your scene.

Mesh Model

Include an animated or static 3D mesh model. This may be the object that is tracked by your camera or you may choose to follow another object. You should limit your choice to models in OBJ format, with parts and a low number of vertices/polygons.

Shaders

Write your own Shaders to get better effects such as improved lighting (by pixel rather than by vertex) or to create better shadows, mirrors or animated water or grass. In jogamp's GL2 you can mix GLSL and fixed pipeline functions so you can use for example gluLookAt for the camera and a Phong by pixel shader.

If you have an idea that is different and you want to use it as an advanced extension then you should discuss it with the lecturer for approval before proceeding.

HINTS AND TIPS

- Now you are working with more objects you will need to consider the efficiency of your code. If your animation slows, consider using display lists when appropriate to speed up the rendering of your animation frames. Consider billboarding where appropriate to speed rendering.
- Keep your environment relatively small.
- Test ideas in small test projects. Move them into your actual project once you are happy with the result.
- Don't forget to calculate your normal vectors for the ground and other objects in order to get the correct lighting effects.
- If your textures look dark, make sure that glColor( ) is set to white, because the texture colours are multiplied by the current drawing colour.
- Keep a regular eye on Blackboard as useful information will appear when people ask questions or want to know how to attempt something new. Ask questions and answer other people's questions. You never know what you might learn!
- For a successful project attend class where we will cover topics relevant to your project and work on your project when you have the opportunity in class.
- Ask for help when you get stuck, earlier rather than later.