CPS511 Assignment 3:

Submarine/Dirigible Simulation with Terrain

Due Date: Dec 2 at 11:59pm

In this program, you will merge your assignment 1 and assignment 2 code and then extend your program to create an interactive simulation of a submarine with a pseudo-realistic terrain of hills and valleys on the ocean floor.

This programming assignment will exercise all of the course concepts we have learned. **You may do this option alone or in groups of two. For groups of two, more will be expected. Do not attempt to find source code on the web for this assignment. It will not help you and you risk extremely serious consequences.** Begin designing and programming early! This project is worth **20** percent of your mark. **If there is some part of the assignment you do not understand, please see me (or email me) as soon as possible and I will clarify the issue.**

NOTE: if you wish to do the Dirigible option rather than a submarine, substitute the "ocean floor" with a "hilly terrain". In all other required functionality described below, substitute "dirigible" for "submarine".

Program Functionality Requirements

- You will create a hilly ocean floor terrain using your assignment 2 functionality and load it into your mesh data structure. That is, when the TA runs your A3 program, the ocean floor terrain mesh will already be hilly. You must then add texture-mapping to your terrain. You can limit your submarine depth to prevent the submarine from colliding with the terrain.

- **Teams of two**: you must allow the user, if desired, to lower the submarine to just above the terrain but no lower. As the sub moves forward at this depth, a *collision detection* algorithm must “look ahead” to determine if the terrain depth is higher and therefore the sub would collide with it. In this case, slow and stop the sub. The user can then turn or raise the sub until a collision will no longer occur and the sub may go forward again. Hint: one way to do this is to use a sequence of a few bounding spheres along the length of the sub for collision detection. Check if points are inside one of these spheres. The spheres could extend from the front of the sub to create a “collision look ahead” capability. Or use a bounding cylinder.

- You are to extend your submarine from assignment 1 and add the following functionality:

    a. Texture-map your submarine.
    b. Add a working periscope. The periscope can be raised and lowered using keys. Add the capability of changing the camera view such that you can look out the periscope (i.e. add a “periscope view”). Use a key to switch between normal world view and periscope view. This view can be in a separate viewport or separate window if you prefer.
    c. **Teams of Two**: the periscope can be rotated to look around.

- Implement a “help” key as in assignment 1 to tell a user/TA how to use your program.

- **Add a second submarine that is not controlled by the user but rather moves around your scene. Periodically (I suggest you use a timer) have this submarine make a random change of direction.**
- **Teams of two**: your user controlled submarine must not interpenetrate the second submarine if they are about to collide. That is, you must implement submarine collision detection and collision resolution. If the submarines are “close” to each other and may collide, you should turn one of them in a direction such that the collision is avoided. Whatever collision

resolution you choose to program, document it in a README file for the TA.

- Add the capability of firing a torpedo from your submarine (key activated). If you wish, you may attach a hollow cylinder to the side of your sub where the torpedo will emerge from.
- If the torpedo interpenetrates the second submarine (i.e. collides with), then it should disappear. Hint: use a bounding box (a sphere, a cylinder or a rectangular block). Another submarine should then randomly appear in another part of your world to replace it. You may add a “score” counter to your program and make it a game, if you wish. No marks are awarded for this game functionality though.
- **Teams of Two**: same as above except that you must use some special effect to destroy the second submarine (e.g. an explosion, sub breaks into pieces and pieces disappear etc. Document your special effect in a README file).

- **Final Requirement**: To be eligible for full marks on this assignment you must use a GPU vertex shader and a GPU fragment shader to render your scene. I will post an example program that will help you write the shaders. If you choose **not to** implement this requirement then the most you can receive is 17/20.

Bonus

- Use bump mapping somewhere in your program, implemented in a shader (you may other objects in your scene)(1 mark).
- Use a stencil buffer or shader to create a circular window for the periscope view (1 mark).
- Add some other special effect (0.5 marks)

The TA will judge the difficulty of your added functionality when deciding on bonus marks. The maximum bonus marks are 2.

Grading (Out of 20 points)

Texture-mapped submarine with all functionality from A1, plus raise/lower periscope and periscope-view	5 points (**teams of two**: periscope must be able to rotate around by the user)
Hilly, Texture-mapped Terrain	2 points
Second submarine that moves around randomly	2 points (**teams of two**: submarine collision detection and resolution)
Torpedo can be fired from sub	2 points
Collision detection of torpedo with second submarine. If hit, second sub disappears and another sub appears in random location.	5 points (**teams of two**: if hit, second sub disappears using explosion or some other special effect)
Use of vertex shader and fragment shader to render scene	3 points
Help key	1 point
Bonus	2 points
Total	22 (including bonus)

Program Submission

Use D2L to submit your assignment. Zip up all your source files and submit the zip file. Include a README file describing your interface, what works, what doesn't work etc. to help the TA during marking. **It is your responsibility to ensure the TA has enough information so that he can, with little effort, compile and run your program.** I am being flexible in terms of your programming language and operating systems choice so you must make an effort to meet me halfway. If the TA has trouble compiling your program, he will have the discretion to deduct marks and/or he will ask you to compile and run your program in his presence. **Do not include executable files (they are too big).**