UNIVERSITY OF Hull

DEPARTMENT OF COMPUTER SCIENCE AND TECHNOLOGY

ASSESSMENT COURSEWORK DESCRIPTION 2020/21

MODULE DETAILS:

Module Number:	500079	Trimester:	2
Module Title:	3D Computer Graphics		
Lecturer:	Dr Xinhui Ma & Dr Qingde Li		

COURSEWORK DETAILS:

Assessment Number:	1	of	2		
Title of Assessment:	OpenGL Basic 3D Graphics Effects				
Format:	Program	Demonstration	3rd format		
Method of Working:	Individual				
Workload Guidance:	Typically, you should expect to spend between	50	and	70	hours on this assessment
Length of Submission:	This assessment should be **no** more than: *(over length submissions **will be** penalised as per University policy)*	N/A - coding exercise **words** *(excluding diagrams, appendices, references, code)*			

PUBLICATION:

Date of issue:	15 Febuary 2021

SUBMISSION:

ONE copy of this assessment should be handed in via:	Canvas		If Other (state method)	
Time and date for submission:	**Time**	2pm	**Date**	See Below
If **multiple hand–ins** please provide details:	Source code and demo Video – 26 March 2021			
Will submission be scanned via TurnitinUK?	No	If submission is via TurnitinUK, these should be one of the allowed types e.g. Word, RT, PDF, PPT, XLS etc. Specify any particular requirements in the subumission details Students MUST NOT submit ZIP or other archive formats. Students are reminded they can **ONLY** submit **ONE** file and must ensure they upload the correct file.		

The assessment must be submitted **no later** than the time and date shown above, unless an extension has been authorised on a *Request for an Extension for an Assessment* form: search 'student forms' on https://share.hull.ac.uk.
Canvas allows multiple submissions: only the **last** assessment submitted will be marked and if

submitted after the coursework deadline late penalties will be applied.

MARKING:

Marking will be by:	Student Name

ASSESSMENT:

The assessment is marked out of:	100	and is worth	50	% of the module marks
N.B If multiple hand-ins please indicate the marks and % apportioned to each stage above (i.e. Stage 1 – 50, Stage 2 – 50). It is these marks that will be presented to the exam board.				

ASSESSMENT STRATEGY AND LEARNING OUTCOMES:

The overall assessment strategy is designed to evaluate the student's achievement of the module learning outcomes, and is subdivided as follows:

LO	Learning Outcome	Method of Assessment *{e.g. report, demo}*
1	*Demonstrate a practical understanding of graphics hardware.*	Program, Demo
2	*Implement an efficient real-time graphics application.*	Program
3	*Provide evidence of knowledge and understanding of 3D graphics and rendering techniques.*	Program, Demo
4	*Apply appropriate mathematical techniques to solve graphical problems.*	Program, Demo

Assessment Criteria	Contributes to Learning Outcome	Mark
Quality of basic OpenGL effects	1, 2, 3, 4	65%
Advanced Technical Features	1, 2, 3, 4	25%
Quality of Code	2, 3, 4	10%

FEEDBACK

Feedback will be given via:	Verbal (via demonstration)	Feedback will be given via:	Mark Sheet
Exemption (staff to explain why)			
Feedback will be provided no later than 4 'teaching weeks' after the submission date.			

This assessment is set in the context of the learning outcomes for the module and does not by itself constitute a definitive specification of the assessment. If you are in any doubt as to the relationship between what you have been asked to do and the module content you should take this matter up with the member of staff who set the assessment as soon as possible.

You are advised to read the **NOTES** regarding late penalties, over-length assignments, unfair means and quality assurance in your student handbook, which is available on Canvas.

In particular, please be aware that:

- Up to and including 24 hours after the deadline, a penalty of 10%
- More than 24 hours and up to and including 7 days after the deadline; either a penalty of 10% or the mark awarded is reduced to the pass mark, **whichever results in the lower mark**
- More than 7 days after the deadline, a mark of zero is awarded.

- The overlength penalty applies to your written report (which includes bullet points, and lists of text. It does not include contents page, graphs, data tables and appendices). 10-20% over the word count incurs a penalty of 10%. Your mark will be awarded zero if you exceed the word count by more than 20%.

Please be reminded that you are responsible for reading the University Code of Practice on Academic Misconduct through the Assessment section of the Quality Handbook (via the SharePoint site). This govern all forms of illegitimate academic conduct which may be described as cheating, including plagiarism. The term ‘academic misconduct’ is used in the regulations to indicate that a very wide range of behaviour is punishable.

In case of any subsequent dispute, query, or appeal regarding your coursework, you are reminded that it is your responsibility to produce the assignment in question.

500079 Assessment 20/21 – ACW1

Assessment introduction

ACW1 is worth 50% of a 20 credit module. Code can be written in any language, but it is **highly recommended** that you use C#. OpenGL must be used and your solution should be compiled and linked using Visual Studio 2017. If you are using C# then OpenTK should be used to provide a window, keyboard and mouse input as well as maths functions. **You should test your code**. **The graphics cards used in different PCs may be different.**

Hand in

You should develop your coursework using the framework supplied as part of the labs. The work should be submitted to **Canvas** with **one ZIP** file includes:

1. **Entire VS 2017 or 2019 project source code**, including all dependences and resource files. Please 'clean' your VS solution prior to submission so it is not full of intermediate build junk.
2. **Video** (5 minitues) demonstrating all the required features of your program.

Specification

This project is to create and render a scene of at least **two** types of 3D primitives and **one** existing 3D model, as illustrated in Fig.1. 3D primitives include cube, cylinder, tube, sphere, torus and cone,etc. Existing 3D models can be Utah teapot, Armadillo monster, Stanford bunny or other 3D model.

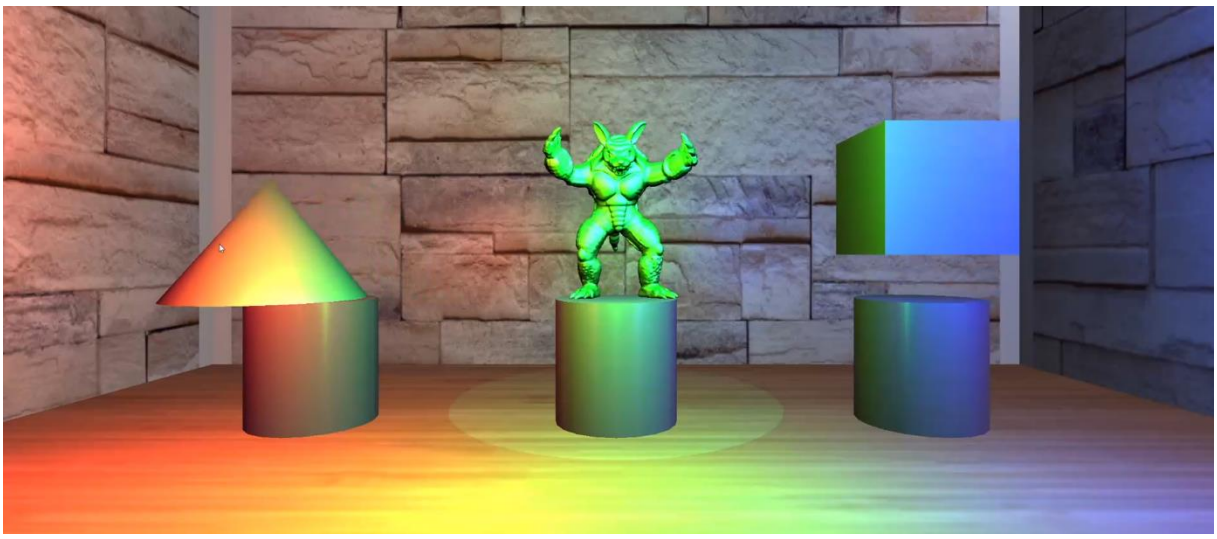

Fig.1. An example scene of rendering on 3D primitives, room texture and the Armadillo monster model

Marking criteria

The marking criteria are defined in the table below. Please note that all aspects are assessed by software and demonstration. You are to produce three dimensional objects rendered using OpenGL.

This means that in order to achieve the full marks in any section you must:

- Submit an implementation of the criteria.
- Achieve the level indicated in the criteria for excellence section.
- Demonstrate understanding of the work you have submitted.

Uses and demonstrates understanding of..	Marks	Assessed by
Draw Calls	5	Software and Demo video
Vertex Buffer Objects	5	Software and Demo video
Vertex Array Objects	5	Software and Demo video
Shaders	5	Software and Demo video
Camera	15	Software and Demo video
Lighting	15	Software and Demo video
Textures	15	Software and Demo video
Special feature: object animation	15	Software and Demo video
Special feature: frame buffer	10	Software and Demo video
Quality of Code	10	Software and Demo video
Total	100	

Criteria for excellence

This section expands upon the marking criteria, and gives guidance towards achieving full marks for each criterion.

1. Vertex buffer objects and draw calls

An excellent student will use vertex buffer objects to load data onto the graphics card, and use draw calls to render objects in 3D space. They will use both data buffers and element buffers and will demonstrate an understanding of different types of drawing primitives. All data loaded onto the graphics card should be deleted before the program ends.

2. Vertex Array Objects

An excellent student will put vertex array objects to good use in order to minimise the impact of state changes between draw calls. All data loaded onto the graphics card should be deleted before the program ends.

3. Shaders

An excellent student will have written shaders to render their objects. They will demonstrate an understanding of the syntax used in the shader code, and how the vertex and fragment shader work together. An excellent student will be able to use multiple shader programs in the same frame. All data loaded onto the graphics card should be deleted before the program ends.

4. Camera Model

An excellent student will implement multiple cameras, including a fixed camera, a user controlled camera.

5. Lighting Model

An excellent student will implement a lighting model capable of simultaneously calculating lights from several different types of sources, e.g. directional lights, point lights and spotlights. Both lights and materials should have colour properties. The lighting model should include ambient, diffuse and specular terms. The student should demonstrate awareness of efficiency issues in the design of their lighting model.

6. Textures

An excellent student will be able to texture objects using more than one texture in the same frame. An excellent student will be able to integrate their textures with their lighting model. All data loaded onto the graphics card should be deleted before the program ends.

7. Special Features

An excellent student will have done extra work and study to implement advanced features such as using frame buffer objects or animation by transformation (e.g. 3D primitives rotating around the teapot) to create special effects.

8. Quality of Code

An excellent student will produce an object oriented solution, creating appropriate classes that are representative of the problem as well as the underlying hardware. For example, they may create classes to ensure that data is not loaded onto the graphics card unnecessarily.