

CS 112 Fall 2008: Introduction to Computer Graphics
Programming Assignment 3
Due Date: Wednesday, Nov 5, 2008

PROJECT GOAL

- a) Write an animation using OpenGL
- b) Verify OpenGL functions with your my_gl functions

WRITE AN ANIMATION USING OPENGL

Write a program to display two cubes - Spinner and Revolver, that satisfy the following constraints:

- a) Spinner should rotate about an axis through its center, parallel to any edge.
- b) Revolver rotates about the same axis as Spinner revolving around Spinner.
- c) Revolver DOES NOT rotate about an axis through its own center.
- d) The frequency of Spinner rotating about its own axis should be the same as that of the rotating Revolver. This means, for every complete revolution of Revolver around Spinner, Spinner should itself have completed 360 degrees.
- e) Spinner should be larger than Revolver.
- f) Spinner and Revolver should NOT be of the same color.

Your program should support interactive viewpoint manipulation, i.e., the user should be able to use the mouse select viewpoint position in space. - Use glLookAt for this.

Hints that MIGHT be useful

- a) Use glTranslate, glRotate, glScale - Look them up in the OpenGL reference
- b) OpenGL uses a local co-ordinate system, and POST-MULTIPLIES successive transformations.
- c) Think about how the mouse motion should affect the image. Look up glLookAt.
- d) Reuse any part of the code from last weeks assignment that you think will be useful.
- e) glutIdleFunc() can be used to trigger a refresh of the image every short period of time, and also update the animation. Read more in the URL:
<<http://www.opengl.org/resources/libraries/glut/spec3/node63.html>>

VERIFY OPENGL FUNCTIONS WITH YOUR MY_GL FUNCTIONS

- a) You have to verify your my_gl functions using your “Spinner” assignment.
- b) You should follow the following rules to verify.
 - i) Whenever you use glLoadIdentity() for a MODEL_VIEW matrix, initialize your matrix to a 4x4 identity matrix.
 - ii) Whenever you use glPushMatrix/glPopMatrix on your MODEL_VIEW stack, use glGetMatrix(...) to get the top matrix in the current MODEL_VIEW matrix stack, into *newmatrix* after the glPush/glPop operation.
 - iii) Replace every call to any OpenGL transformation function with the following sequence of calls. The following example is given for glTranslate. Perform similar changes to other calls you have implemented.
Replace every glTranslate(tx,ty,tz) with:

```
// This operates the OpenGL way, retrieves the result and undoes it
glPushMatrix();
glTranslated(tx, ty, tz);
glGetMatrix(tempmatrix);
glPopMatrix();
```

```
// This uses you're my_gl* functions, and will have effect
my_glTranslated(tx, ty, tz);
my_glGetMatrix(newmatrix);
printMat(newmatrix);
printMat(tempmatrix);
if( compareMat(newmatrix, tempmatrix) ) printf("Matrix is wrong");
// compareMat raises the flag if the matrices don't match.
```

- iv) For all this to work properly, you must make sure to delete the list of #define statements you created for programming assignment #2, so that no automatic substitutions are performed and you control which version (gl or my_gl) of the transformations is called in each moment.
- v) Notice that some my_gl functions not used in the previous assignment will be used now. This might cause problems if they were not correctly implemented before.

The above change is going to print a **lot** of matrices. Once you have verified your *compareMat* function correctly compares two matrices, you can remove the printMat statements. It does not need to be part of your submission.

Once you have verified that there is no statement displayed that “*Matrix is wrong*” (in other words, your matrix in *newmatrix* and OpenGL’s matrix in *tempmatrix* are the same all the time), then you should replace OpenGL’s computation of transformations with your equivalent computation of the transformations as follows. For example, a *glTranslated(tx,ty,tz)* should be substituted by *my_glTranslated(tx, ty, tz)*.

Check if your output is the same as before (before substituting gl* with my_gl*). If so, then you have successfully emulated OpenGL transformation calls using your own my_gl transformation calls.

WHAT AND HOW TO SUBMIT? (THE STANDARD PROCEDURE)

- a. Go to the EEE dropbox for the 3rd programming assignment.
- b. Create a folder named after your UCInet-ID (i.e. jstudent, for Joe Student).
- c. Put all your source files (modified or not) in that directory. By source files I mean any C++ code; both headers (*.h) and *.cpp files.
- d. No executables, intermediate files, project files or any similar stuff is required.
- e. Also, upload a file named README.TXT (capital letters, please) briefly telling us what you did, what bugs are known to be present, relevant difficulties you found and any other comment you want to make. Of course, extra features should be explained here as well.