Course: "Computer Graphics," ECS 175, Fall Quarter 2019
Instructor: Bernd Hamann

Project 2: **"A 3D TRANSFORMATION AND PROJECTION SYSTEM"**

Date due: Monday, October 28, 2019

The second project requires the implementation of **3D translation** (specified by a 3D vector), **3D rotation about an arbitrary axis in 3D space** (specified by two 3D points and a rotation angle in degrees), and **3D scaling** (specified by one scaling factor used in all three directions). Scaling should be done with respect to an object's centroid. All objects are to be transformed with respect to their 3D world coordinates, *i.e.*, when an object is transformed for the first time, you transform its 3D coordinates you read from the input file describing it.

Furthermore, you need to implement **three orthographic projections** (projecting all objects in the scene into the xy-plane, xz-plane, and the yz-plane). You need to make sure that all 3D objects are being displayed on the screen at the same time. You can ensure this by computing a 3D bounding box (Xmin, Xmax, Ymin, Ymax, Zmin, and Zmax being its coordinate extrema) containing all 3D objects after they have been transformed and making sure that the eight vertices of this bounding box are all visible on the screen (discussion in class). There is still plenty of freedom for you, *e.g.*, how to map from world coordinates to final screen/device coordinates. Make use of it, and do not expect us to prescribe all the details. For this project, you can assume that all objects lie within the unit cube, $[0,1] \times [0,1] \times [0,1]$. By making this assumption, you just have to ensure that the unit cube is mapped to a square area on the screen.

You can utilize the OpenGL graphics library calls whenever appropriate. You can also utilize your own graphics macros you have developed in previous projects.

Besides displaying all objects in the form of wire frames (*i.e.*, line drawings) and transforming a set of **predefined 3D polygons/polyhedra**, user menues must be provided to **interactively specify**

- the **name** of the **input file** defining all objects,
- the **ID** of the object to be **transformed**,
- the **ID** of the object whose **vertex coordinates** are to be **shown**,
- the **scaling factor, translation vector**, and
  two points on the rotation axis and the **rotation angle**.

The user should be able to change these parameters easily by providing a screen area used for displaying and manipulating these parameters. To assist the user when specifying a rotation, display the line segment defined by the two points the user specifies as the rotation axis. Make yourself familiar with the portion of that part of the OpenGL graphics library you need for this project. Define all 3D objects to be manipulated and displayed in a data file you read in. Each object is defined in terms of a set of 3D points and a set

of edges defining the connections among the points. The data file is to be stored in this fashion:

1	number of objects
	definition of 1st object:
4	number of points of 1st object
0.0 0.0 0.0	coordinates of 1st point
1.0 0.0 0.0	coordinates of 2nd point
0.0 1.0 0.0	coordinates of 3rd point
0.0 0.0 1.0	coordinates of 4th point
6	number of edges of 1st object
1 2	edge from point 1 to point 2
1 3	edge from point 1 to point 3
1 4	edge from point 1 to point 4
2 3	edge from point 2 to point 3
2 4	edge from point 2 to point 4
3 4	edge from point 3 to point 4

The scene should consist of **at least three different objects**. Make sure that the user can specify **which object is to be manipulated**. Whenever an object has been translated, scaled, or rotated, make sure that you store the latest coordinates of all points so that you can update the data file later on. Whenever a single object has been transformed with respect to its 3D world coordinates, you have to redraw the entire scene.

Finally, the scene (if it has been altered by transformations) should be **written to a data file** replacing the one you have read initially.

Besides having to hand in a program listing, please prepare a "manual sheet" explaining how to use your program.

The overall grade (on a scale from 0 to 100) will depend on i) **completeness** (40%), ii) **correctness** (40%), iii) **interface quality** (15%), and iv) the **manual sheet** (5%). No project will be accepted when it is more than seven (7) days late; for each day, one (1) point will be deduced.

DO NOT REMOVE YOUR PROGRAM! YOU WILL BE ABLE TO USE IT IN THE NEXT ASSIGNMENT(S).

H A V E F U N ! ! !