Programming Assignment 4
Texture Mapping in OpenGL
ICS 112, Fall 2016
Due: 11:55PM, Nov 30, 2016

In this assignment, we are going to apply texture mapping in OpenGL.

For theoretical explanation of texture mapping, please refer to our lecture slides, or
https://www.cs.uic.edu/~jbell/CourseNotes/ComputerGraphics/TextureMapping.html
http://www.movesinstitute.org/~mcdowell/mv4202/notes/lect18.pdf

For texture mapping in OpenGL, please read:
http://www.glprogramming.com/red/chapter09.html
http://www.gamedev.net/page/resources/_/technical/opengl/opengl-texture-mapping-an-introduction-r947

Task: Implement the 12 TODOs in the skeleton code. We have provided detailed comment for each TODO, and Each TODO can be implemented with 1~5 lines of code.

In main.cpp:

```
void initTexture()
{
  makeCheckImage();

  // TODO: 1. generate texture using glGenTextures

  // TODO: 2. bind the texture, using glBindTexture

  // TODO: 3. specify texture parameters, using glTexParameteri

  // TODO: 4. specify image data to the texture, using glTexImage2D
}
```

In OpenGL applications, textures are usually read from texture/image file, or procedurally generated. In this assignment, we have created a checkerboard texture pattern for you, stored in array checkImage. In “initTexture”, you will bind it to OpenGL texture.

In sceneModule.cpp

```
void drawScene(){
    // lighting
    glMaterialfv(GL_FRONT_AND_BACK, GL_AMBIENT, ambient);
    glMaterialfv(GL_FRONT_AND_BACK, GL_DIFFUSE, diffuse);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SPECULAR, specular);
    glMaterialfv(GL_FRONT_AND_BACK, GL_SHININESS, shininess);
    glColor3fv(diffuse);

    // TODO 5. enable texture 2D using glEnable

    // TODO 6. set texture environment parameters using glTexEnvf
    // recommended parameters are
    //    target: GL_TEXTURE_ENV
    //    pname : GL_TEXTURE_ENV_MODE
    //    param : GL_MODULATE
    // after you finish your project, try replace GL_MODULATE with GL_REPLACE

    // TODO 7. create a quad using gluNewQuadric

    // TODO 8. enable quadric texture generation using gluQuadricTexture

    // TODO 9. draw a sphere using gluSphere
    //    radius = 1
    //    slices = 20
    //    stacks = 20

    // TODO 10. disable texture using GLDisable
}
```

In drawScene, you will draw a sphere using gluSphere, and the texture you binded before will be applied on the sphere.
To make the texture mapping properly work, the uv coordinate in texture space, for each vertex, should be specified. It is a challenging task to generate such coordinates without visible artifacts. Fortunately, gluQuadricTexture will do this for you. What you need to do is to simply call this function.
In real world applications, the uv coordinates are mostly specified by the designer who creates the texture for the surface.

In inputModule.cpp:

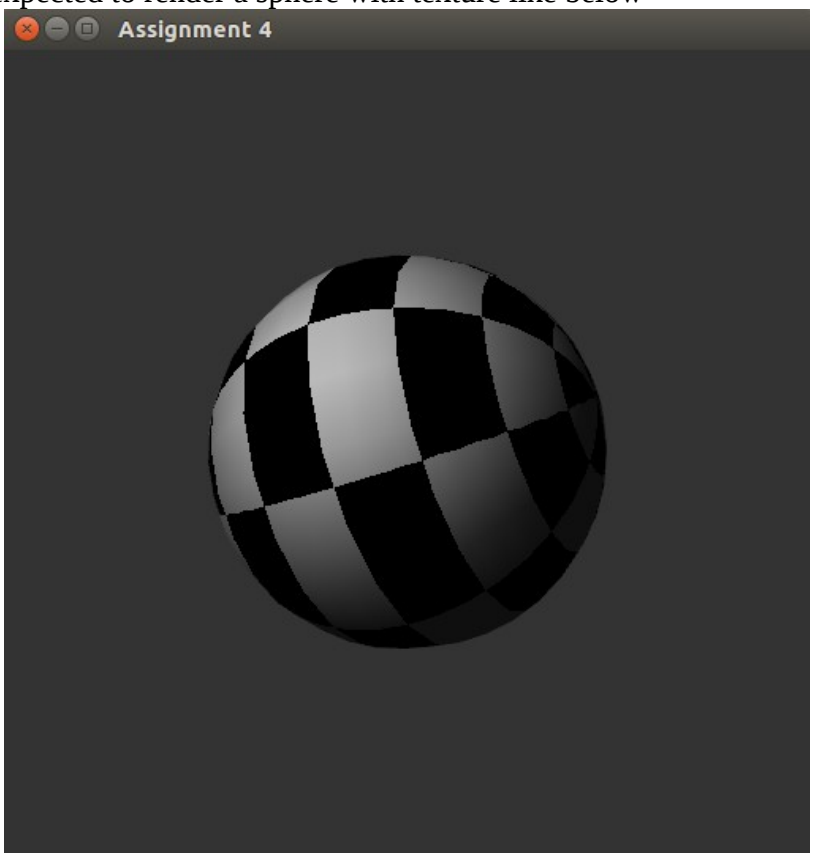
```
case 'a':
  antialising = !antialising;

  if (antialising) {
    // TODO 11: enable multisample
  } else {
    // TODO 12: disable multisample
  }
  break;
```

You may enable mutlisample to improve the image quality. For what it does, please refer to https://en.wikipedia.org/wiki/Multisample_anti-aliasing. The anti-aliasing result may be not as good as

you expected, it is normal, anti-aliasing is a hard task after all.

Expected result:
1. Your program is expected to render a sphere with texture like below



2. You need to have an understanding of what happens in the process of texture mapping, and you are encouraged to try different parameter setting in each function. For example, what will happen if you use GL_REPLACE rather than GL_MODULATE in glTexEnv? Such questions may be asked in demo sessions.

Troubleshooting:
1. If you have compiling error, for example, "*glut.h cannot be found*". Try modify

```
#include <GL/glut.h>
to
#include <GLUT/glut.h>   // if you are on Mac, this most likely works
or
#include <glut.h>      // if you are on Windows, this most likely works
```

2. If you have questions, please ask on EEE message board, instead of sending us E-mails. Your question has a higher chance to be answered on message board, and more students will benefit from the discussion too.