**Module: CMP-5010B Graphics 1**

**Assignment: Design and implement a two dimensional interactive track-based game using OpenGL and C++**

**Set by:**  Dr Stephen Laycock, [S.Laycock@uea.ac.uk](mailto:S.Laycock@uea.ac.uk)

**Checked by:** Dr Rudy Lapeer, [R.Lapeer@uea.ac.uk](mailto:R.Lapeer@uea.ac.uk)

**Date set:** 16th February 2021

**Value:** 60%

**Date due:** 15:00 4th May 2021 [week 10]

**Returned by:** 31st May 2021

**Submission:**  Blackboard

**Learning outcomes**

1. To introduce you to the underlying principles of computer graphics. These principles include rendering, transformations in 2D, collision detection between objects and texture mapping.
2. To introduce you to the OpenGL graphics API, used in conjunction with C++.

**Specification**

**Overview**

To develop a 2D game in OpenGL, which is viewed from above, where the player navigates around a track.

**Description**

To design and implement your own 2D game using OpenGL and C++. The user should control a vehicle around the 2D environment that can be viewed from above. There must be a clear start and end to the game and the user must navigate their vehicle around a track or route with boundaries (e.g. a racetrack with grass verges, a river with banks). You may decide on the vehicle and the track but you should incorporate collision detection between your vehicle and the boundaries.

The following features are essential:

* The game has a clear starting position and the user’s controls should be clearly stated.
* A clear objective and approach for determining when the player wins.
* The world should be larger than the screen size which means the background (world) should scroll both horizontally and vertically as the user’s vehicle moves.
* A track or path should be defined with clear boundaries for the user to navigate around.
* There should be a response to colliding with the boundaries (e.g. the vehicle bounces off or slows down).
* Collisions should be detected between the user and the other vehicles in the environment.
* There should be another computer controlled vehicle or object which targets or chases the player to prevent them from winning.

You may create windows using Win32, freeglut, GLFW or other suitable window managers (subject to approval).

**Relationship to formative assessment**

During the lab sessions the teaching staff will provide feedback on your solution to help you progress with this assignment.

**Deliverables**

A full submission should include:

* A zip file containing all the source code, textures and shaders required to compile and run the project.
* A short user manual explaining the controls.

Demonstrations will also take place using MS Teams where you can demonstrate the features of your solution and the markers will ask you questions about it. *Demonstrations will take place in Week 10. (Times to be allocated in Week 8)*

**Resources**

You may use any code samples posted on the module’s blackboard site. The following are also useful resources.

* OpenGL Programming Guide: The Official Guide to Learning OpenGL, Version 4.5
* <https://www.khronos.org/opengl/wiki/Getting_Started>
* [www.opengl.org](http://www.opengl.org)
* <https://learnopengl.com/>
* [www.Opengl-tutorial.org](http://www.Opengl-tutorial.org)

**Marking scheme**

The following topics will be used when marking the solution.

1. Game Design (Objective, UI).
2. Motion of the user controlled vehicle & its interaction with

the boundaries of the tracks/channels in the game environment.

1. Collision detection & Response between the objects in the environment.
2. Computer Controlled objects and their movements
3. Graphical Quality of the environment (textures, sprites, background).

The next page shows the marksheet that will be used for the assignment.

CMP-5010B – Computer Graphics I: Coursework Demo

Student ID:

Place small screenshot of game here!

Student Name:

Marker:

Date:

TITLE:

I. **Essential** features of the game:

1. The game has a clear starting position **□**
2. The user’s controls are clearly stated. **□**
3. There is a clear objective. **□**
4. A method of tracking who wins is present.  **□**
5. The window scrolls horizontally and vertically. **□**
6. A track or path should be defined with clear boundaries for the user **□**

to navigate around.

1. There should be a response to colliding with the boundaries (e.g. the **□**

vehicle bounces off or slows down).

1. There should be at least one other vehicle or object that **□**

prevents the user from winning

1. Collisions should be detected between the user and the other vehicles **□**

in the environment.

BRIEF DESCRIPTION and COMMENTS:

II. **Additional marks** for:

* The vehicle is animated

(use of different sprites/textures) **□**

* Sound **□**
* Collision Response beyond a simple stop or slow down **□**
* Special effects e.g. particle systems **□**
* More advanced AI for NPC’s **□**
* More than one level **□**
* A level editor **□**

COMMENTS:

III. **Marks deducted** for:

1. Unstable execution (e.g. game freezes or exits unexpectedly). 🞏
2. Incorrect blending of textured polygons. 🞏
3. Incorrect scrolling when vehicle/character reaches edge of viewport. 🞏
4. No clear objective 🞏
5. Incorrect collision detection (objects do not collide at all or interpenetrate). 🞏
6. Cluttered or overly complex game which is not playable by an ‘average’ player. 🞏

COMMENTS:

MARK based on criteria:

REVISED MARK after discussion between markers: