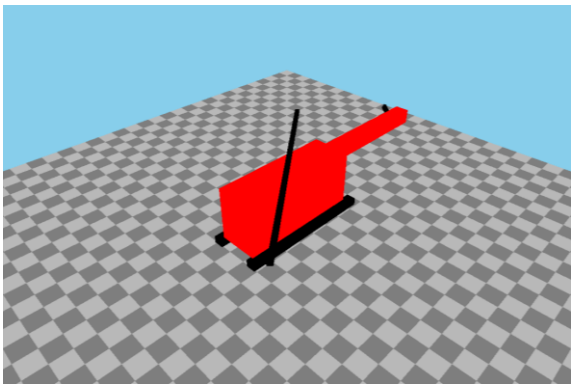
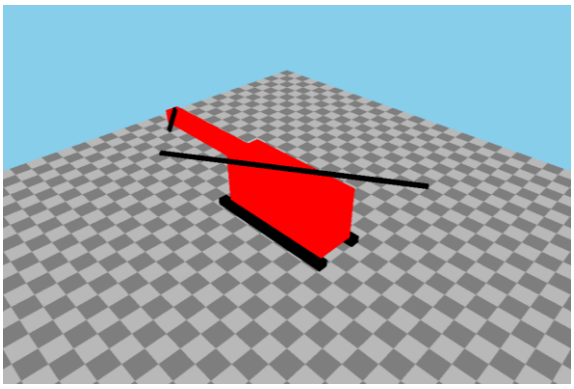
Proseminar Visual Computing
Winter Semester 2022

CG Assignment 1
Hand-out: November 22, 2022
Hand-in: December 5, 2022

Topics

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

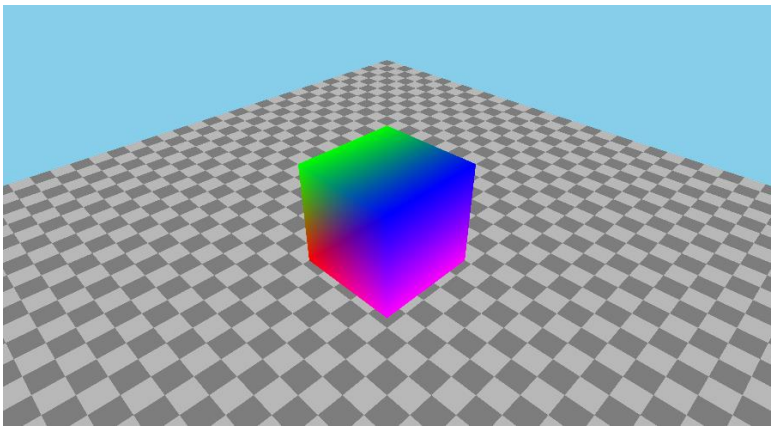
Outline

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

Template code

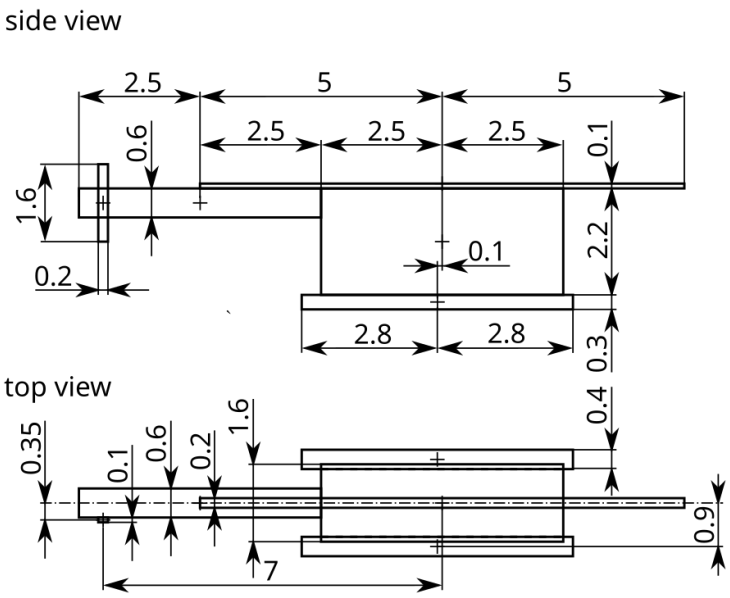
A template code is provided for this assignment. You can modify this code to build your own scene. The functions to modify are all implemented in the main source file **assignment_1.cpp**.

The scene available in the template code contains a base plane 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 using the mouse by pressing the left mouse button. Then it follows the mouse movements and rotates around the cube on a sphere with a fixed radius. To zoom in and out (i.e., de- and increase the radius of the sphere) the mouse wheel is used. In the figure below, an example of the template scene is depicted:



Tasks

1. Setup the hierarchical geometrical model of the helicopter. For that, disable the scaled cube model from the template code. Then start with **6 cubes** (size: 2x2x2) and apply transformations to them (**scaling, rotation, and translation**) to obtain the **helicopter parts** representing: **body, rotor, tail, tail rotor, and left and right slides** (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 helicopter, 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. Add **user control** and **animation** to the helicopter. It should be possible to fly **forward, backward, left, right, up and down**. In addition, it should be possible to **turn the helicopter in place** around its **yaw axis** (see Aircraft principal axes for a description of the principal axes of aircrafts).

    - In a first step, it should be possible to control the helicopters' **forward and backward movement** (*e.g.*, by pressing W and S). As **animation during flying forward/backward**, the helicopter should **tilt forward/backward** along its **pitch axis**.

    - Like forward and backward controls, **flying left and right** should be controllable (*e.g.*, by pressing A and D). For the animation, **tilting** along the helicopter's **roll axis** should be implemented.

    - Flying **up and down** should be controllable too (*e.g.*, using the left Shift and the Space key).

    - Likewise, for **turning** the helicopter around its **yaw axis**. It should be possible to **rotate counter- and clockwise** (*e.g.*, by pressing Q or E).

    - Finally, the main rotor and the side rotor in the back should both be animated to rotate.

3. Finally, implement a second camera mode. The first mode is very similar to the camera available in the template code. But instead of initially looking at the center of the cube, the camera should initially look at the origin of the world coordinate system (i.e., $(0,0,0)^T$). Further, in the second mode the camera (and its lookat position) should follow the helicopter (third person). Use two different keys (*e.g.*, 1 and 2) to switch between the two modes. If switching from mode 2 to mode 1, it is not required to reset the focus to the origin, but just keep the last focus point from mode 2.

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 December 5th, 2022 (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**)
- Helicopter control and animation (**5.5 marks**)
- Additional camera mode **(2.5 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!