Assignment 5: Texture mapping Deadline: **October 19, 2010 at 23:00**

In this assignment, you will add the texture mapping functions to the Gz library. Note that we only consider the simple texture mapping without shading. So you do not need to use the illumination in the previous assignment.

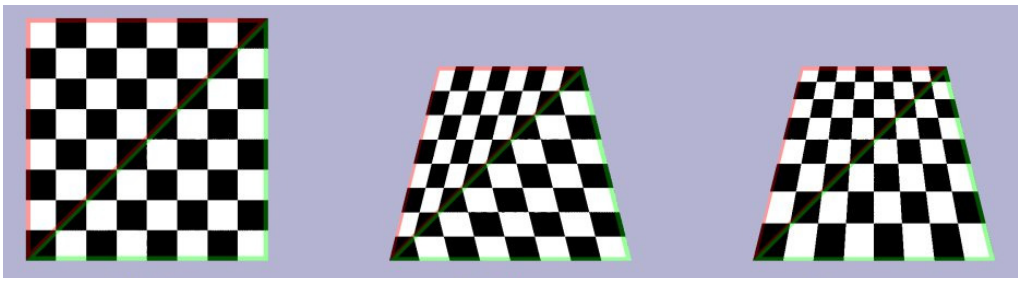
With your provided API functions, the main application will read series of triangles form the text file **`Tris.txt`**, load a texture from the image **`Texture.bmp`**, and draw the tea pot. This time, **`Tris.txt`** only provides you the coordinates of the vertices and the corresponding texture coordinates.

All the data you need for this assignment is put in the zip file **`hw5.zip`**. There are some files have been updated, please check them:

| File | Description |
|---|---|
| Gz.h | Updated to support texture mapping. Note that we do not support texture mapping with lights. You need to add some code to these files. |
| Gz.cpp | |
| GzFrameBuffer.h | Updated with the solution of HW4. You may put your rasterization with texture for HW5 in these files. |
| GzFrameBuffer.cpp | |
| Tris.txt | An input text file contains the list of triangles. You can figure out the format by reading the source code in file main.cpp or check the description bellow. |
| Texture.bmp | The texture file. In main.cpp, we use the class GzImage to load this texture. |
| TeaPot1.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. |
| TeaPot2.bmp | |

Here are some details you may need to pay attention:

1. Since we do not use the lights, you can directly apply the color of the texture pixels for rendering. The only thing you need to figure out is for each rendered pixel, what is the corresponding one in the texture.
2. Your program need to support both orthogonal projection and perspective projection. With the orthogonal projection, you can do the linear interpolation for texture coordinates as you did in previous assignments. However, for the perspective projection, you need to do the perspective correction to avoid the artifact likes following:



Texture Artifact Correct

Please read the follow link about how to do the perspective correction:

http://en.wikipedia.org/wiki/Texture_mapping#Perspective_correctness

3. The format of the **`Tris.txt`** has changed. After the first 3 numbers represent for vertex coordinate, we have 2 numbers u and v represent for texture coordinates. The ranges of u and v are [0, 1]. (u, v)=0 corresponds to pixel (0, 0) of the texture, (u, v)=(1, 1) corresponds to pixel (width-1, height-1) of the texture.

(0, 0) (1, 0)

(0, 1) (1, 1)