3D Graphics Project – Projet Graphique 3D
Sea diving

This worksheet describes the project assignment you will supply for the end of semester. You will work on this project during the practical sessions. You are asked to provide rendering and modelling solutions arising for the proposed project scenario, and will present your work during a mini-defence at the end of the semester. Please check the course webpage periodically for additional guidelines.

1. Practicalities

Grading Criteria. Pertinence of proposed choices and solutions, elegance and efficiency of the proposed code, the quality and difficulty of implemented techniques and the presented results. The grade for this project counts for 50% of the total 3D Graphics grade.

Teamwork. Projects are in groups of 3. A very small number of 2 person groups will be tolerated for parity reasons. Projects should be teamwork; one person shouldn't be doing everything and this will be verified during the presentation. If a difficulty arises in the group, be sure to notify your professor as soon as possible (e.g. only days before the assignment deadline is too late). Be efficient during the practical sessions in order to avoid homework, so as to leave yourself time for other semester projects. Don't wait until the last minute.

Project deposit. You will deposit an archive of your project on Teide at a later specified date. This code ***must*** easily compile and run on Ensimag PCs as reference platform. You can still code at home but should ensure compatibility. The archive can also contain a presentation video of your project to enhance the interactive behaviour of your program. (We will provide you piece of code to help you making a video). Your deposited archive name should be **G3D-NAME1-NAME2-NAME3.tgz** where NAME* are group member last names. It should decompress ***as a folder*** named **G3D-NAME1-NAME2-NAME3/**. Provide a ***README.txt*** file at the root of your folder explaining how to compile and run, directory organisation, and any practical information for the build.

Project presentations. Each group will present its project in front of the class and professors, and will demo its project based on the code deposited on Teide. You will be given 10-15 minutes with a PC and projector to present your work, ideas and contributions, including perhaps code fragments of your project that illustrate your choices and research (choose wisely as you do not have much time to develop examples). You may prepare some slides to go along with your presentation, but this is not mandatory. However, we need to see a live demo of your project, and may ask to navigate the code.

2. Subject Description: Sea-Diver

2.1. General Guidelines

We examine the case where a sea diver is swimming in the sea. He sees fish, corals, rocks, and then a **sudden, unexpected underwater encounter**… (use your imagination). A sea diver may have a flexible air tube connecting him to an air source. As he breathes, he releases bubbles of air. In your project, you will need to render and animate the sea-diver and other objects in OpenGL within the provided viewer framework.

The sea diver will have a simple humanoid structure, using the notions and model started in TP2-TP3. Basic motions of the diver (such as swimming, walking on the sea bed, hunting, …) are expected in the demonstrated animation. The diver can appear in a simple textured scene, but the scene modelling should be kept as simple as possible, focus should be on the diver. You may use basic lighting or animated particle systems to enhance the visual effects of moving sand, underwater light torch, air bubbles... The diver may have a flexible air tube, which can be animated using mass spring models, which will be seen in TP6. For the purpose of demonstration and grading of the work performed, any specific animation you design shall be triggered by a specific key explained in the joined README.txt file or by pressing F1 in the application.

A set of more detailed project guidelines is presented below.

2.2. Minimal Project Content & Challenges

1. **Light scattering**: under water, the light scatters with an effect similar to fog. Your renderings should simulate this.
2. **Texture.** The seabed below the diver is made of sand, and possibly rocks, seaweed, corals... The diver's suit or equipment could also be of non-uniform appearance. Use at least one occurrence of texture mapping to render some surface details of your choice.
3. **Articulated animation:** propose a simple articulated animation of the diver (e.g. swimming, hunting action…)
4. **Air bubbles**: propose solutions to represent and animate the air bubbles the player is generating in the water. (note that bubbles become bigger as they rise, and may merge to form larger ones before reaching the surface of the sea).
5. **Flexible tube, rope or net:** the diver is connected to the surface by a tube or rope which flexes as he moves or with currents, or uses a net for hunting. Propose a representation and physical model to animate this.

Some suggestions for advanced effects you may include:

6. **Fish school.** Propose a layered model to represent and animate the fishes. For each layer, define its goal and explain your choice of model among those studied in class. The last layer should be a geometric model of your choice (a mesh, a spline surface or an implicit surface), animated over time. Extend the model to get a fish school swimming away from the player.
7. **Caustics.** Light coming from above the water surface gives interesting light patterns you could model with advances lighting models. Some simplifications exist based on judicious use of animated texture mapping and blending.

8. **Reflection/Transparence.** The sea diver wears a reflective/transparent mask; his suit might also be reflective and bounce an image of the diver's surroundings.
9. **Particle system.** If the sea diver (or some other character you include) swims near the seabed, he lifts underwater clouds of sand.
10. **Shadows.** The sea diver or surrounding objects (fish) may cast shadows on the seabed or other objects.
11. **Other ideas.** You may think of other intersecting ideas to include (validate them with your professor before implementing).

APPENDIX: Recording a Video

1. Saving snapshots

In the file **viewer.h**, add the variable **bool toggleRecord;** to the class **Viewer**, and initialize it to **false** in **Viewer::init()**. At the end of **Viewer::draw()**, add the command **if (toggleRecord) saveSnapshot();**. Finally, in **Viewer::keyPressEvent()**, add the line **} else if (e->key()==Qt::Key_R) { toggleRecord = !toggleRecord;** just before the last **else**. You can now toggle the snapshot recording by pressing the key **R** when your program is running.

2. Building the Video

Once the snapshots are recorded, on Ensimag PCs, you can use the x264 codec to produce H264 quality videos of small size. Note that your input images width and height (resolution) should be multiples of 2.

Usage:

x264 -o test.avi image%04d.jpg

where image%04d.jpg denotes your snapshots numbered image0000.jpg image0001.jpg ... image000N.jpg (%04d means "integer with 4 digits") .

or for example:

x264 -o test.avi image%d.jpg

where image%d.jpg when your snapshots are numbered image0.jpg image1.jpg ... imageN.jpg (no zero preceding your frame number).