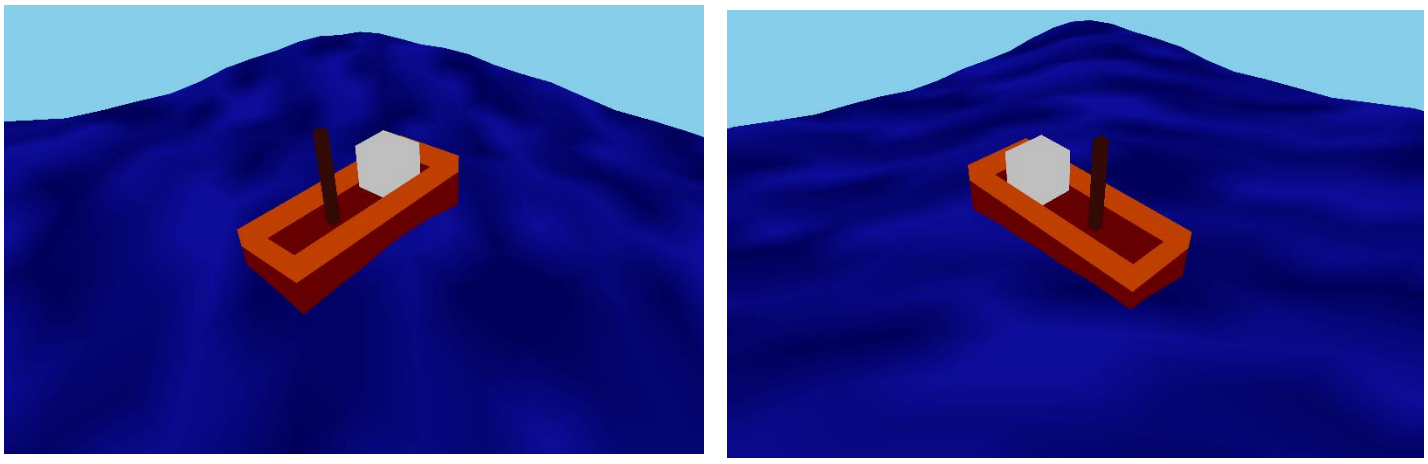
Proseminar Visual Computing
Winter Semester 2023

CG Assignment 1

Hand-out: November 14, 2023

Hand-in: November 27, 2023



Topics

- General OpenGL programming
- Transformations
- Basic animations and user controls
- Camera control

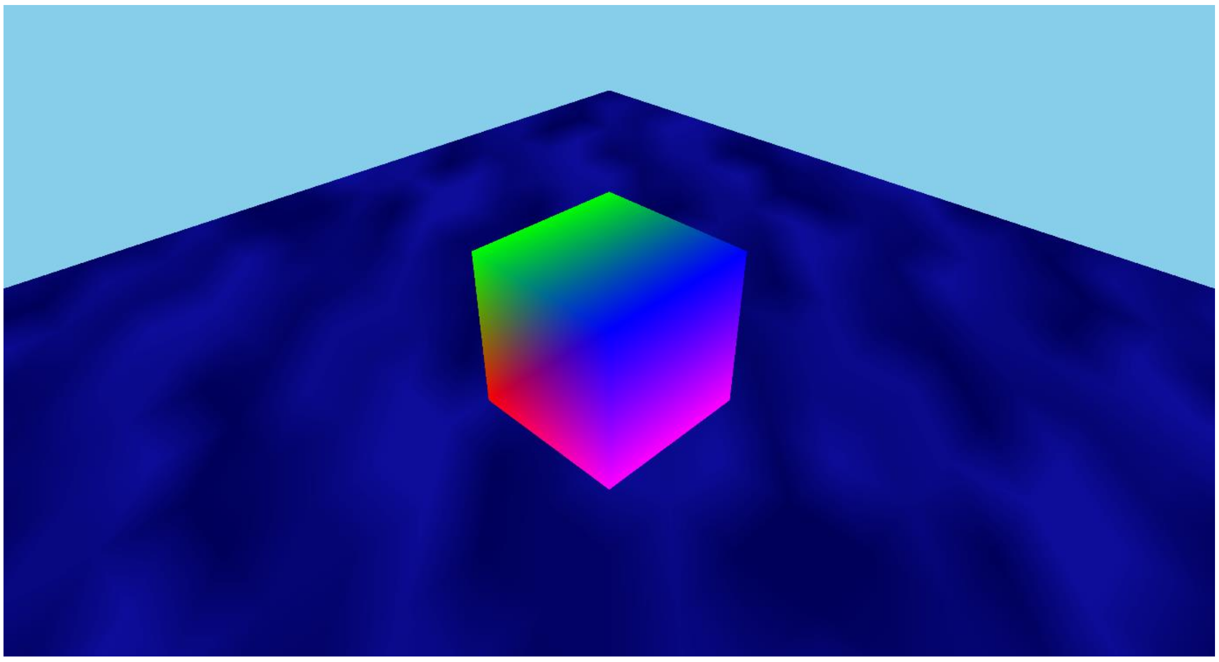
Outline

The goal of the Computer Graphics assignments of the Visual Computing PS is to build a controllable boat moving on an animated water surface. This work is divided into 3 steps. Each step corresponds to a programming assignment. In the first assignment, we focus on geometric transformations and animation. Its objective is to build a very simple boat made from basic geometric primitives (*i.e.*, cubes) by applying different transformations. The boat should be controllable by user input. Further, the water surface should be animated to simulate waves. Finally, an additional camera mode should be added that follows the boat. See the example video provided with this assignment for a working solution including all features.

Template code

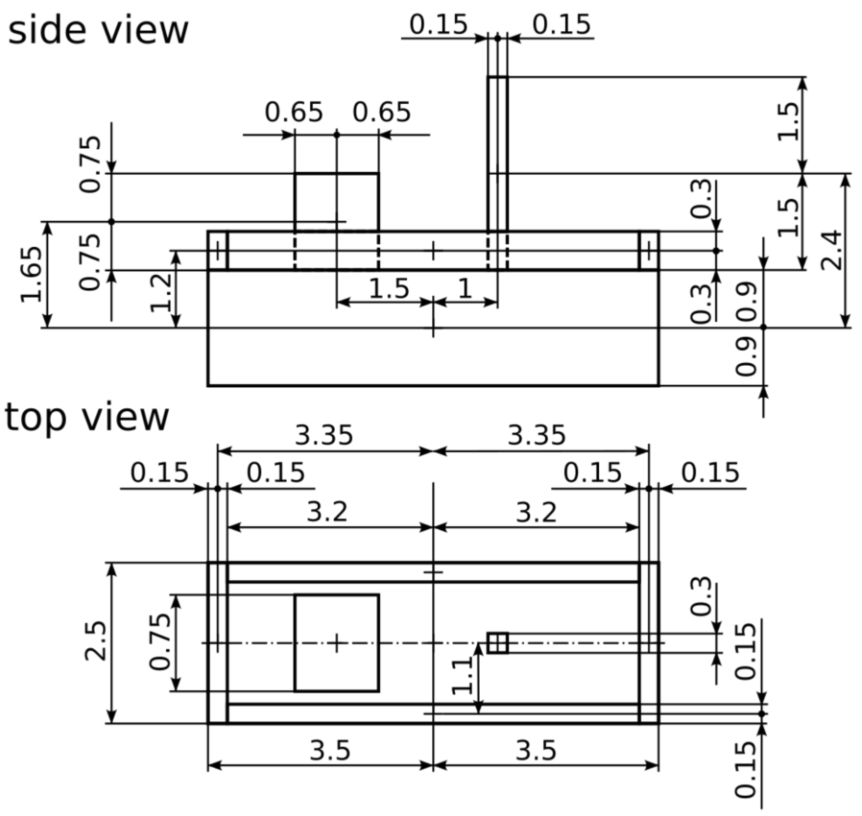
A template code is provided for this assignment. You can modify this code to build your own scene. Implementation can be done mainly in the main source file **assignment_1.cpp** and in the **water.cpp/h** file, but feel free to create your own files and implement wherever you want.

The scene available in the template code contains a base plane (water) and one cube that can be rotated around its *x*- and its *y*-axis using the W, S, A, D keys. Moreover, an orbit camera with the cube in its center is available. The camera can be controlled by holding the left mouse button; it follows the mouse movements and rotates on a sphere with a fixed radius. You can zoom in and out (*i.e.*, de- and increase the radius of the sphere) with the mouse wheel. In the figure below, an example of the template scene is depicted:



Tasks

1. Setup the hierarchical geometrical model of the boat. For that, remove the scaled cube model from the template code. Then start with **7 cubes** (size: 2x2x2) and apply transformations to them (**scaling, rotation, and translation**) to obtain the **boat parts** representing: **body, bridge, bulwarks for all sides (left, right, front, back), and a mast** (see introduction figure). All the necessary measurements for the various parts can be taken from the drawing below. You are free to design your own boat, but it must include **at least the same number of parts**. In addition, at least two of the three transformations **scaling, rotation,** and **translation** must be used for its creation.



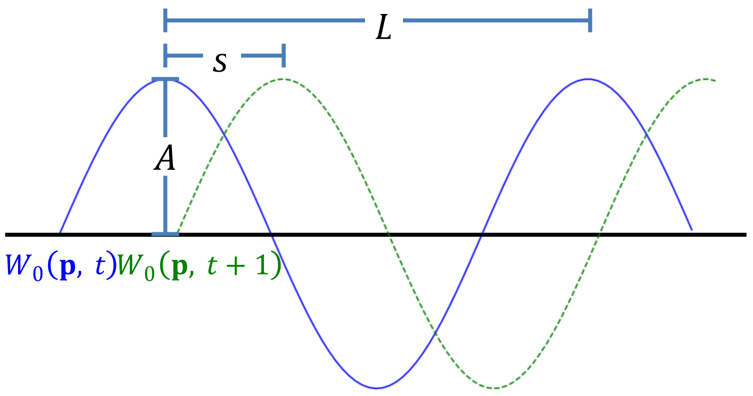
Use **different colors for the different parts** (see example images).

2. Next, animate the water surface over time. In each frame, you'll need to compute the displacement (offset in *y*-axis) of each vertex of the water plane, and update the corresponding OpenGL vertex buffer with the new positions.
   The height of the water surface (at location $\mathbf{p}$) is modeled with a sum of periodic waves:

$$H(\mathbf{p}, t) = \sum W_i(\mathbf{p}, t)\,,$$

each defined with varying amplitude A (wave height), phase φ (speed $s = \frac{\varphi L}{2}$), frequency ω (wavelength $L = \frac{2}{\omega}$), and direction $\mathbf{d}$ **:**

$$W_i(\mathbf{p}, t) = A \sin(\omega(\mathbf{p} \cdot \mathbf{d}) + t\varphi)\,.$$



In the provided video we used three wave functions with the following parameters (see *struct WaterSim* in **water.h**) :

$$A_0 = 0.6, \quad \varphi_0 = 0.5, \quad \omega_0 = 0.25, \quad \boldsymbol{d_0} = \left[\sqrt{\mathbf{2}}, \sqrt{\mathbf{2}}\right]^{\boldsymbol{T}}$$
$$A_1 = 0.7, \quad \varphi_1 = 0.25, \quad \omega_1 = 0.1, \quad \boldsymbol{d_0} = \left[\sqrt{\mathbf{2}}, -\sqrt{\mathbf{2}}\right]^{\boldsymbol{T}}$$
$$A_2 = 0.1, \quad \varphi_0 = 0.9, \quad \omega_0 = 0.9, \quad \boldsymbol{d_0} = [-\mathbf{1}, \mathbf{0}]^{\boldsymbol{T}}$$

Note: By OpenGL convention, the water surface is placed in the *xz*-plane and the *y*-axis represents height.

3. Add **user control** and animate the wave response of the **boat**: It should be possible to steer the boat **forward** and **backward** and to **turn left** and **right**. In addition, the boat should **rotate** and **move up and down** following the wave movement.

   - As a first step, it should be possible to control the boats' **forward** and **backward movements** (*e.g.*, by pressing W and S) and steering **left** and **right** (*e.g.*, by pressing A and D). It should only be possible to turn while moving (*i.e.*, forward, or backward). You can ignore the movement of the waves and only transform the boat in the *xz*-plane.

   - Next, the boat should be **translated** and **rotated** to follow the movement of the water surface.
     The translation along the y-axis is determined by the height of the water surface at the boat's position.
     To determine the rotation of the boat due to the wave movements, you can use the orientation of a triangle placed at the boat's current position.
     That is, define a triangle in the *xz*-plane with its center at the boats position. Calculate the height of each of the triangle corners and use the resulting 3D points to compute a rotation matrix with which the boat can be aligned with the waves.

     Hint: A rotation matrix is a square matrix whose columns and rows are orthonormal vectors. As a result, finding 3 orthonormal vectors that represent the target orientation can be stacked to retrieve the required rotation matrix.

4. Finally, implement a second camera mode, where the camera (and its *lookat* position) should follow the boat (third person camera). Use two different keys (*e.g.*, 1 and 2) to switch between the initial camera mode and your third-person camera. You do not have to reset the focus to the origin when switching back to the initial camera mode (just keep the last boat position as your focus point).

Implementation Remarks

Make sure that your code is clear and readable. Write comments if necessary. Your solution should contain a readme file with the names of the team members, a list of keyboard controls, and any explanation that you think is necessary for the comprehension of the code.

Submission and Grading

Submission of your solution is due on November 27th, 2023 (23:59). **Submit the sources** (*i.e.*, only the content of the *src* folder) in a ZIP archive via OLAT. Do not submit the executable and the content of the *build* folder. Do not submit the external dependencies either. Both folder and archive should be named according to the following convention:

Folder: **CGA1_<lastname1>_<lastname2>_<lastname3>**

Archive: **CGA1_<lastname1>_<lastname2>_<lastname3>.zip**,

with <lastname1>, etc. the family names of the team members. Development in teams of two or three students is requested. Please respect the academic honor code. In total there are 15 marks achievable in this assignment distributed as follows:

- Geometrical models (**5 marks**)
- Boat control and wave animation (**6 marks**)
- Additional camera mode **(2 marks)**
- Code readability, comments, and proper submission: (**2 marks**)

Resources

- Lecture, proseminar slides, assignment description video, and the template code are available via OLAT.
- OpenGL homepage
  http://www.opengl.org
- OpenGL 3.3 reference pages
  https://www.khronos.org/registry/OpenGL/specs/gl/glspec33.core.pdf
- OpenGL tutorial
  https://learnopengl.com/
  http://www.opengl-tutorial.org
- GL framework GLFW
  https://www.glfw.org/documentation.html

Note: Be mindful of employed OpenGL and GLSL versions!