

CSC 471: Program 2 – Due Friday, April 24th at 11:59 pm. The late policy on the syllabus applies. This programming assignment should be done **individually!** You may talk to one another about the program, but you may not look at someone's working code.

Objectives: Learn and apply 3D transformations. Write a program that displays a 3D object in both wire-frame and solid. Allow the user to manipulate the object directly using the mouse. You can do this task in stages. To start, download the example code from the class webpage.

- **Stage 1:** Finish creating the 3D unit cube object. The colors of the vertices are from a color cube going from black (0,0,0) to white (1,1,1) (i.e. one vertex is colored (0,0,0), its neighbors are colored (1,0,0), (0,1,0) & (0, 0, 1) etc.). The cube should be able to be rendered in both wire-frame and solid.
 - Place this cube so that it is centered at the origin of the world frame (i.e (-0.5, -0.5, -0.5) to (0.5, 0.5, 0.5)).
 - Create the XYZ axes of the world frame. Allow the display of these axes to be toggled on/off with the key board key "a".
 - Note we will use orthographic projection with glOrtho.

- **Stage 2:** Add transformations using the mouse and a menu (i.e. glTranslate, glRotate, glScale). Allow the user to select a specific transform from a menu of translate, scale and rotate. The user should be able to apply multiple transforms. **All the transformations are relative to the cube's center!** Note: the world axes stay put and do not get transformed like the cube. This will help give you a frame of reference. Work on translations and scaling first and get these two transforms to work.
 - Use the mouse to determine the amount to translate or scale the object. Click down anywhere in the OpenGL window, move the cube in the direction that the mouse is moving. Translations need only be done for X and Y. You will need to convert viewport (pixel) coordinates from the amount of mouse movement to world coordinates to determine how far to move the cube. Use some reasonable scaling term. Set bounds on the possible translation and scaling amounts so that the object does not fly off the screen. If the cube has previously been translated, and is then rotated, it should be rotated about its new position.
 - Implement a virtual trackball to rotate the cube. The rotation is again with respect to the cube's center. We will talk more about how to do this in class. You must implement this on your own! Only use class notes as a resource.
 - Add a reset key "r" to bring the cube back to the origin of the world with the original size and orientation -- +/- 1/2.

Your program should work (as expected) when the window is resized or moved. Point breakdown (A completely functional program is worth 100 points total.):

20 points for working translate

20 points for working scale

30 points for working rotate

30 points for transforms working together and general program behavior