Bournemouth University

Faculty of Science and Technology - Assignment

Course:	BSc (Hons) in Games Technology and Games Programming
Level:	I
Unit:	Programming for Graphics and Games
Assignment:	2 of 2
Lecturer:	Dr Leigh McLoughlin
Quality Assurer:	Dr Andrew Watson
Weighting:	40%
Issue Date:	26/01/2015
Due Date:	24/04/2015

Signature Unit Leader ………………………………. **Signature Q.A. ……………………………….**

ASSESSMENT TASK:
This is an individual assignment. There are two options, you must choose one option to complete.

To achieve an average grade, this task should take you approximately 20 hours.

Option A – 3D Game

Demonstrate your understanding of 3D graphics and programming principles by developing a simple 3D game of your choice.

You must use C++ and OpenGL for the graphics. You may use a fixed 2D perspective and gameplay (such as for a scrolling shooter) but you must use at least one 3D model.

You may use external art assets if you wish (i.e. ones you didn't make yourself), but make sure to reference them.

Your game must display some *appropriate* in-game information to the user on-screen – such as player health, current score, game progress etc.

The game must include at least one additional type of non-player controlled object.

The game can be anything you want or any genre, but keep in mind that you ***cannot*** submit a project you have previously submitted for this or other units.

Example game genres include (but are not limited to):

- Scrolling shooters
- Platform games with 3D graphics
- Racing games
- Strategy games
- First person shooters

Advanced aspects to consider:

- Multiple lights, shader effects, shadows, image post-processing
- Particle effects
- Multiple AI-controlled enemies, 'boss' enemies at the end of the level
- Modular design and scripting interface and e.g. scripted levels

Option B – 3D Technical Demonstrator

Technical demonstrators are small prototype programs that are written to show off a specific advanced effect in an often game-like environment. These are not fully playable games, but can sometimes include interactive components beyond controlling the camera.

For this assignment option, demonstrate your understanding of 3D graphics and programming principles by researching and developing a 3D technical demonstrator to showcase an effect that interests you.

You must use C++ and OpenGL for the graphics and your program should run at near interactive framerates (15-30 frames per second) on the lab machines, though it may of course take additional time for pre-processing and data loading.

You may use third-party libraries to help build up your demo in aspects that you do not wish to be assessed – *i.e. you will only be marked on code that you write yourself.*

Examples include (but are not limited to):

- Advanced lighting or rendering
- Animation blending and skinning
- Physics demonstrator – *you* ***cannot*** *use a third-party physics library if you are demonstrating a physics technique*
  - o 3D pool table simulation
  - o Vehicle simulation showing classical mechanics and collision detection with primitive objects
  - o Fluid simulation (Eulerian or Lagrangian)

DELIVERABLES:

1. Code – a Windows executable (compiled in release-mode) with any required asset files and libraries as well as complete source code with appropriate build files for compiling (such as with Visual Studio)
2. Report – write a report (max 2500 words) which includes:
    a. An overview of your project and an initial specification of what it is supposed to do
    b. Any research you undertook while writing the program (including looking at what work others have done in this area), and details of mathematical solutions you used
    c. A high-level description of your program showing how it works, with program logic flow diagrams and system-level block diagrams as appropriate
    d. A brief project diary / history report, detailing how you progressed through the project and solved problems as you encountered them
    e. Some basic user testing of your program, with feedback comments on usability and design
    f. An analysis of your program, identifying its strengths and weaknesses
    g. Conclusions with proposals for future improvements
3. User Guide – just a simple text file in your project directory to explain how to run and use your program. An in-program user guide would also be extremely helpful.

MARKING SCHEME:

Element	Contribution
Code	**75%**
Report	**25%**

Deliverable	0% → 39%	40% → 49%	50% → 59%	60% → 69%	70% +
Report Introduction, overview and specification 5%	No real project introduction or overview.	Little understanding of project provided in overview.	Some understanding of project in overview and introduction.	Good understanding of project in overview and introduction, with an initial specification.	Excellent understanding of project in overview and introduction, showing contextual understanding and detailed specification.
Research, Mathematical solutions 5%	No real evidence of research or mathematical understanding of solutions.	Little evidence of research or some attempts at mathematical solutions.	Some relevant research with multiple sources and mostly correct mathematical solutions.	Good evidence of research with survey of several relevant sources, with correct mathematical formulations for fully described problems encountered.	Excellent and comprehensive literature review of relevant research, with clear and correct mathematical formulations of complex problems encountered.
Program description, system diagrams 5%	No real description of program, no evidence of how it works or high-level diagrams.	Brief description of program, providing some description of how some of it works, some attempt at high-level diagrams.	Adequate description of program providing clear description of how most of it works, with logic-flow and high-level diagrams as appropriate.	Good description clearly describing program and how it all works, good and appropriate logic-flow and high-level diagrams.	Excellent description clearly describing all aspects of program, how it works, with analysis and justification of design choices that were made, excellent and appropriate logic-flow and high-level diagrams that clearly show the system.
Project progress and problem solving 5%	No project diary or history report, or any attempt at problem solving.	Little evidence of history and problem solving.	Very brief project diary / history with some evidence of problem solving.	Good summary of project progress with clear evidence of problem solving.	Excellent summary of project progress with clearly documented problems and well-

					reasoned solutions.
User testing, analysis and conclusion 5%	No attempt at user testing, analysis or conclusion.	Little evidence of user testing. Brief project analysis or conclusion.	Some evidence of limited user testing. Project analysis identifying some strengths and weaknesses; brief conclusion.	Clear evidence of basic user testing with feedback and comments on playability and design. Clear project analysis showing strengths and weaknesses and identifying their causes; good conclusion identifying potential future work.	Excellent user testing with clear results and analysis of playability and design. Excellent project analysis comprehensively identifying strengths and weaknesses together with their causes and potential solutions; excellent conclusion with clear roadmap for future work.

Deliverable	**0% → 39%**	**40% → 49%**	**50% → 59%**	**60% → 69%**	**70% +**
Code Quality, structure, organisation and readability 15%	No real code structure or attempts at organisation. Poor code with many errors.	Poor code quality with some clear errors, badly formatted, difficult to read, poor attempts at structure.	Acceptable code quality with some errors, mostly readable, some structure and organisation, but inconsistencies in style.	Good code quality, readable and consistent in style, with a clearly defined structure and good attempt at design.	Excellent code quality, clearly designed structure, well organised and written to a consistent standard.
Code comments 10%	No comments in code.	Minor attempt at code comments.	Some meaningful code comments.	Good and clear code comments at appropriate places which enhance code readability.	Excellent code comments, clearly enhancing code readability, written to a standard for documentation (e.g. Doxygen).
Program 50%	Code does not compile / program does not run and does not attempt at answering the task.	Program runs as expected, clearly attempts and mostly achieves the basic task as described, with a wide range of bugs. Interaction may be unintuitive or unresponsive.	Program performs the basic task and evidence of attempts at some minor advanced features or extensions, with some bugs.	Program performs the basic task well with good range of advanced features and extensions, with a few bugs. Interaction is much more intuitive.	Program performs task extremely well with a full set of very advanced features and extensions, with very few bugs.

LEARNING OUTCOMES

Having completed this assignment the student is expected to be able to demonstrate:

Unit ILO 1 – Apply underlying concepts and principles to the design, development and implementation of graphics, animation and games;

Unit ILO 2 – Demonstrate the application of computer graphic techniques for rendering 2D graphics.