CS4351 Real-Time Graphics (2018/2019 Semester 1)

Programming Assignment #5

Release Date: 2 November 2018, Friday

Submission Deadline: 18 November 2018, Sunday, 11:59 PM

TASK

You are to complete an OpenGL program (a Tessellation Control Shader (TCS) and a Tessellation Evaluation Shader (TES)) to perform procedural displacement mapping by using the tessellation capability in the OpenGL rendering pipeline. The following images show sample views of the result that your program is expected to produce:

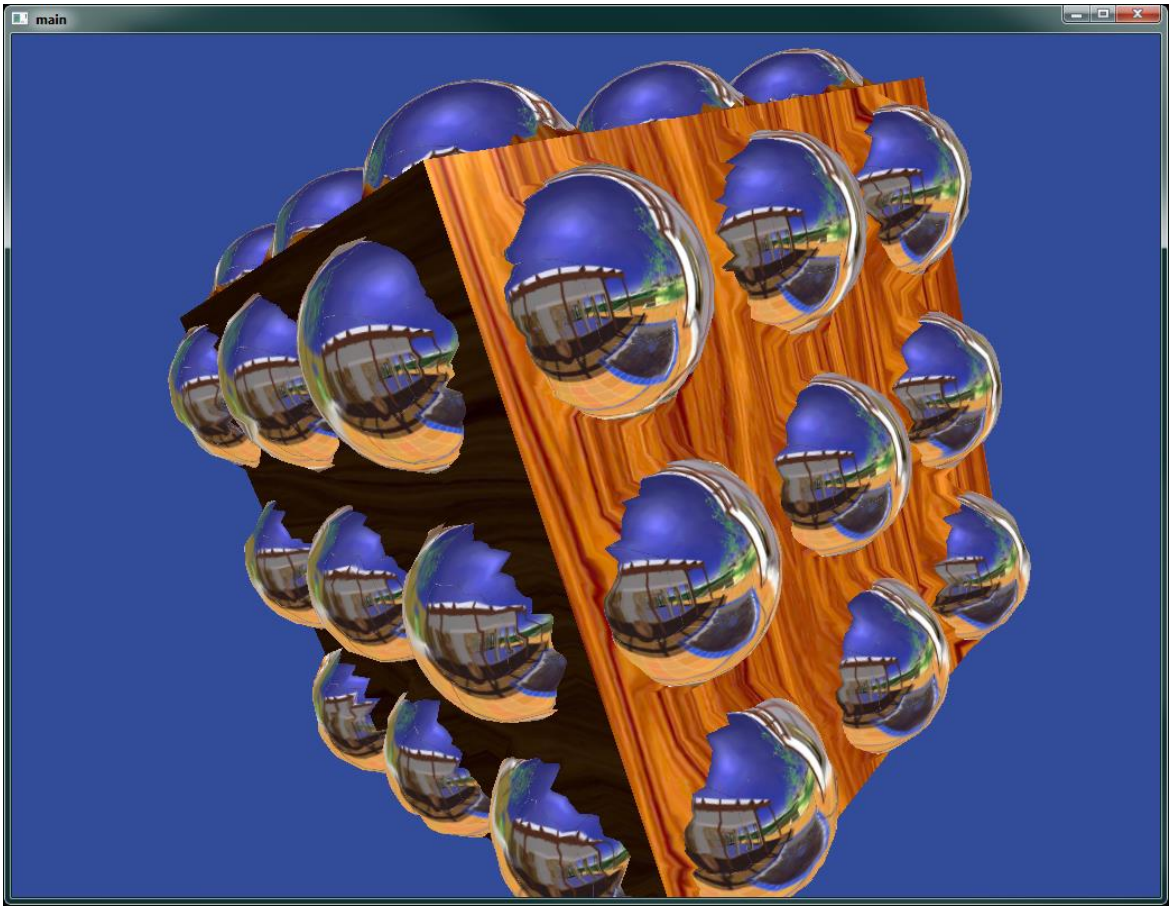

Figure 1

In Figure 1, you should be able to recognize that this is similar to the wooden cube that you have worked with in Assignment #1, only that this is augmented with tessellation-based procedural displacement mapping, which has given real geometry to the hemispherical mirrors.

The reflection mapping on the hemispherical mirrors are still performed using the same procedural bump mapping method to compute the perturbed normals, which in turn are used for computing the reflection vectors.

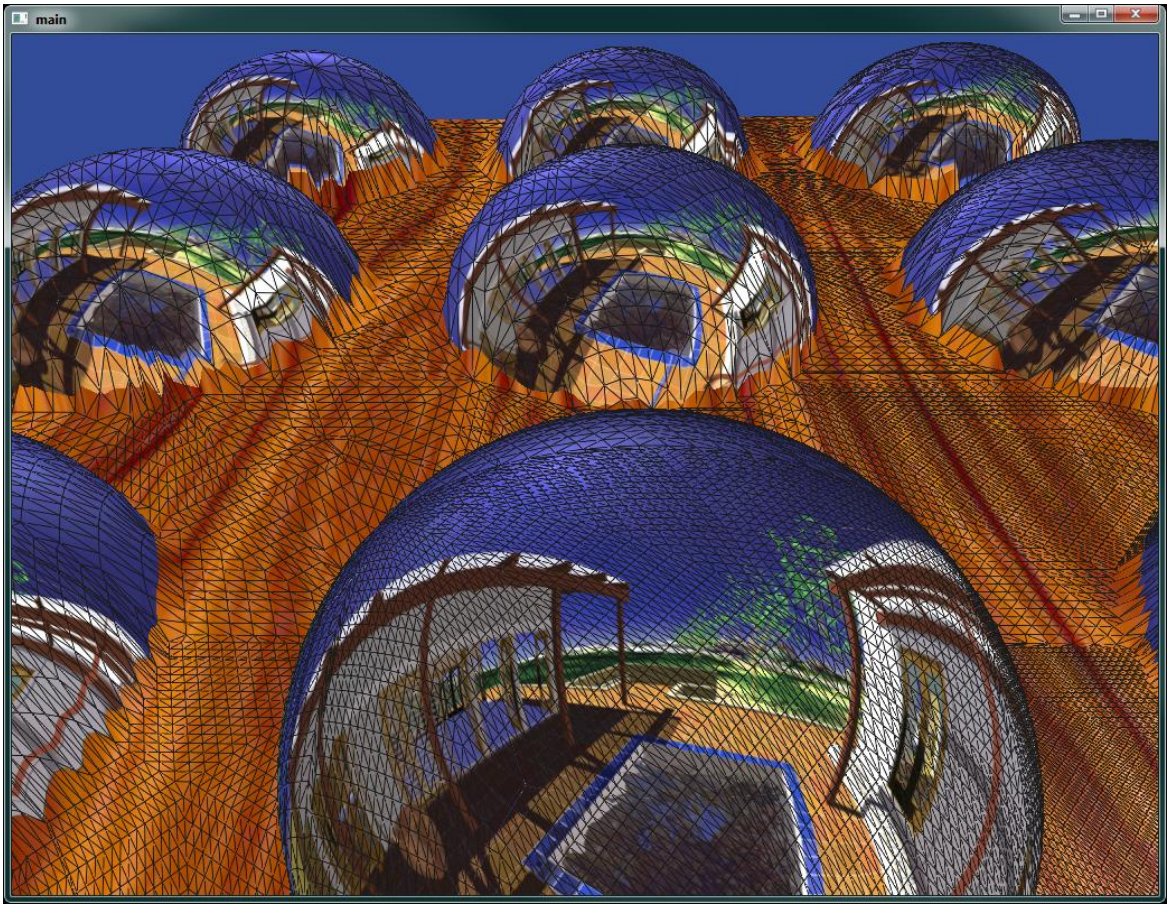

Figure 2

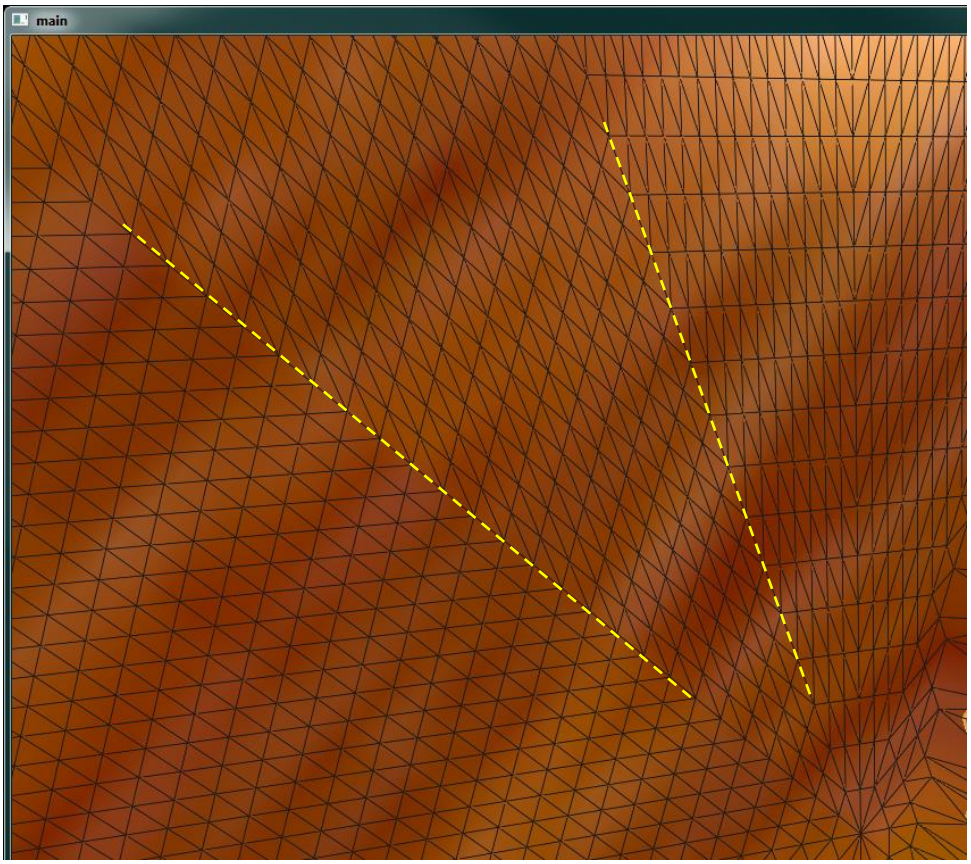

Figure 3

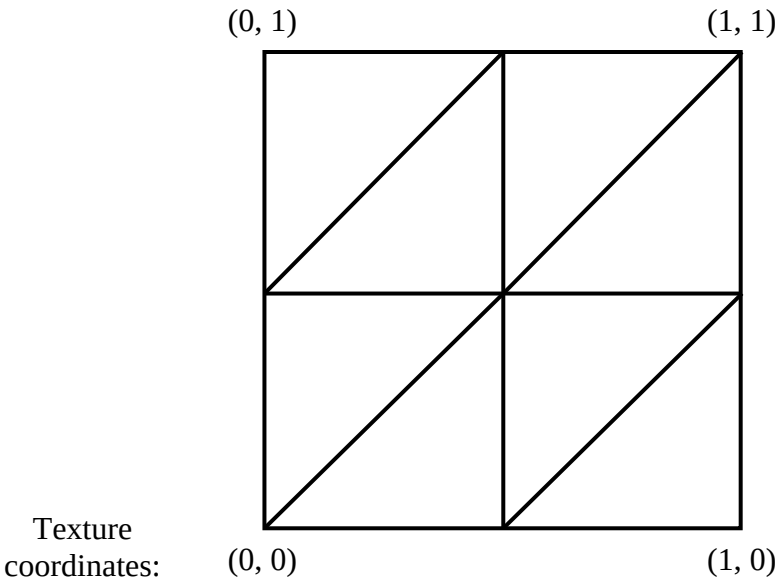

Figure 4

Each face of the cube is specified by a mesh of **triangle patches**. Figure 4 shows an example of such a mesh of triangle patches. The corner vertices of a face have been assigned texture coordinates as shown in Figure 4.

You need to complete the **Tessellation Control Shader (TCS)** to compute the **tessellation levels** for each triangle patch. The tessellation levels are proportional to the pixel lengths of the patch edges — the longer the edge (in the number of pixels on the screen) the higher its tessellation level. The effect can be seen in Figure 2, in which you will see that the hemispherical mirrors that are further from the viewpoint are less tessellated than those nearer to the viewpoint. The detailed requirements can be found in the given TCS source code.

Another requirement for the tessellation is that patches that share a common edge must be tessellated equally along that edge. This is to avoid cracks between patches after tessellation. You can see this result in Figure 3, in which two shared patch edges have been marked with yellow dashed lines.

You also need to complete the **Tessellation Evaluation Shader (TES)** to compute the 3D positions of the vertices of the triangles created by the tessellation primitive generator (TPG). Each new vertex is displaced (lifted up) by an amount depending on where it is in the "tile space". In the TES, we will not compute the normal vectors of the displaced vertex, instead we leave it to be computed in the Fragment Shader, where it will be derived procedurally. The detailed requirements can be found in the given TES source code.

Something to think about. You can notice that the boundaries between the wooden surface and the hemispherical mirrors are extremely distorted. What are the causes of that? What can we do to avoid that kind of artifact?

Please download the ZIP file **cs4351_1819S1_assign5_todo.zip** from the **Assignments** folder in the IVLE Workbin.

You need to complete the TCS **ProcDispMap.tcs.glsl**, and the TES **ProcDispMap.tes.glsl**. In both shaders, all necessary **uniform variables**, and **global input/output variables** have already been declared, and **you must not add new ones**. You can add new functions in your shader. Note that you should adhere to the **variable naming convention** where the prefix "ec" is used to indicate that the entity is expressed in the eye space, the prefix "wc" to indicate world space, and the prefix "tan" to indicate tangent space.

A Visual Studio 2017 solution **main.sln** is provided for you to build the executable program. The application program loads all the necessary shader source files, and use them in the rendering. It sets up the texture maps and also provides the values for the **vertex attributes** and **uniform variables** to the shaders. In this assignment, **you are not required and must not change any C/C++ source files**.

You should study **main.cpp** and all the other given shaders carefully to help you complete the TCS and TES. You can run the executable application program **main.exe** directly to test your shader.

The detailed requirements for each task can be found in the TCS and TES source code.

GRADING

The maximum marks for this programming assignment is **100**, and it constitutes **10%** of your total marks for CS4351.

Note that marks will be deducted for bad coding style. If your program cannot be compiled and linked, you get 0 (zero) mark.

Good coding style. Comment your code adequately, use meaningful names for functions and variables (adhere to the new variable naming convention), and indent your code properly. You must fill in your **name**, **matriculation number**, and **NUS email address** in the **header comment**.

SUBMISSION

For this assignment, you need to **submit only**

- Your completed **ProcDispMap.tcs.glsl**, and
- Your completed **ProcDispMap.tes.glsl**.

You must put them in a ZIP file and name your ZIP file **<*matric_no.*>_assign5.zip**. For example, **A0123456X_assign5.zip**. All letters in your matric. number must be capitalized.

Submit your ZIP file to the **Assignment #5 Submission** folder in the IVLE Workbin. Before the submission deadline, you may upload your ZIP file as many times as you want to the correct folder. **We will take only your latest submission.** Once you have uploaded a new version to the folder, you **must delete the old versions**. Note that when your file is uploaded to the Workbin folder, the filename may be automatically appended with a number. This is fine, and there is no need to worry about it.

DEADLINE

Late submissions will NOT be accepted. The submission folder in the IVLE Workbin will automatically close at the deadline.

——— End of Document ———