INF2110 - Introduction à l'Informatique Graphique **M1 Informatique**
TP5 **Fall 2019**
Prof. Caroline Larboulette

Viewing in 3D

What you need to turn in at the end of the lab session

At the end of the lab session, you should upload a .zip file containing the source code .cpp (+ eventually other classes you have used) of each exercice as well as a .pdf shortly describing what you have done for each question with screenshots to illustrate the results you have obtained. Remember the assignment may be graded.

1. Viewing transformation

To position and orient the camera, use the `glLookAt()` function. Try different parameters.

```
void glLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez,
              GLdouble centerx, GLdouble centery, GLdouble centerz,
              GLdouble upx, GLdouble upy, GLdouble upz);
```

Defines a viewing matrix and multiplies it to the right of the current matrix. The viewpoint is defined by `eye(x,y,z)`. `Center(x,y,z)` is a point on the line of sight. `Up(x,y,z)` is the up-vector. By default, parameters are 0,0,0, 0,0,-100, 0,1,0.

In OpenGL, the viewing transformation is combined with the modeling transformation of the object into the `GL_MODELVIEW` matrix.

2. Projection transformation

To create a projection transformation different from the default one, you need to set up the `GL_PROJECTION` transformation matrix (`glMatrixMode(GL_PROJECTION)`).
For a perspective transformation, you may use `glFrustum()` or `gluPerspective()`. For an orthographic parallel projection, use `glOrtho()`.

- `void glFrustum (GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);`
Creates a matrix for a perspective-view frustum and multiplies the current matrix by it. *`Near`* and *`far`* give the distances from the viewpoint to the near and far clipping plane (must always be positive).
- `void gluPerspective(GLdouble fovy, GLdouble aspect, GLdouble near, GLdouble far);`
Creates a matrix for a symmetric perspective-view frustum and multiplies the current matrix by it. *`Fovy`* is the angle of the field of view in the x-z plane [0,180]. *`Aspect`* is the aspect ratio (width/height).
- `void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);`
Creates matrix for an orthographic parallel viewing volume and multiplies the current matrix by it. Both *`near`* and *`far`* can be positive or negative.

3. Viewing the house

First, write a function to draw the house model of the lecture. You may check it using the *libqglviewer* library and code from previous labs.

Then, with viewing and projection transformations, create a one-points (2-point, 3-points) perspective of the house and parallel projections (see examples seen in class and create new ones). Explain the results you obtain. What happens if you change the up-vector parameters ?

4. Flight Simulator

You want to display the world from the point of view of the pilot of a plane. Its position is at *`x,y,z`* and *`roll`*, *`pitch`* and *`heading`* are the rotation angles of the plane relative to its center of gravity. Use the keyboard to bind the different variables (position, roll, pitch and heading).