Cardiff School of Computer Science and Informatics

Coursework Assessment Pro-forma

Module Code: CMT107
Module Title: Visual Computing
Lecturer: Dr Xianfang Sun
Assessment Title: Visual Computing Coursework
Assessment Number: 1
Date Set: 28 October 2019
Submission Date and Time: 6 December 2019 at 9:30am
Return Date: 18 January 2020

This assignment is worth 30% 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

Complete the programming tasks as described in the Assignment section, and upload the files listed in the following table onto Learning Central in the CMT107 module Assessment Coursework section by 6 December 2019 at 9:30am.

Description		Type	Name
Cover sheet	Compulsory	One PDF (.pdf) file	[student number].pdf
Source code	Compulsory	A zip file contains one or more Java files and other files necessary for the programs to run correctly	Code_[student number].zip
Readme	Compulsory	One plain text file indicating which features you have implemented	Readme_[student number].txt

Any code submitted will be run on the School provided Linux laptop and must be submitted as stipulated in the instructions above.

You may copy small code fragments from the CMT107 handouts and lab classes, or any recommended and background textbooks for CMT107, including from the printed text,

CDs/DVDs, websites, etc. for the textbook, provided that your source code contains clear reference to the origin of the code. You may not reproduce code written by any other student or code obtained from any other source not mentioned above. If you are in doubt about whether you may include a code fragment that you have not written yourself, ask the lecturer. Do not share your code with the other students, or publish it on any publicly accessible websites, such as Github, Stackoverflow, etc. You will be under unfair practice investigation if your code is found almost the same to another student's, and the possible consequence for both students is a mark of zero for the assessment or worse.

Use the laboratory classes to get guidance in doing this exercise. You are encouraged to contact me by e-mail if you have any queries; it may well be possible for me to answer them almost immediately. If necessary you are welcome to arrange to come and see me individually about the coursework. But the meeting is only for clarifications about the question, not for help in solving the problem.

Any deviation from the submission instructions above (including the number and types of files submitted) will result in a mark of zero for the assessment.

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

---

Assignment

This coursework consists of two parts related to graphics and Image processing, respectively. For you to complete the coursework in time, you are suggested to complete the graphics part before the end of week 7, and then finish the image processing part. The total marks assigned to each part and divided into each task are listed in the brackets.

Graphics **[45 marks in total]**

Write a program using JOGL to render a 3D scene, which can be navigated in a walk-through fashion. You are free to choose the scene, but the scene and your program should exhibit the features described below.

1. Rendering Framework [15]
   Implement a framework to render a small shaded 3D scene with JOGL. It should provide the following functionality:
   (a) A rendering system which sets up the scene. It is important that the rendering system is clearly structured to easily adjust the types and positions of light sources, and add and remove objects, etc. from the scene (see below for the object types).
   (b) A navigation system which allows the user to display the scene from an arbitrary camera position with keyboard AND mouse control to facilitate moving about the scene by adjusting the view direction and moving forward/backward (or a similar 3D navigation system). This should change the view in a natural and predictable manner.
2. 3D Objects [15]
   Use the rendering framework to create a scene which contains at least three objects shaded with different material properties. Your scene should contain at least one

object that is not provided in the labs, i.e., an object that is not a plane, sphere, cube, or teapot, and it must be created by yourself rather than from a third party.

3. Texture mapping [15]
   Implement the texture mapping on one of the objects in the scene.

Image Processing **[35 marks in total]**

Write a Java program to implement the Harris corner detector, each step of which should be clearly shown in the code. It is not allowed to use functions from the computer vision libraries, such as OpenCV, for the algorithm implementation. You can use the attached image Room.jpg as a testing example, but you are free to choose any other images for testing. The program should be able to:

1. Load and display an image to be processed by the algorithm [10]
2. Implement Harris corner detection algorithm [20]
3. Display the corner detection result [5]

Documentation **[20 marks in total]**

The source code should include concise comments, describing how the program is achieving the various tasks. A Readme file is required to describe which tasks you have completed.

Learning Outcomes Assessed

- Programming of simple visual computing algorithms, including data handling.
- Understanding of the computational burdens associated with different processing techniques and be able to select appropriate methods depending on the intended application and context.

Criteria for assessment

Credit will be awarded against the following criteria.

	Excellent (≥70)	Good (60~69)	Adequate (50~59)	Poor (<49)
Functionality: to what extent does the program realise the task described? **(50%)**	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: how clear is the structure of the code and how well are data structures and algorithms used? **(30%)**	well structured code with highly suitable data structures and elegant, 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: to what extent do the comments help a reader to understand the code? **(20%)**	clear, concise comments describing ideas and high-level structure without unnecessary detail	clear 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	hardly any comments or only very confusing or low-level comments about single instructions

Feedback and suggestion for future learning

Feedback on your coursework will address the above criteria. Feedback and marks will be returned on 18 January 2020 via Learning Central. If you have any questions relating to your individual solutions talk to the tutor or the lecturer.