

CE318: Games Console Programming

Lab 3: Introduction to 3D

28th October 2011

1 Primitives

In the first part of the lab, you should draw some triangles using `TriangleList` and `TriangleStrip`, with and without indices, with and without buffers. You should then move on to more complex geometries, a closed cube, a pyramid and a cone (optional). You should try to code your geometries using OO design so that you can re-use them for the second part of the lab.

You can make use of the code template called `LAB3TRIANGLE.ZIP` (template code) available on the course web page. Also, in case you do get stuck: `TRIANGLE.ZIP` (drawing primitives) contains code for numerous primitives (and rotations) while `BOX3D.ZIP` (world transforms) uses translations, rotations and scaling for a simple model (a cube). Try to code the primitives yourself, however, as a good understanding is essential for the remainder of this course. And please ask questions if you encounter any problems.

At first, use `Matrix.Identity` and use the standard view and projection as defined in the lecture notes. You can also make use of the code for the `BasicEffect` as defined in the slides. This is all already included in the template code.

A detailed list of instructions is as follows:

- Draw a triangle using `TriangleList`
- Draw a triangle using `TriangleStrip`
- Draw a quad using `TriangleList` with indices
- Draw a quad using `TriangleStrip` with vertex buffer
- Draw a quad using `TriangleList` with indices, vertex and index buffer
- Draw a closed cube
- Draw a pyramid
- Draw a cone (optional)
- Apply transformations to each of these

2 Populating a 3D world

- Download the code `BASIC3DWorld` from the course website. Open the solution file, compile and execute. You should see a simple 3D world composed only by a grey floor and a blue sky. You can move around the world by using the cursor keys and the mouse.
- Take your time to familiarise yourself with the code. Pay special attention to the following:
 - The file `BasicModel.cs`, that is being used to handle the floor of the world.
 - The file `ModelManager.cs`, used to handle all the models loaded into the game.
 - The files `Camera.cs` and `FirstPersonCamera.cs`, that contain the code for the camera that you can move through the world.
 - The latter classes inherit from `GameComponent` and `DrawableGameComponent`. These XNA classes are game components that allows an object oriented design of the code of your game, as seen in the second lecture of this course.

Don't worry if these concepts seem a bit complex, we will cover most of them next week.

- Now, add to this world some of the primitive figures you learnt to create in the first part of this lab, according to the next requirements:
 - Draw at least three different shapes (but more than one of each) in different locations over the floor.
 - At least one of them must be completely static.
 - At least one of them must be moving constantly.
 - At least one shape must be in a non-stop rotation.
 - For the rest of the shapes drawn, combine movements and rotations at different speeds.