Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code: CM3114
Module Title: Graphics
Lecturer: Dr Xianfang Sun
Assessment Title: Graphics Coursework
Assessment Number: 1
Date Set: 1 November 2021
Submission Date and Time: 13 December 2021 at 9:30am
Return Date: 17 January 2022

This assignment is worth 100% of the total marks available for this module. If coursework is submitted late (and where there are no extenuating circumstances):

1 If the assessment is submitted no later than 24 hours after the deadline, the mark for the assessment will be capped at the minimum pass mark;
2 If the assessment is submitted more than 24 hours after the deadline, a mark of 0 will be given for the assessment.

Your submission must include the official Coursework Submission Cover sheet, which can be found here:
https://docs.cs.cf.ac.uk/downloads/coursework/Coversheet.pdf

Submission Instructions

Upload the files listed in the following table onto Learning Central in the CM3114 module Assessment Coursework section by 13 December 2021 at 9:30am.

Description		Type	Name
Cover sheet	Compulsory	One PDF (.pdf) file	[student number].pdf
Source code	Compulsory	One zip file containing the Java files and other files necessary for the program to run correctly	Code_[student number].zip
Report	Compulsory	One report (.pdf) file explaining your implementation	Report_[student number].pdf

Any code submitted will be run on a system equivalent to those available in the Linux laboratory and must be submitted as stipulated in the instructions above.

Any deviation from the submission instructions above (including the number and types of files submitted) may result in a deduction in marks for the assessment.

Staff reserve the right to invite students to a meeting to discuss coursework submissions

Assignment

Task 1. Basic renderer

This task consists of three individual JOGL programming subtasks. Partially functional programs are provided. Download the file CGCW.7z from the Assessment section in Learning Central. You are required to modify the provided programs and submit your new versions. You should be able to do the first task after week 5, the second after week 6, and the third after week 7's lectures/labs. *You are strongly advised to start the coursework from the first week you have received the assignment, rather than wait until the last minute.*

1. ***Interactive Transformations*** [20]
   The purpose of this task is to implement interactive model--view transformations with keyboard input.

   You need to modify the source code CGCW01.java. The initial program has implemented the rendering of an object (teapot) and allows interactive enlarging of the object. Comments in the program show where you should add code for tasks (b) and (c), but you should find the location yourself for task (a). You are required to

   a) find an appropriate location in the program and add suitable definitions for additional parameters that are required for the transformations below;
   b) fill in code in the function `keyPressed()` to calculate the parameters;
   c) fill in code in the function `display()` to perform the transformations.

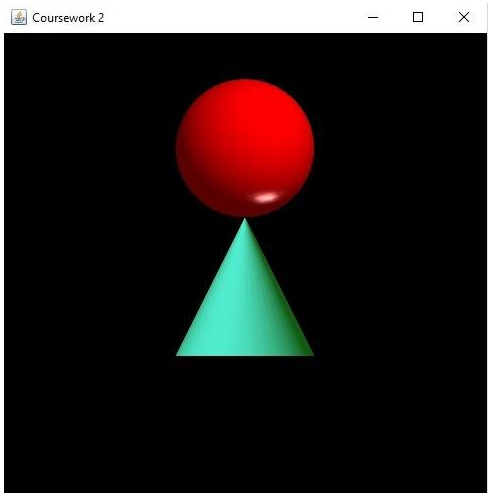
   The transformations to be implemented are:
   - on pressing the key `M', the object should expand; (this function has already been implemented)
   - on pressing the key `N', the object should shrink;
   - on pressing the Left/Right arrow key, the object should move left/right;
   - on pressing the Up/Down arrow key, the object should move up/down;
   - on pressing the key `X'/ `C', the object should rotate around the ***x*** axis clockwise/anti-clockwise;
   - on pressing the key `Y'/ `U', the object should rotate around the ***y*** axis clockwise/anti-clockwise.

2. ***Modelling and Lighting*** [20]
   The purpose of this task is to build a 3D polygonal model, and render a scene including more than one object with different material properties.

   You need to modify the source code CGCW02.java and create a new file SCone.java.
   The initial CGCW02.java program has implemented rendering one object (a sphere). You are required to

   a) create a new class `SCone` extended from `SObject`, which builds a cone model. It is expected that the class `SCone` works like the class `SSphere` provided for rendering the sphere, except that `SCone` is used for rendering a cone;

b) in the function `init()` in CGCW02.java, add code to create a cone, and set its material to one which is different to the sphere's;
c) in the function `display()` in CGCW02.java, add code to transform the sphere and cone to appropriate positions so that the rendered image looks like that shown in the following picture.

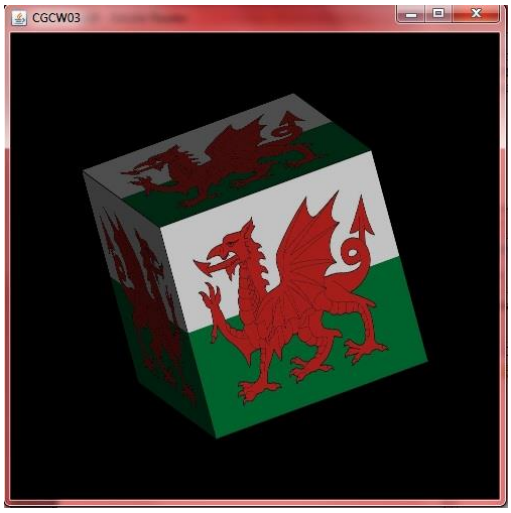


3. ***Texture mapping*** [25]

The purpose of this task is to perform texture mapping onto a 3D cube.

Write a program CGCW03.java to implement texture mapping on all six sides of a cube object. You need to create an `SCube` class extended from `SObject`, which defines the vertices, normals, and texture coordinates of a cube. You also need to write your own vertex and fragment shaders (called Texture.vert and Texture.frag), which should implement blending of texture and Gouraud shading. You can use Gouraud.vert and Gouraud.frag as initial shaders, and add texture mapping in the shaders to form Texture.vert and Texture.frag.

A texture file WelshDragon.jpg is provided. The rendered image should look like the following picture, but it is not required that your rendering result should have exactly the same viewpoint as this.



Task 2. Advanced features

Implement one of the following advanced features. Clearly indicate in the accompanying report which one of these you have chosen; you will only get marks for one of them. To implement feature 1 or 2, you need to load a 3D Teddy mesh model into the scene. The model, named as Teddy.ply, is attached with this coursework. It is in ASCII PLY file format (see http://graphics.stanford.edu/data/3Dscanrep/, http://www.cc.gatech.edu/projects/large_models/ply.html, or https://en.wikipedia.org/wiki/PLY_(file_format) for details about PLY). [25]

1. Smooth the Teddy mesh by applying a suitable subdivision scheme multiple times to it before you render the mesh.
2. Compute and render the shadow of the Teddy mesh on a single planar surface cast from a single light source. Recall that shadows are projections, similar to perspective projections.
3. Create an object which consists of at least four transparent polygons moving through the scene, e.g. a rotating glass box. Note that when you render more than one transparent polygon you have to make sure that there are no visible artefacts from every camera position.
4. Generate a close-to realistic scene using ray tracing or radiosity method, or high-level texture mapping (bump mapping, displacement mapping etc.)

Task 3. Documentation

1. The source code of Tasks 1&2 should include concise comments, explaining the meaning of the code. [5]
2. The report should describe which and how the tasks/features you have completed. [5]

Learning Outcomes Assessed

- Using a standard graphics API to write graphics software for creating images of 2D and 3D scenes and user interaction.
- Devise modifications to existing graphics algorithms for special cases, and design new ones
- Implement a basic renderer and adjust them to special image quality requirements
- Create appropriate models of 3D objects and scenes

Criteria for assessment

Credit will be awarded against the following criteria.

	Excellent (≥70)	Good (60~69)	Adequate (40~59)	Poor (<40)
Functionality: (Tasks 1&2) to what extent does the program realise the task described? **(60%)**	efficient and complete implementation; all cases are considered	Feasible implementation, but not optimal; not all cases are considered	progress towards a full implementation, but not fully working with major deficiencies	little or no progress towards implementation; approach not suitable
Design and Structure: (Tasks 1&2) how clear is the structure of the code and how well are data structures and algorithms used? **(40%)**	well structured code with highly suitable data structures and optimal, clear algorithms	good structure with suitable data structures and algorithms; sometimes not optimal	attempt at using appropriate data structures and algorithms visible; sometimes structure is confusing	code is mostly obscure; little or no structure is visible in use of data structures and algorithm design
Code Documentation: (Task 3) **t**o what extent do the comments and the report help a reader to understand the code? **(100%)**	clear and concise report and comments describing ideas and high-level structure without unnecessary detail	clear report and comments about high-level structure and ideas; sometimes incomplete or too focused on details	some comments about the structure and ideas present, but hard to follow and too focused on details; the report describes the work done, but misses some details	hardly any comments or only very confusing or low-level comments about single instructions; no report, or the report is confusing

---

Feedback and suggestion for future learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned on 17 January 2022 via Learning Central. If you have any questions relating to your individual solutions talk to the tutor or the lecturer.