Computer Graphics and Visualization
Practices – 2024/2025
Practice Nº 3C – Mouse selection and interaction

Goals:

- Select a 3D model by clicking.
- Interact with a model by using a mouse.

Number of hours to finish the practice: **4 hours**

Assessment: **2 points over 10**

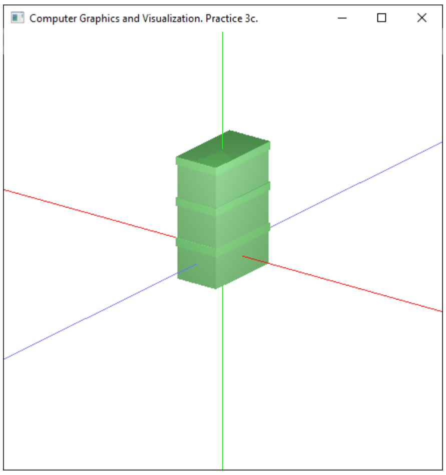
Submission: Upload a zip file **fullname_pr3c.zip with the source code. Files .cpp and .h** of your solution.

PRACTICE Nº 3C

The initial source code is available to develop the practice:

- **pr3c.cpp**: main() function of the program.
- **cgvInterface.h** and **cgvInterface.cpp**: specification and implementation of class *cgvInterface.* It contains the basic functionality to create a display window, to configure it, and to manage the events of the system.
- **cgvScene3D.h** and **cgvScene3D.cpp**: specification and implementation of class *cgvScene3D***.** It includes the basic functionality to render a scene.
- **cgvPoint.h** and **cgvPoint.cpp**: specification and implementation of class *cgvPoint3D*. It includes the functionality to declare points and vectors.
- **cgvCamera.h** and **cgvCamera.cpp**: specification and implementation of class *cgvCamera*. It includes the basic functionality to add and manage cameras.
- **cgvBox.h** and **cgvBox.cpp:** specification and implementation of class *cgvBox*.

After running the program, the first time the result is shown in the following figure:



A) (**1.25 points**) Implement the mouse selection by using the color buffer technique:

- Assign a color as identifier to each box.
- In the specification of `cgvInterface` class:
    - The attribute `cgvInterface::mode` controls whether the rendering pass is for selecting a box (using the color buffers as IDs) (CGV_SELECT) or a common render (CGV_DISPLAY).
    - The attributes `cgvInterface::cursorX` and `cgvInterface::cursorY` store the position of the mouse when it has been clicked.
    - The attribute `cgvInterface::pressed_button` indicates whether the mouse button is pressed or released.
- In the specification of cgvScene class:
    - Add the required code to control the color assigned to each box.
    - Add the required code to control the rotation applied to the selected object and to color it yellow.

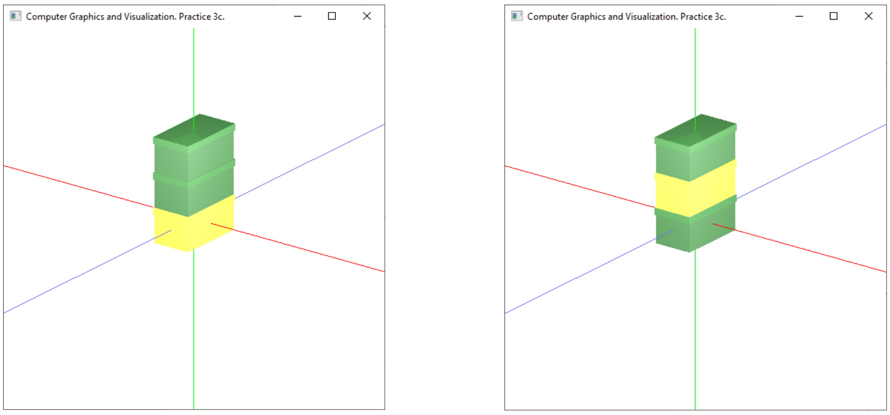
Just after a mouse button is pressed the selection mode enters (mode = CGV_SELECT). The required initial tasks are called from the method `cgvInterface::set_glutDisplayFunc(),` and will be included in the method `cgvInterface::init_selection()`. After rendering the scene, while in selection mode, there are some tasks to finish the selection. These tasks should be inserted in the method `cgvInterface::finish_selection().`

The required code to select a given part of the model consists of:

- Fill the methods `cgvInterface::init_selection()` and `cgvInterface::finish_selection()` following the instructions given by comments in the code. Take into account the attributes included in class cgvInterface.

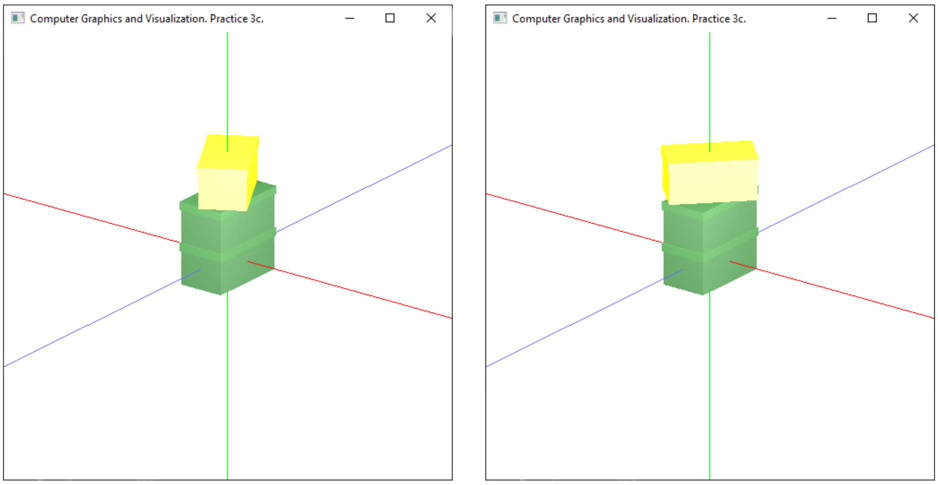
- Following the instructions in the comments, fill the method `cgvInterface::set_glutMouseFunc(GLint x,GLint y)` to change the required attributes to:
    - o Enable the selection mode,
    - o Store the position and the state of the mouse (pressed or released) when clicking the left button and,
    - o Redraw the scene in order to compute the color buffer technique and the selected object.
- Add the required code to class cgvScene3D (attributes and methods) to highlight with a different color (yellow, for example) the box that is selected. This part should return to its original color when the background of the window is clicked.
- **Note:** When in *selection mode* use the function glColor3f instead of glMaterialfv to define the color of the parts of the model.

Next figure shows two examples of the result obtained by clicking the mouse over different boxes:


Computer Graphics and Visualization. Practice 3c.
Computer Graphics and Visualization. Practice 3c.

B) (**0.75 points**) Interact with the selected box by moving the mouse while its left button is pressed. Fill the method `cgvInterface::set_glutMotionFunc(GLint x,GLint y)` to change the model by taking into account the new position of the mouse.

This figure shows a pair of examples after clicking and moving the mouse:


Computer Graphics and Visualization. Practice 3c.
Computer Graphics and Visualization. Practice 3c.