Mae 152 Fall '03
UCSD Department of Mechanical and Aerospace Engineering
Alex Simpkins
Fri Nov. 28, 2003

Basic texture mapping with OpenGL

This is a **very basic** tutorial on making an image, then applying it to a surface as a texture map. For more information, read chapter 9 of the red book (for example, for environment mapping and more detail). This is a basic survival kit only, and does not consider all possible techniques.

There are a few steps to do in order to make this a success.

1) **Set up the program**.

Texture map loading code which was included with the official GLUT examples is included on the webpage handout section (*texture.c* and *texture.h*).

a) You must create a file of type .rgb, which is in some image editing programs referred to as SGI format. It MUST be dimensioned (width and height) by a power of two. For example, you could have an image that is 512x256, or 256x256, (width and height don't need to be the same) etc. There is also a minimum and maximum (varies across some systems and graphics cards, but >64x64 pixels is safe – generally you do not need larger than 512x512, and normally 256x256 will suffice well).You can save as SGI format as well (.sgi – basically the same file format). BE SURE THIS FILE IS IN THE SAME FOLDER AS YOUR PROJECT!

b) If there are options for compression of the image, use RGB Verbatim. Otherwise your program may not be able to load the image.

c) you will need a few variables:

```
GLubyte *texData;
int texWidth, texHeight;
GLuint texId1;
char *texFilename1 = "texmap.rgb"
```

d) Be sure to include the file texture.c with your project in visual c++, then you'll need to type into your code at the library includes:

```
#ifdef WIN32
   extern "C" {
   #include  "texture.h"
   }
#endif

#ifndef WIN32
   #include "texture.h"
#endif
```

We have to do this in order to make the code work properly on the windows operating system as well as other systems.

This will allow you to access the texture image loading functions from the handout code. Realize that a 'file format' is just a method of storing information. OpenGL has a particular format for its data, which may not be convenient to create the textures in (Adobe Photoshop does not have any 'Save As OpenGL texture image' format options). This code will allow you to create a texture in SGI format, as explained above, and just load it into an OpenGL compatible format. Cool!

One handy shareware file conversion utility is called graphic converter. It can save as SGI format and can open most image file formats. If you are running Linux you can use GIMP, which is free.

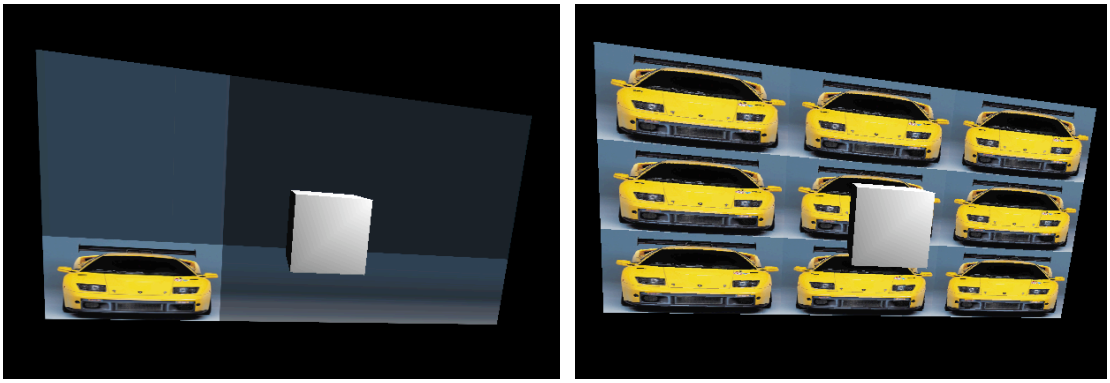
2) <u>Load the texture image into a pointer variable (*texData).</u>

You can create a function to encapsulate loading the texture data, then setting the appropriate options. You can also put that into an initialization function which is called when your program is first loaded. The following are the basic commands to generate a texture, select which texture will be linked to that texId# (a unique name for that texture and its options), load the image data into our texData pointer, then create the texture with glTexImage2D (note that there is a texImage1D, and a texImage3D).

```
glGenTextures(1,&texId1);
glBindTexture(GL_TEXTURE_2D, texId1);
imgLoad(texFilename1, 0, 0, &texWidth, &texHeight, &texData);
glTexImage2D(GL_TEXTURE_2D, 0, 4, texWidth, texHeight, 0, GL_RGBA,
                                        GL_UNSIGNED_BYTE, texData );
```

You will most likely want to set various options for how the texture looks, what happens at its edges, etc. You would do that with glTexParameterf(..) commands.

GL_CLAMP… GL_WRAP…



Several options have been set for you in the example code. These mostly deal with how your texture will perform when the S, T values are greater than 1. This repeat option is handy for surfaces such as wood flooring, brick walls, etc.

3) Draw your object, and bind the texture to it!

You should at this point consider that each point in your texture image is called a *texel* named after 'texture element' rather than a pixel. Pixels are the points on your display, while texels, since they are stretched could be any size per point in the image. The texture has coordinates in S, T instead of X, Y (we do this to avoid confusing them together). As you draw your object, you apply your texture starting at the lower left corner, then moving in counter clockwise direction as you draw your object (a GL_QUADS, for example).

You also need to create a normal if you plan to use lighting. This is done with the glNormal3f() command. You can imagine applying a texture in this way as applying a sticker to a surface, tacked at four corners. The 'tacks' are placed by a glTexCoord2f(S,T) command. NOTE that this is a two dimensional command in S, T coordinates, and that ***S and T go from 0 to 1***. This means that 0,0 is the lower left of your texture image, and 1,1 is the upper right corner of your texture image.

```
glEnable(GL_TEXTURE_2D);

  glBegin(GL_QUADS);
    glNormal3f(0.,0.,1.);
    glTexCoord2f(0.,0.);
    glVertex3f(-20.,-10.,-10.);

    glTexCoord2f(1.,0.);
    glVertex3f(20.,-10.,-10.);

    glTexCoord2f(1.,1.);
    glVertex3f(20.,10.,-10.);
```

```
    glTexCoord2f(0.,1.);
    glVertex3f(-20.,10.,-10.);
  glEnd();

glDisable(GL_TEXTURE_2D);
```

4) **When you've written all your code, compile it, and you should see a texture mapped QUAD.**

Please see the included handout code, texture_mapping.zip which includes *texture.c* and *texture.h* as well. In that code I used the glutIdleFunc() to animate the quad rotating. I also included lighting, but not much else (in order to reduce confusion).

5) **Troubleshooting common problems:**

Problem	**Possible Solution**
Code compiles, but no window pops up, or it crashes	Be sure image file is in same directory as your program and has the same name your code asks for, and it is of type SGI (*.rgb) if you use texture.c
I get errors for all my calls to the texture.c functions (ie imgLoad(..))	It is likely the texture.c file is not correctly added to your project, or there is a problem with **#include "texture.h"**
No texture is displayed, but my scene works otherwise	Be sure to turn on texture mapping by a glEnable call in your display code or init code
Everything is texture mapped, and I only want one thing mapped	You can turn on and off texture mapping, or switch between textures on the fly in your display code (see glBindTexture(..) in the red book for more info on switching- basically you make this call to switch texture objects before drawing the object you want to map the texture to). Turn off texture mapping with a glDisable(..) call
I fixed all above, and my texture still won't appear, or my program won't run, but it compiles fine.	Make sure your texture width and height are a power of two in their dimension. The width and height can be different, but must must MUST be a power of two (ie 512x512 pixel original SGI image)
This is too easy	a) You haven't tried this yet. b) You need to try the more advanced

	texture mapping techniques, such as environment matching. c) You copied and pasted all the code from the web, and you probably don't understand it entirely d) This really is too easy.