3D Virtual World

Graphics I (CSA2207) Assignment
Sandro Spina / Colin Vella
March 15th 2010

This is the description for the assignment of unit CSA2207, Computer Graphics I, for the year 2009/2010. This assignment is worth 40% of the total mark for this unit. You may carry out this assignment in a group of at most four students. If you choose to work in a group, make sure that the work gets evenly distributed. Under no circumstances should code be shared outside your group. Assignments should be submitted by hand on optical medium; each team will be asked to demonstrate their implementation and each member of the team should submit his own documentation. Please remember that plagiarism will not be tolerated; the final submission must be entirely your work.

Deliverables

You will submit your project source code, executables and a PDF file containing the project documentation on optical medium. Ideally binaries should be made to run straight from CD/DVD-ROM without the need for any installation unless specified in the accompanying document, in which case a detailed installation guide must also be provided. DO NOT submit with the assignment a printout of your source code.

Background / Description

For this assignment you are asked to create, render and navigate a 3D virtual world. You are allowed to create (or use freely available) 3D geometry to populate your world which would ideally consist of a terrain, skybox and a number of buildings spread across the terrain. All your geometry should be contained within a spatial data structure which is used to optimise the rendering of your world. You should also make use of proper lighting techniques to simulate directional, point and spot lights in your virtual environment. Each team is allowed (or rather expected) to be creative and come up with different ideas to populate this virtual world. Each team is also encouraged to be creative in choosing which virtual world to create. The application may be developed in C#, Java, C or C++.

Task Breakdown and Marking

The project is broken down into a number of tasks, each of which outlines a different aspect of the final deliverable. I am not attaching any marks to each component simply because each one of them can be dealt with at varying levels of difficulty. I will then determine the mark when reading the documentation of each member of the group. Note that members can potentially get different marks for their assignment depending on their contribution to the project. Each member of the group can opt to cover one or more of the tasks mentioned below.

Task 1: Creation of a common framework/core within the group.

Since this is potentially (you can still opt to work on your own) a group assignment you shall be required to design and create a common core component to which different aspects of the project will plug into. This will essentially consist of your main gaming loop. Remember that a spatial data structure (Octree, BVHs, etc) containing all your meshes needs to be used in order to accelerate rendering.

Task2: Creation of content and population of Virtual World

The aesthetics of your virtual world depend on the quality of the assets used to populate it. For this part of your assignment you are required to create, re-use (and possibly modify) 3D content / textures to represent buildings (or whatever geometry makes up your virtual world). You will also need to specify the world coordinates (placement) of the different geometric objects. A procedural approach can be used here in order to generate (for eg.) different object placement with every render of the world.

Task3: Terrain and Sky Generation

Your world (assuming this is not outer space) needs a terrain. In this part of your assignment you should create the necessary geometry to represent it. A flat terrain (basically two triangles) is clearly very easy to implement so you need to consider more complex types of terrain with a varying height field. Once you have your terrain you also need to cover your world with a skybox. A skybox simulates (using textures) a distant environment such as mountains on the horizon, sky, etc ... Depending on the virtual environment you are simulating you should create an appropriate skybox to represent it.

Task4: Lighting your World

Illumination is an important aspect of your world. For your implementation you need to make use of shader programs which implement different types of illumination. Light sources would then need to be integrated into the virtual world in order to illuminate its different parts. Clearly the positioning of all these light sources in the world is very important.

Task5: Creativity

You should try to be as creative as possible in your assignment. In this task you are allowed to implement anything which might make your environment more interesting (or look more realistic). A couple of examples are shadows and particles. But you can also come up with different ideas such as implementing a 'game like' component within your environment.

Task6: Assignment Write-up

This task is extremely important as you are expected to clearly describe how you architected and built your virtual world components, outlining any important design decisions taken. You are encouraged to use diagrams to explain the architecture where necessary; moreover, feel free to use pseudo-code to illustrate the algorithms implemented. If the deliverable has short-comings or bugs or even lacks some of the expected functionality, you must put this down in the write-up.

Final Notes

Remember that the first step in achieving a solid, stable and efficient deliverable is that of sitting down as a group and designing the system before anything else. Identify how everything is supposed to work at a high-level, and then start by singling out the various components of your virtual world. Once you identify the components, start categorising the entities and their relationships, how they interact and work within their sub-systems, and how the sub-systems communicate together within the system. Outline all this design, and make sure you are able to describe to yourself precisely what each component is responsible of and how it would be carrying out its duties. Make sure you do this before doing any programming as this will save you time discarding broken designs or trying to patch badly designed components to make them talk, and will let you concentrate on the actual problem at hand. Good luck, feel free to experiment and try to enjoy this assignment. The effort put into this assignment will be duly noted during marking.