Assignment 2: Transformations & Viewing
CSL307 – Computer Graphics

Due Date: 7th October 2013

Part 1: Implement 3D Transformations using homogenous coordinates

- Create a vector class (**Vector4**)
  - Addition, Subtraction, Homogenize, Relation Operators
  - Additional methods as required
- Create a 4x4 matrix class (**Matrix4**)
  - Addition, Subtraction, Multiplication (by scalar, vector, & matrix), Transpose, Inverse
  - Additional methods as required
- Affine transformations: **Translate**, **Scale**, **RotateX, RotateY & RotateZ**
- Arbitrary Rotation function that rotates around any axis at any point in space (**Rotate**)
- Viewing transformation similar to the OpenGL gluLookAt function (**LookAt**)

Part 2: Mesh Rendering

A triangle mesh is a type of polygon mesh in computer graphics. It comprises a set of triangles (typically in three dimensions) that are connected by their common edges or corners. We will only look at mesh models with an easy-to-use indexed face set format (IFS). The basic object in this mesh model format is a vertex, a point in 3D space with the origin at (0,0,0). An object in IFS format is formed by a set of triangle faces, which is defined by 3 non-identical vertices.

You are provided with starter code that parses an input IFS file and stores triangle meshes in a data structure for you. You will need to create a class to read the vertices of the triangles from this data structure, render the object in 3D perspective view using OpenGL in wireframe format and perform required transformations on the object. Explore lighting options in OpenGL such that the depth of object surface is clearly seen.

A 3D model of teapot in IFS format is included with starter code. For more models, see the Brown Mesh Set library. Your program should handle any of the IFS mesh models from this library.

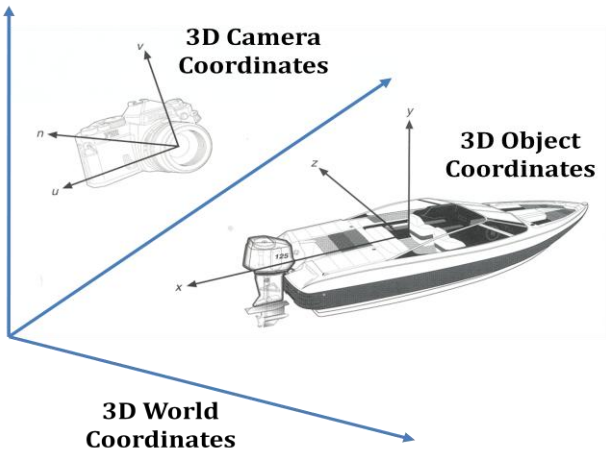
Part 3: Build a simple program for viewing 3D objects

Once you have implemented the transformation and rendering functions, allow the user to manipulate and view the object as desired.

Remember we discussed different types of coordinate systems in the class:

- World Coordinate System
- Camera Coordinate System
- Object Coordinate System

Using the transformations defined above, the program should interactively make it possible to:



- Move the observer/camera along different directions of the world coordinate system
- Set the position (move) & orientation (rotate & tilt) of the camera around view coordinate system
- Rotating the object around its own axes & scale the size of the object

You can either create a simple graphical user interface or use the keyboard for input. For example, use the keys x/X, y/Y, z/Z to change the position of the camera along the world's coordinate axes, i/I, j/J, k/K to rotate the camera, keys u/U, v/V, w/W to rotate the object, a/A to move the camera forward/backward, and +/- to scale the object.

• Submit

1. Complete source code as a tar-gzipped archive.
    a. Include only source code but **NO** executables.
    b. When unzipped, it should create a directory with your ID. Example: **P2008CS1001** (case sensitive!!!)
2. It should include a **makefile** for compiling your code. It should compile without any errors and produce the executable for testing.
    a. ***Negative marks for any problems/errors***
    b. Executable should be called “viewer”
    c. Command to test your program with input: ***viewer teapot.ifs***
3. The code must be reasonably commented and written in an easily understandable manner.
    a. ***Negative marks for illegible code!!***
4. Include a README file to convey any details.
5. Submit/Upload it to moodle.

• Grading: 100

1. Helpers: Vector & Matrix (5)
2. Affine Transformations: Scaling (5), Rotation X/Y/Z (5), Translation (5)
3. Other Transformations: Arbitrary Rotation (10), glLookAt Equivalent (10)
4. Mesh Rendering (10), Lighting (5), Transformation (5)
5. Manipulation: World Coordinates (10), Camera Coordinates (10), Object (10)
6. Graphical User Interface(10)
7. Total = 100
8. Late Submission: 20% Penalty per extra day
9. **Can discuss ideas, but STRICTLY no copying/sharing or source code.**