UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG
SCHOOL OF COMPUTER SCIENCE & APPLIED MATHEMATICS

COMS3006A: Computer Graphics and Visualisation Project Outline

Dr. Richard Klein

Semester 1, 2020

1 Introduction

In this project, you will work in teams of 4 to build a 3D computer game with 3 levels/stages that run in the browser (Chrome or Firefox on Ubuntu) using the Three.js graphics framework. Your game should demonstrate mastery of various technical aspects of graphics, for example, hierarchical modelling, viewing and cameras, lighting and materials.

Beyond these basic requirements, you are free to design any 3D game that you like. I encourage you to do something exciting and interesting. There are grades allocated to polish (have you put effort into the look and feel of your game beyond what HTML and Three.js give you by default) and innovation (have you implemented something that is not obvious – e.g. networking, multi-player, special effects, novel models, animated models etc.).

2 Three.js

More information can be found at `threejs.org/`. Check out their 'Getting Started' and 'Examples' pages for assistance. The textbook also discusses the library in Chapter 5. The library itself provides a scenegraph API. You can load and build models, attach physics engines, lights and cameras to your scene. The library traverses the graph to render the scene much in the same way that we have seen in the examples in class.

You can place lights/cameras in the world, or attach them as children of objects which allows them to move together in the scene. You need to think carefully about the hierarchy of these objects in your world. Both the sensibility of your design as well as the motivation thereof will be assessed.

The reason we choose Three.js is that the framework operates at a high enough level that you can create interesting and impressive games but at a low enough level (relative to Unity) that you still have to demonstrate a grasp of the design concepts considered in this course.

Note that Three.js is a JavaScript library and all your code will be executed in a local browser. This means that you have to be cognisant of efficiency.

3 Grading

You will be graded on the following aspects of your project. They will be assessed as is relevant to your game, but you should show a meaningful effort to illustrate each concept within your game.

Viewing (10%) Are you able to load up a 3D scene? Is the scene animated? Can the user change the view in some way? Can the camera move through the world in some way? Are there any animation glitches? Is there a 3D avatar for the main character or other characters/elements of your game? Do you have objects that move with the world and items that move with the camera? Do you have

multiple views (e.g. first- and third-person views, or an orthogonal project as a minimap)? Do you have a picture-in-picture?

Control (10%) Keyboard and mouse controls are required. Are they smooth? Are they simple and logical for your game? Do they effectively control the scene/view/avatar as relevant?

Playability (10%) Does the game have an objective? Is the player able to succeed and fail at this objective? Is it competitive and fun? Is the game played in all three dimensions (for example, you want to do more than implement a 2D platformer that looks 3D – controls and movement should work in all three dimensions as is relevant for your game)? Does the game have a working physics model (as is relevant to your game)?

3D Effects (15%) Demonstrate the use of various graphical and 3D effects that are discussed in the course. For example antialiasing, depth tests, colour, multiple light sources, smooth/flat shading, curves, surfaces, static/dynamic skyboxes, shadows, reflections, refractions etc. Demonstrate the use of all these ideas in your game to improve your grade. Using these ideas in a visually appealing way contributes to polish while using them in interesting and unusual ways can also contribute to innovation.

Coding Style (10%) You should implement all your code using best coding practices that you are learning in other modules such as Software Design. At a minimum, all code should use sensible variable/function/object/file names, be indented, commented, documented, version-controlled (git). Correct use of Object Oriented Programming (OOP) techniques where appropriate will be rewarded.

Design Style (25%) The quality of your hierarchical modelling and your motivation of that design. Explain how the world/objects/subobjects were modelled and why. Discuss the advantages and disadvantages of your approach. Grades are awarded based on your design and your ability to articulate and criticise your design.

Polish (10%) Impress me with the look and feel of your game. Can the game restart rather than having to refresh the page? Does it lag? Is there a dashboard and/or in-game menu? Is there a colour scheme? Have you added extra bells and whistles?

Innovation (10%) Impress me with new ideas. Try something different for your game idea, use your own models/textures designed in Blender, consider using other effects/techniques not considered in the course. Networking? Multi-player? Sound? Efficiency? Speed? Add the extra value that makes your game memorable.

Note that you may use some character and texture models from online, but these should *all* be referenced and acknowledged in a credits screen accessible in the game.

To earn the last 20% of the grades in the Polish & Innovation sections, your project should have something that makes it stand out as special. It should be memorable.

4 Groups & Mentors

You should work in groups of 4 which should be captured on Moodle. The head tutor for CGV will allocate a mentor to each group from the pool of CGV tutors. During each lab session, you should meet with your tutor for a few minutes to ask questions and get advice/ideas. It is your responsibility to approach your tutor with questions that you may have.

5 Assessment

You will have 3 opportunities for feedback.

5.1 Alpha

The alpha assessment is not for marks and is purely a formative assessment. You should have Three.js up and running. You need to have some preliminary implementation that you can use to illustrate to your mentor what the game will ultimately look like.

Your mentor should walk through the rubric with you, and you need to be able to answer questions about how you have implemented/plan to implement the required functionality for each aspect of the rubric.

5.2 Beta

The beta version of your project will be graded and counts as the 'beta project' mark towards your class mark. This will be graded according to the rubric by a combination of tutors and myself. You will need to demonstrate your project in action, talk us through the game, and be able to answer our questions about design choices, special effects etc.

It is acceptable if your project has minor bugs at this stage, or if you still need to complete a level or two, but the project should fundamentally be finished and almost ready for production. This is your last chance to get formal feedback before your final hand-in/demonstration.

5.3 Final Project Submission

The final project submission will be graded according to the rubric by myself and the tutors. Everyone will be graded and should demonstrate their game, multiple times. You need to submit a maximum 2-page pdf describing/analysing/motivating/criticising your design decisions. This will count towards the practical exam mark of the course.[1]

Overall, the beta and final submissions will count around 30% of your final mark.

5.4 Individual & Group Assessment

The project will be assessed as a group. However, similar to group work in Software Design, each group member will submit information (via Moodle), about what their and others' responsibilities were, what they contributed to the project, and in what proportions they think the grades should be awarded in the group. Your individual mark may be adjusted by up to 20% based on my assessment of these results.

[1] Although this does not preclude other practical questions in the final exam.

6 COVID-19

Based on issues around the ongoing COVID-19 pandemic, it is unclear how the rest of this semester is going to unfold. Perhaps the end of semester will be postponed, perhaps contact time will be replaced with forms of online learning. It is not clear whether the labs will be open or not. At this stage, it is unclear what the way forward will be. Because of this, it is difficult for me to provide deadlines in the usual way.

Provisionally, your Alpha will be due on the 7 April 2020 – this has been moved back 1 week. Depending on the health & safety considerations, we may take these meetings online over Google Meet or another similar platform.

The provisional date for the Beta version of the project is on Tuesday 30 April 2020 (also a week later) and the final submission is at the end of semester – Friday 15 May 2020 with presentations on the following Monday and Tuesday – as originally planned. These may take place online, may be postponed, or reconfigured based on declarations made by Wits leadership.

Your GSuite account with Wits gives you access to Google Hangouts and Google Meet. I do suggest making use of these and other online tools (whatsapp/zoom/github/gitlab/private moodle forums etc.) to chat with your group members and start thinking about and working on the project if you are able to.

I will be as accommodating as possible when dealing with issues arising from COVID-19, but please let me know early if you have laptop/connectivity/health issues. Remember that your health and that of those around is a priority.