# Real-Time Rendering and 3D Games Programming

# ASSIGNMENT 1 – REPORT (v1.1)

## INTRODUCTION

Which shape did you choose to draw? Did you derive the algorithm on your own or did you find some other resource to help? List any sources used (books, articles, videos, ...).

<https://github.com/rosamarco/menger-sponge>

<https://thecodingtrain.com/CodingChallenges/002-mengersponge.html>

These are the resources used to help me create the Menger Sponge, where they helped with the initial process of creating cube objects to help describe the concept more cleanly. By having loops that went through making 20 cubes from 1 cube that made the basis of a single Menger Sponge at subdivision 1, it helped conceptualize the idea in programming. However, this left the idea of making the cubes being created at the right locations and with the right vertices, so I had to add my own algorithm of having each cube be created around the vertices of the center cube, as it had the most consistent vertices throughout each subdivision.

Describe the hardware you used to perform the tests described in this report. Include detailed CPU and GPU information. What screen resolution and refresh rate did you use?

CPU: Intel(R) Core(TM) i5-8500 CPU @ 3.00GHz

GPU: NVIDIA GeForce GTX 1080

Screen Resolution: 1920 x 1080

Refresh Rate: 60 Hz

Describe your data structure and algorithm. Are you duplicating vertices that are used by multiple triangles or did you implement shared vertices? Are these cases where multiple faces might overlap? Which OpenGL drawing primitive are you using?

I majorly used c++’s vectors, which are flexible arrays that did not need to be manually resized, as I figured it would be better for the cpu to handle the data structure whenever it needed to, rather than personally pinpointing every step of when to manually resize a regular array. This allowed further flexibility for when I had to calculate and replace vertices, and store them within other vectors, which resulted in some 2D and 3D vectors. Furthermore, I have duplicating vertices within my common, shared data structure, although for some certain scenes I had to manipulate the common shared data structure within the individual scenes to make them work with the various OpenGL drawing methods. Although generally, all scenes had data structures with shared vertices. Additionally, I used GL\_TRIANGLES as the main drawing primitive for all my cube rendering.

How did you choose to colour the shape? How many materials did you use and how were they assignment to faces? How did you 'communicate' face material data to the Shaders?

All faces of a cube had the same colours across each, although the colours present in scene 1 will differ from those in scene 2-5. Furthermore, the ambient and diffuse materials were assigned per vertex, with the specular added if there was only light active. Additionally, scene 1 used immediate mode material assignment through glMaterialfv, whereas the materials for scene 2-5 was communicated through uniforms in the fragment shader.

How have you decided to position each light source? How did you assign light colours to show off the full capabilities of your lighting model?

Answer here.

## SCENE 1

Start your testing at subdivision level 1 (base), Lighting On (1 light), Backface Culling On and Depth Testing On.

Create a table showing the average frame rate, number of vertices and number of faces at each level of subdivision that your hardware can handle with a frame rate greater than 1 frame per second.

Answer here.

Draw a chart showing the average frame rate achieved at each level of subdivision.

Answer here.

Run some tests with Lighting Off while keeping everything else as above. Describe the impact this has on frame rate and why? Use a table and a chart to show the data.

Answer here.

Run some tests with Backface Culling On and Off, while keeping everything else as above. Describe the impact this feature has on frame rate and why? Use a table and a chart to show the data.

Answer here.

Run some tests with Depth Testing On and Off, while keeping everything else as above. Describe the impact this feature has on frame rate and why? Use a table and a chart to show the data.

Answer here.

Run some tests with Backface Culling On and Off in combination with Lighting On and Off, while keeping everything else as above. When Lighting is On is there a difference in Frame Rate when Backface Culling is On vs Off? Describe Why or Why Not and show data to support your answer. Did you expect there to be a difference? Why?

Answer here.

Run some tests with Depth Testing On and Off in combination with Backface Culling On and Off, while keeping everything else as above. When Depth Testing is On is there a difference in Frame Rate when Backface Culling is On vs Off? Describe Why or Why Not and show data to support your answer. Did you expect there to be a difference? Why?

Answer here.

Discuss the performance characteristics of adding lights to the scene. Include a chart showing impact on frame rate for number of lights from 0 to 9. Discuss the shape of the curve and what it means.

Answer here.

Is there anything you found interesting or unexpected while running the above tests? Explain why.

Answer here.

## SCENE 2

Start your testing at subdivision level 1 (base), Lighting On (1 light), Backface Culling On and Depth Testing On.

Describe how you have decided to handle normal vectors. Are you specifying them per-vertex or per-face? Are you calculating them on the CPU or GPU? If CPU, how do you communicate them to the GPU? Are you storing them in a data structure or are you calculating them when needed in the shader?

Answer here.

Vary the subdivision level and move around the scene. Describe the performance characteristics you're seeing at the different levels of subdivision? Is the scene getting smoothly animated as you move around? Does it seem to speed up and slow down depending on what's currently being rendered? Why? At what level of subdivision do you start to notice that your machine is struggling with the drawing load? What are some things that **might** be causing it to 'struggle'?

Answer here.

Create a table showing the average frame rate, number of vertices and number of faces at each level of subdivision that your hardware can handle with a frame rate greater than 1 frame per second.

Answer here.

Draw a chart showing the average frame rate achieved at each level of subdivision. Compare this to the results you had for Scene 1. What is the data telling you about Immediate Mode vs Modern Mode? What sort of speed-up are you seeing?

Answer here.

Run some tests with Lighting Off while keeping everything else as above. Are the performance characteristics similar as for Scene 1? Why or Why Not? Use a table and a chart to show the comparison.

Answer here.

Run some tests with Backface Culling On and Off, while keeping everything else as above. Are the performance characteristics similar as for Scene 1? Why or Why Not? Use a table and a chart to show the comparison.

Answer here.

Run some tests with Depth Testing On and Off, while keeping everything else as above. Are the performance characteristics similar as for Scene 1? Why or Why Not? Use a table and a chart to show the comparison.

Answer here.

Run some tests with Backface Culling On and Off in combination with Lighting On and Off, while keeping everything else as above. When Lighting is On is there a difference in Frame Rate when Backface Culling is On vs Off? Describe Why or Why Not and show data to support your answer. Did you expect there to be a difference? Why?

Answer here.

Discuss the performance characteristics of adding lights to the scene. Include a chart showing impact on frame rate for number of lights from 0 to 9. Discuss the shape of the curve and what it means. Is there any difference between these results and Scene 1 results?

Answer here.

Is there anything you found interesting or unexpected while running the above tests? Explain why.

Answer here.

## SCENE 3

Create a table and chart showing the frame rate for each level of subdivision your machine can handle with a frame rate greater than 1 frame per second.

Answer here.

Is this what you expected? Why or Why Not?

Answer here.

Use a table and a chart to show the difference in performance between using GL\_STATIC\_DRAW and GL\_DYNAMIC\_DRAW in your calls to glBufferData(). Run the tests manually by changing the code and recompiling your project.

Answer here.

Discuss the results and whether it is what you expected and, if the two differ, why you think they differ.

Answer here.

## SCENE 4

Is there any difference in performance compared to Scene 3? Is this what you expected? Why or Why Not?

Answer here.

## SCENE 5

There are two sets of position coordinates in your C++ vertex array for this Scene, with three floats each, representing "home position" and "morphed position" for each vertex. You have changed the Vertex Array Object to use the morphed position as the position attribute that is used by the vertex shader. Use RenderDoc to find this data and confirm whether, on the GPU, only the morphed position is being sent across (3 floats) or both the morphed position and the home position (6 floats). Include a screenshot from RenderDoc showing this.

Answer here.

Is this what you expected? Why or Why Not?

Answer here.

## SCENE 6

Show a table and a chart comparing the performance (frames per second) of Scene 5 and Scene 6 at different model subdivisions.

Answer here.

Discuss what the data is showing.

Answer here.