Homework 3: Transformation and Projection

COSC6372 Interactive Computer Graphics (Spring 2023)

DUE: MARCH 21, 2023 AT 11:59 PM

1. Introduction

In this assignment, you will integrate the transformation and projection to your Gz library. With your provided API functions, the main application will read series of triangles form the text file **`Tris.txt`**.

This assignment requires you to read some additional materials about homogenous coordinate, transformation, and projection. Please check the list in the notes section bellow.

All the data you need for this assignment is put in the zip file. Check the file **`handout.pdf`** for some overview of the assignment. Except some files already provided in assignment 2, there are other files are new and some files have been updated, please check them:

2. Files to update

File	Description	Type
Gz.h	Updated to support viewport, transformation and projection. You need to complete the required functions in file Gz.cpp.	Incomplete
Gz.cpp		
GzMatrix.h	The GzMatrix class supports you to manage and manipulate matrices. Note that you can use the 4x1 matrix to represent 3D vertex with homogenous coordinate. However, you need to complete converter functions toVertex() and fromVertex() to do so.	Incomplete
GzMatrix.cpp		
Tris.txt	An input text file contains the list of triangles. Note that the content of this file has changed, but the format is still the same.	Data files
TeaPot?.bmp	The sample bmp-format results. Note that you are supposed to generate a result looks like this file, but not exact pixel-by-pixel.	

3. Notes

1. Some additional materials you may need:
   http://en.wikipedia.org/wiki/Transformation_matrix
   http://en.wikipedia.org/wiki/Homogeneous_coordinates#Use_in_computer_graphics
   http://en.wikipedia.org/wiki/3D_projection
   http://en.wikipedia.org/wiki/Translation_(geometry)
   http://en.wikipedia.org/wiki/Rotation_(geometry)
   http://en.wikipedia.org/wiki/Scaling_(geometry)
   http://www.opengl.org/sdk/docs/man/xhtml/glViewport.xml

http://www.opengl.org/sdk/docs/man/xhtml/gluLookAt.xml
http://www.opengl.org/sdk/docs/man/xhtml/glTranslate.xml
http://www.opengl.org/sdk/docs/man/xhtml/glRotate.xml
http://www.opengl.org/sdk/docs/man/xhtml/gluPerspective.xml
http://www.opengl.org/sdk/docs/man/xhtml/glOrtho.xml

2. To complete this assignment, you also need to read comments and hints in the provided source code careful.

In this assignment, we mostly follow the conventions of OpenGL (in provided links). However, we have an inconsistent: In assignment 1, we assumed the Z-buffer selects the pixel with largest Z value to render. But, OpenGL implement Z-buffer by using the distance from eye to the pixel as the Z value. This is the description quoted from The red book:

"The depth buffer stores a depth value for each pixel. As described in "Hidden-Surface Removal Survival Kit" depth is usually measured in terms of distance to the eye, so pixels with larger depth-buffer values are overwritten by pixels with smaller values. This is just a useful convention, however, and the depth buffer's behavior can be modified as described in "Depth Test." The depth buffer is sometimes called the z buffer (the z comes from the fact that x and y values measure horizontal and vertical displacement on the screen, and the z value measures distance perpendicular to the screen)."

To make it consist, we have updated the source code in the file **`main.cpp`**, the results it generates and the function **`Gz::multMatrix()`** in **`Gz.cpp`**. The major change is the value of the zNear and zFar. In the previous version, zNear>zFar and zFar<0. This is different to the OpenGL's conventions on projection functions. The new zNear and zFar satisfies 0 < zNear < zFar as the OpenGL's conventions:

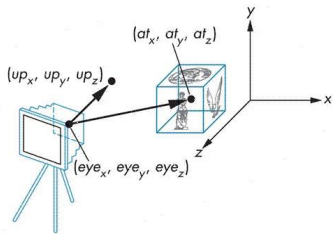
http://www.opengl.org/sdk/docs/man/xhtml/gluPerspective.xml
http://www.opengl.org/sdk/docs/man/xhtml/glOrtho.xml

However, your implementation of the orthographic and perspective projection will not work if you do as described in those links. You need to modify the projection matrix to change the sign of the z values of all vertices. In details, you need to change the signs of 4 numbers in the third row of the projection matrices.

Also there are some attentions you might need to check:

- The fovy angle in Gz:: perspective() and the rotation angle in Gz::rotation() are in degree. But the C++ sin/cos/tan functions work with radian.
- The order of multiplying transformation matrices (We also updated the **`Gz::multMatrix()`** in **`Gz.cpp`**) http://www.opengl.org/sdk/docs/man/xhtml/glMultMatrix.xml - How to combine projection matrix and transformation matrix?

http://www.opengl.org/sdk/docs/man/xhtml/gluProje
ct.xml - How should the Gz::lookAt() work?



4. Requirements

1. Do the assignment independently.
2. You must submit all your source code, project files (MSVC project or Makefile), your results and your report. TA may test your implementation by changing some options, changing the source

   code of the main program or changing the data file.
3. You need to write a detailed report (50% points of the assignment, pdf format), you should state the assignment problem, explain the algorithm or method you use, explain details of

   implementation, discuss your results, etc.
4. Save your results as images.
5. Upload every necessary file to GitHub
6. In your GitHub readme file, put your name and student ID there, including the coding environment and compiling method (command) if necessary.
7. You can only use the libraries we provided.
8. You will lose points if violate any requirement above.