GLSL Texturing

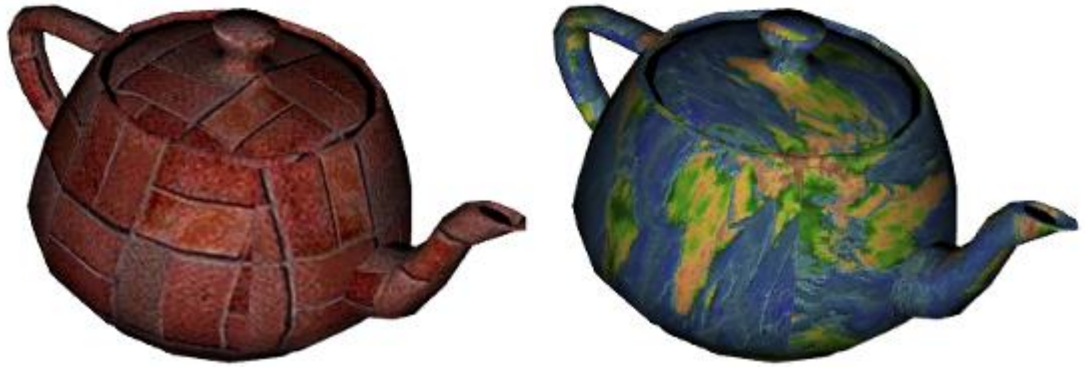
KOREA UNIVERSITY

KUCG

Texturing

What is Texture Mapping?

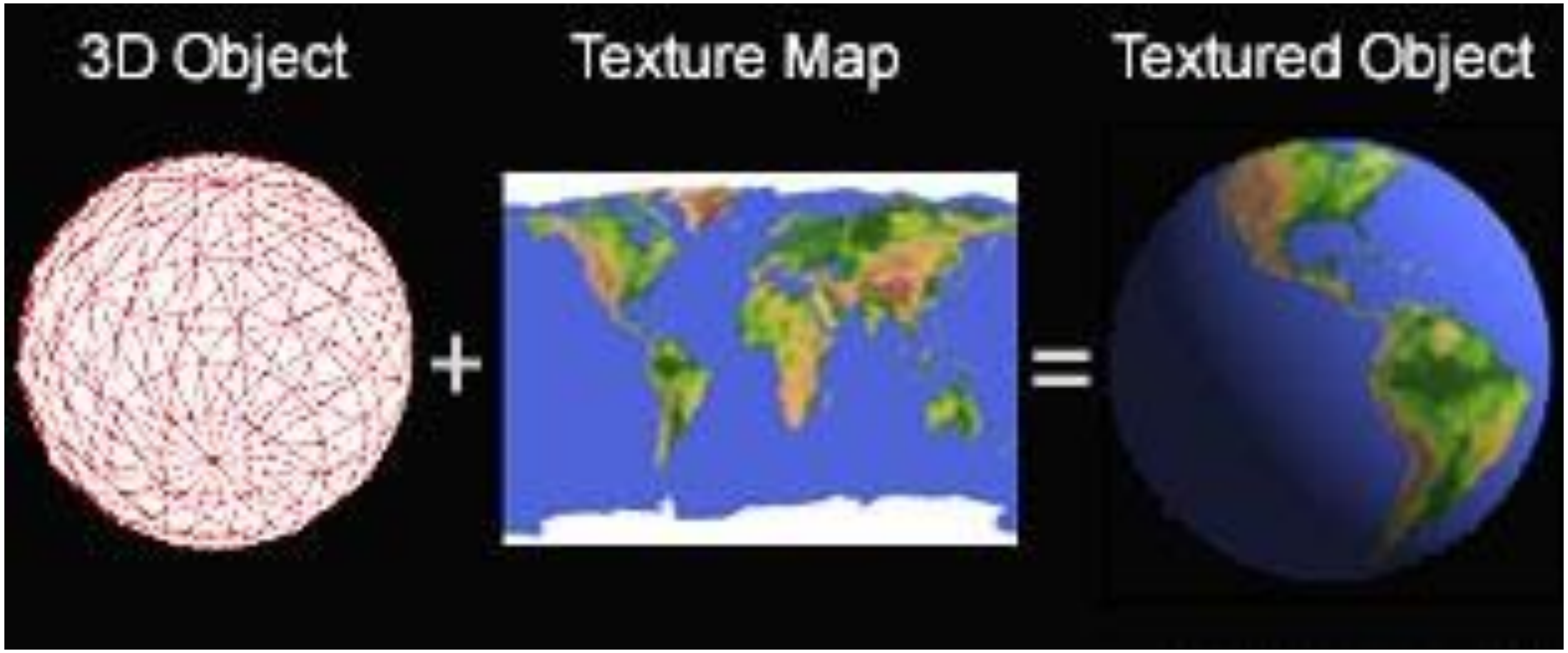
Without Texture Mapping



With Texture Mapping

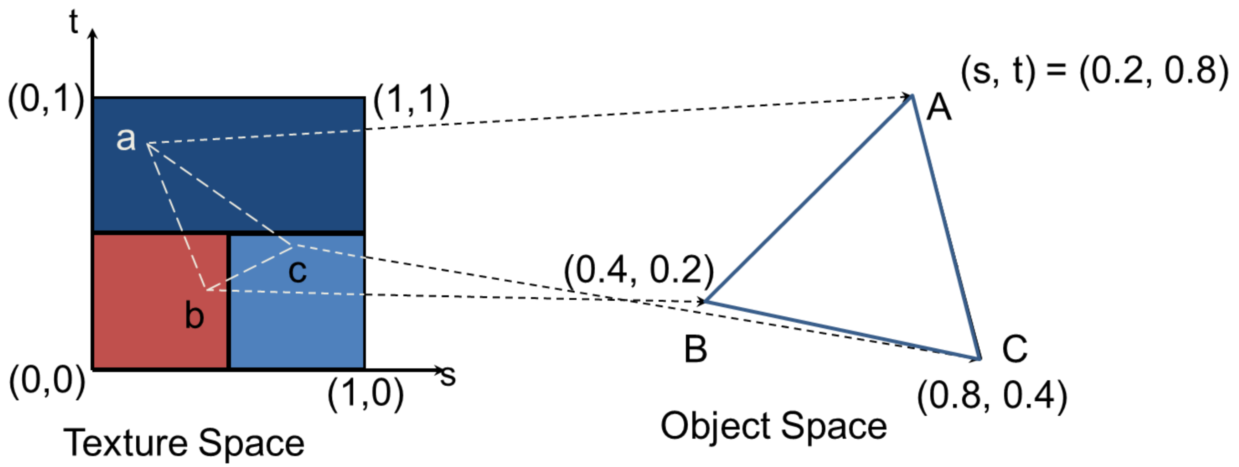
What is Texture Mapping?

- Texture mapping is the process of determining the color of each pixel based on the texture the object is bearing.



What are Texture Coordinates?

```
glBegin(GL_POLYGON);
   glColor3f(r0, g0, b0);
   glNormal3f(u0, v0, w0);
   glTexCoord2f(s0, t0);
   glVertex3f(x0, y0, z0);
   glColor3f(r1, g1, b1);
   glNormal3f(u1, v1, w1);
   glTexCoord2f(s1, t1);
   glVertex3f(x1, y1, z1);
   ...
glEnd();
```



Initialize Texture Mapping

```
void init() {
    unsigned char bitmap[DIMX*DIMY*3];
    ...
    GLuint texID;
    glEnable(GL_TEXTURE_2D);
    glGenTextures(1, &texID);
    glBindTexture(GL_TEXTURE_2D, texID);
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, DIMX, DIMY, 0,
                      GL_RGB, GL_UNSIGNED_BYTE, bitmap);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP);
    // glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, DIMX, DIMY, 0,
    //               GL_RGB, GL_UNSIGNED_BYTE, bitmap);
}
```

Image Data

Texture ID

Enable texturing

Generate texture

Bind generated texture

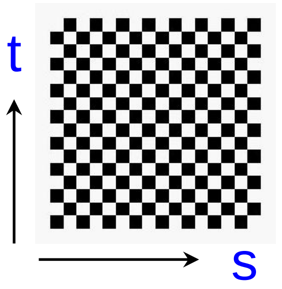
Give the image data for this texture

Specify paramete

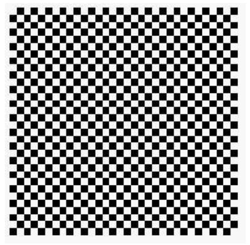
Specifying Texture Parameters
1. Wrapping Mode

- **Clamping:**
  - Use 1 if s,t > 1. Use 0 if s,t < 0.
- **Repeating**
  - Use s,t modulo 1
    - fmod(s ,1.0f), s - floor(s)
- **Usage:**

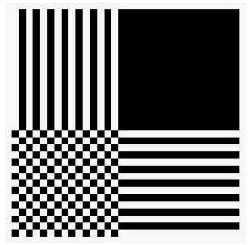
```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
```



texture



With GL_REPEAT

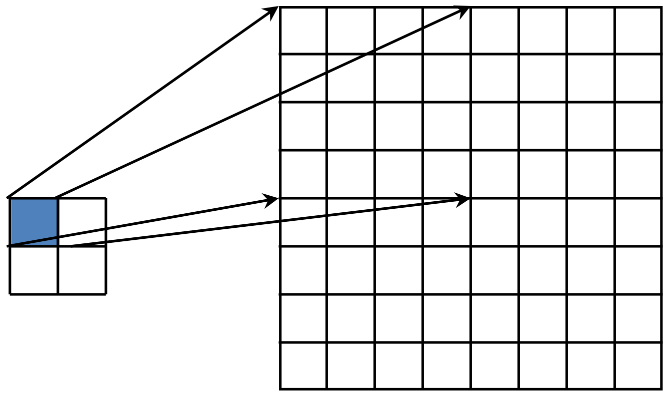


With GL_CLAMP

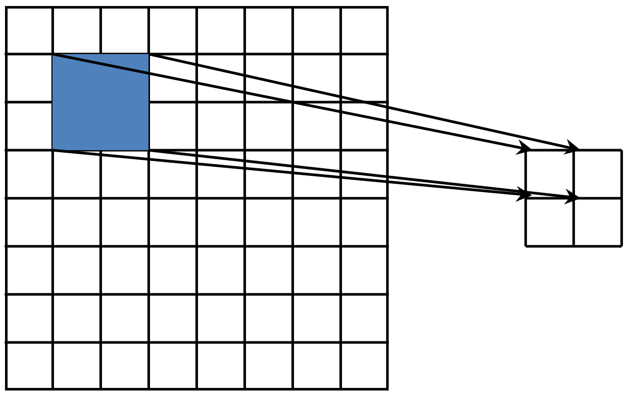
Specifying Texture Parameters

2. Filter Modes

- **More than one texel can cover a pixel (*minification*) or more than one pixel can cover a texel (*magnification*)**
  - Magnification can result in “mosaicking”.
- **Can use point sampling (nearest texel) or linear filtering (2 x 2 filter) to obtain texture values**



Magnification



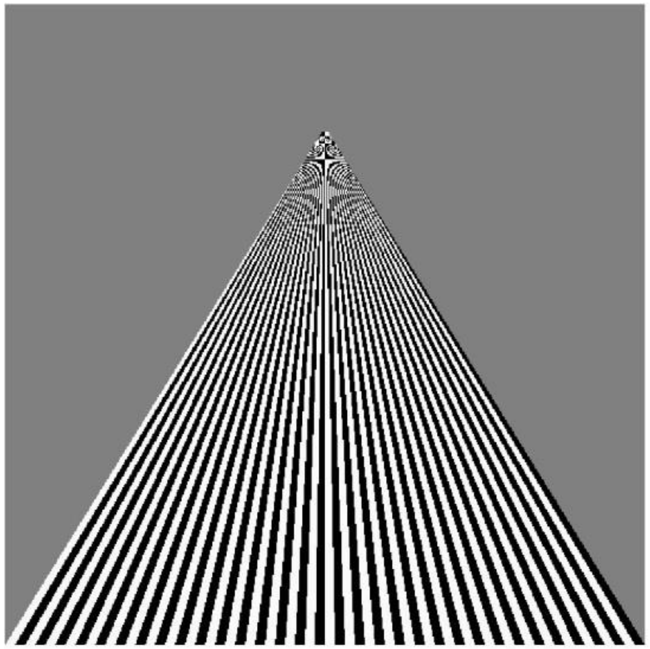
Minification

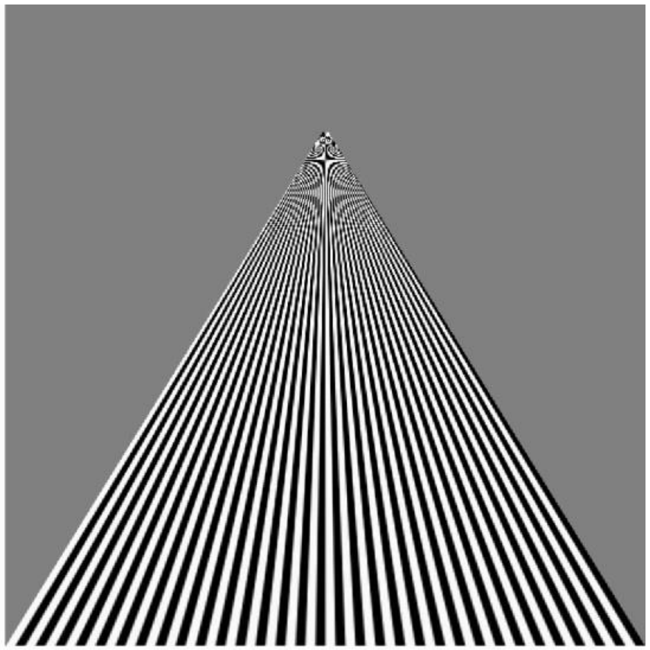
Specifying Texture Parameters

2. Filter Modes: Texture_Parameters

- Filter modes can be set by
  - `glTexParameteri( target, type, mode )`
- Usage:

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```

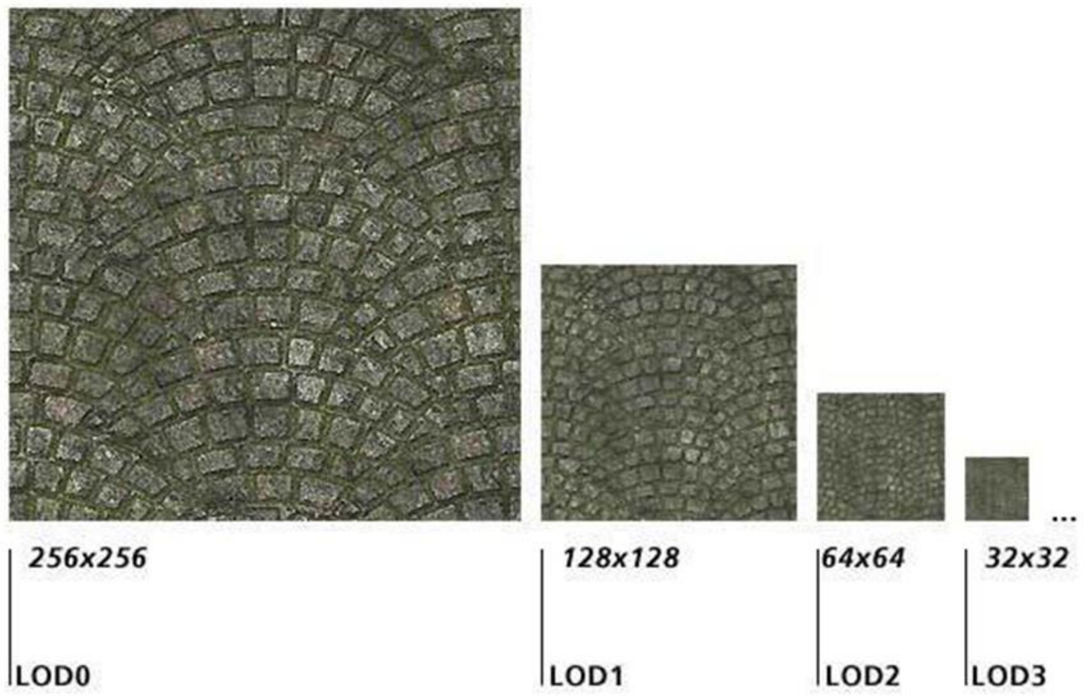

With GL_NEAREST


With GL_LINEAR

Specifying Texture Parameters

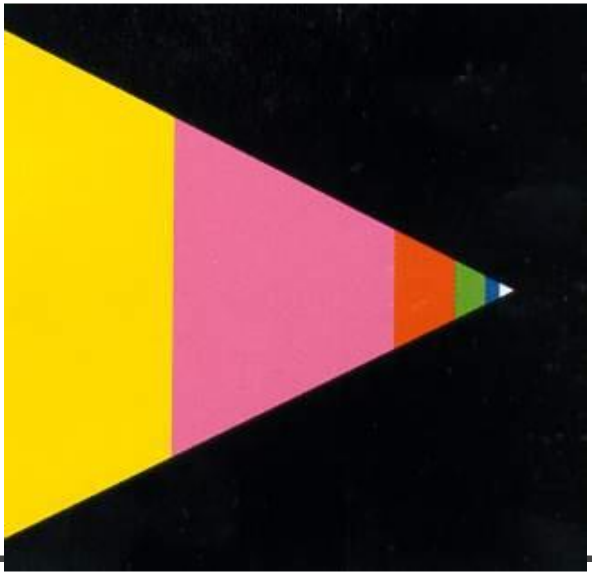
3. Mipmapping

- **Mipmapping allows for *prefiltered* texture maps of decreasing resolutions**
  - Lessens interpolation errors for smaller textured objects



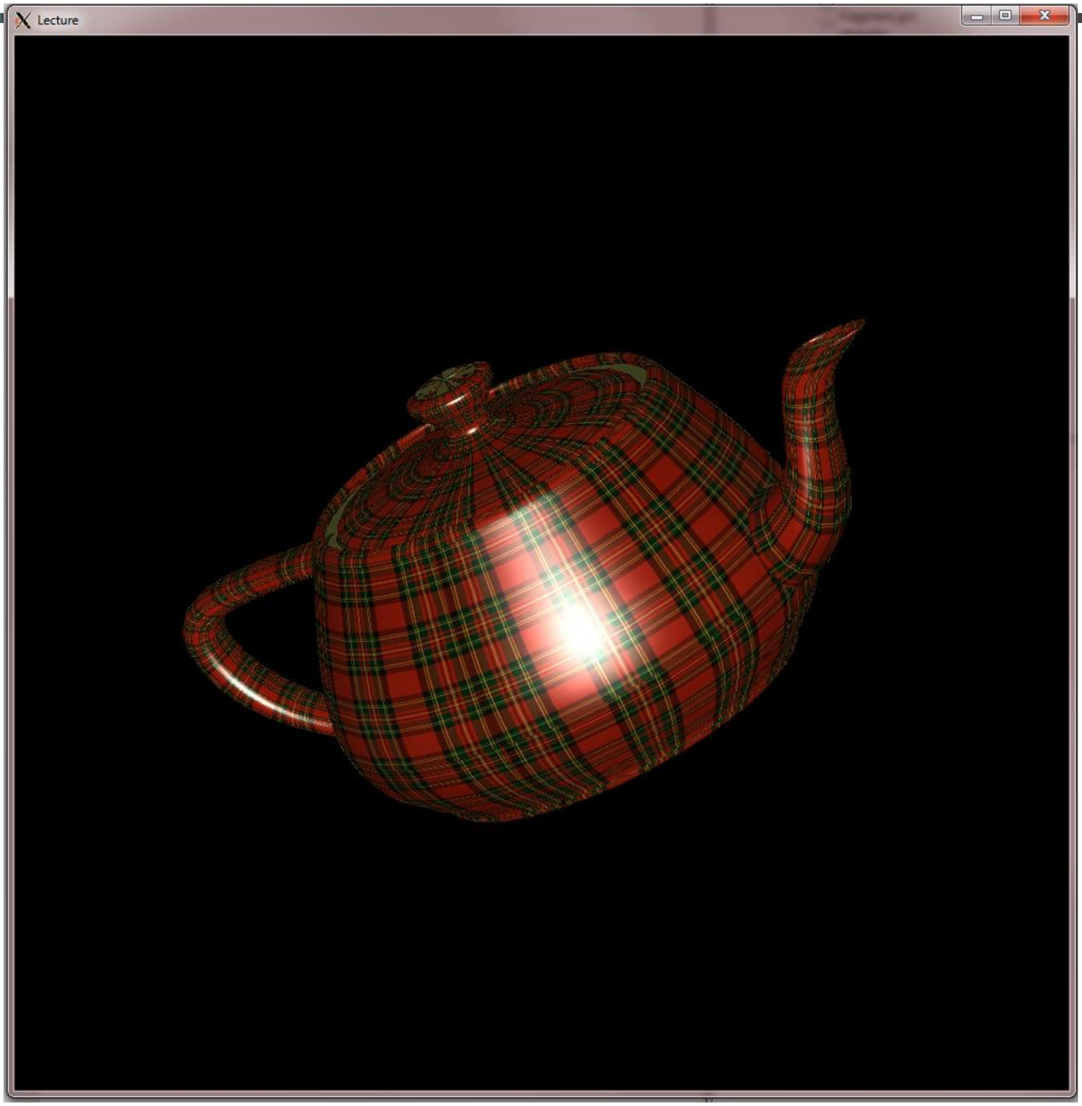
Specifying Texture Parameters
3. Mipmapping Sample Code

```
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGBA, 32, 32, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage32);
glTexImage2D(GL_TEXTURE_2D, 1, GL_RGBA, 16, 16, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage16);
glTexImage2D(GL_TEXTURE_2D, 2, GL_RGBA,  8,  8, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage8);
glTexImage2D(GL_TEXTURE_2D, 3, GL_RGBA,  4,  4, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage4);
glTexImage2D(GL_TEXTURE_2D, 4, GL_RGBA,  2,  2, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage2);
glTexImage2D(GL_TEXTURE_2D, 5, GL_RGBA,  1,  1, 0, GL_RGBA,
   GL_UNSIGNED_BYTE, mipmapImage1);
  ...
glBegin(GL_QUADS);
    glTexCoord2f(0,0); glVertex3f(-2,-1,0);
    glTexCoord2f(0,8); glVertex3f(-2, 1,0);
    glTexCoord2f(8,8); glVertex3f(2000, 1,-6000);
    glTexCoord2f(8,0); glVertex3f(2000,-1,-6000);
glEnd(); glFlush();
```



Texture Practice

Draw Teapot with Texture



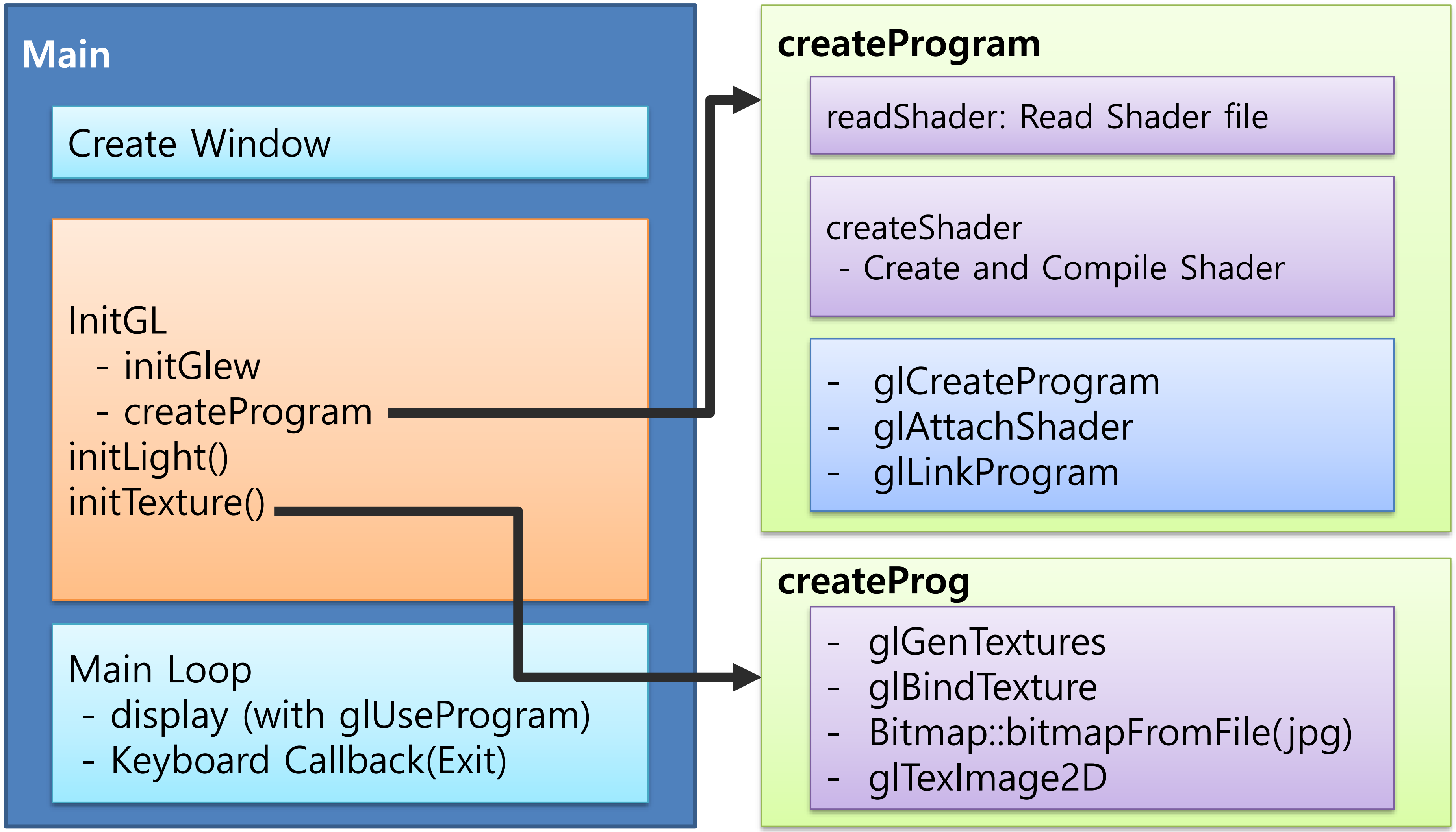
Texture Mapping Coding Exercise

- **Copy Sample Skeleton Code**
  - vglconnect ID@163.152.20.246
  - cp –r /home/share/Texture ./
  - cd Texture

- **Notepad: Shader code 수정**

- **Compile program**
  - make
  - vglrun ./EXE

Program Flow



Program structure

```
#include "XWindow.h"
#include <stdio.h>
#include <stdlib.h>
#include <string>
//function declaration//
//Global variables//

int main(int argc, char *argv[]) {
     //Window Initialization//
     initGL();
     initLight();
     initTexture(); //Initialize Texture


     while(1) {
          Display();
          KeyboardCallback();
           }
      }
}
```

initGL()

```
void initGL()
{
        glewInit(); //glew Initialize Function;
        createProgram(); //Create Shader Program
}
```

Run-time compilation w/ shader source file

initLight(): Light Initialize

```
void initLight(){
        /*Set Light and Material Properties with Array*/
        /*Set light properties*/
        glLightfv(GL_LIGHT0, GL_AMBIENT, lightKa);
        glLightfv(GL_LIGHT0, GL_DIFFUSE, lightKd);
        glLightfv(GL_LIGHT0, GL_SPECULAR, lightKs);
        glLightfv(GL_LIGHT0, GL_POSITION, lightPos);
        /*Set material properties*/
        glMaterialfv(GL_FRONT, GL_AMBIENT, matKa);
        glMaterialfv(GL_FRONT, GL_DIFFUSE, matKd);
        glMaterialfv(GL_FRONT, GL_SPECULAR, matKs);
        glMaterialfv(GL_FRONT, GL_SHININESS, &matShininess);
         /*Enable Light*/
        glEnable(GL_LIGHTING);
        glEnable(GL_LIGHT0);
        glEnable(GL_DEPTH_TEST);
}
```

initTexture(): Initialize Texture

```
void initTexture (){
        glActiveTexture(GL_TEXTURE0); //Activating Texture 0
        glGenTextures(1, &textureID);//Generating Texture
        glBindTexture(GL_TEXTURE_2D, textureID);//Binding Texture
        //Add texture to Back side
        Bitmap bmp = Bitmap::bitmapFromFile("textures/texture.jpg");
        glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, bmp.width(), bmp.height(), 0, GL_RGB,
                        GL_UNSIGNED_BYTE, bmp.pixelBuffer());

        //Set Filter and Wrapping
        glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
        glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
        glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_CLAMP);
        glTexParameterf(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_CLAMP);
        //transfer Texture index to glsl
        glUniform1i(glGetUniformLocation(program, "tex"),0);
        glBindTexture(GL_TEXTURE_2D, 0);
}
```

Display function@Main

```
void display(){
        glClearColor(1.0f, 1.0f, 1.0f, 1.0f);
        glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
        glUseProgram(program);
        glMatrixMode (GL_PROJECTION);
        glLoadIdentity();
        glBindTexture(GL_TEXTURE_2D, textureID);
        glOrtho (-1.0, 1.0, -1.0, 1.0, -10.0, 10.0);
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        glRotatef(40.0, 1.0, -1.0, 1.0);
        glShadeModel(GL_SMOOTH);
        glutSolidTeapot(0.5f);
        glUseProgram(0);
        glBindTexture(GL_TEXTURE_2D, 0);
        glXSwapBuffers(dpy, win);
}
```

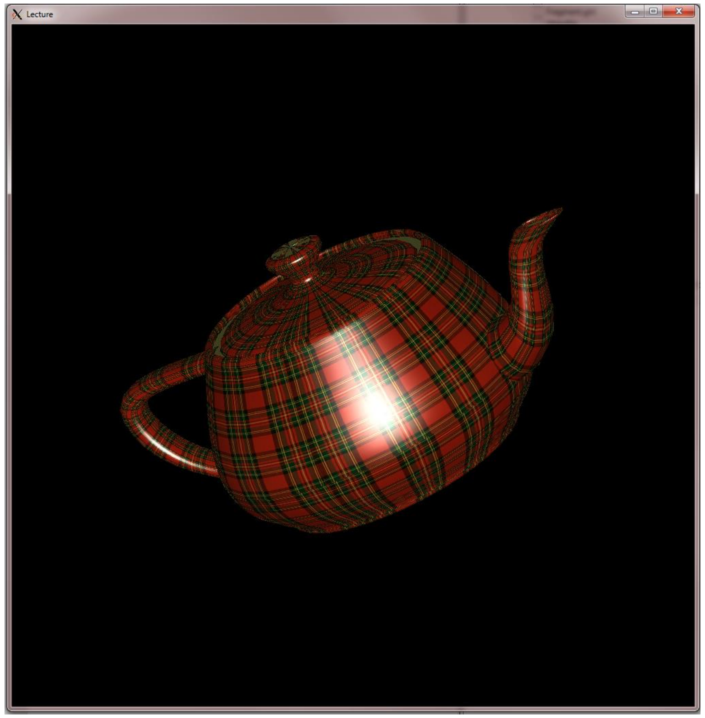
Shader code: Vertex

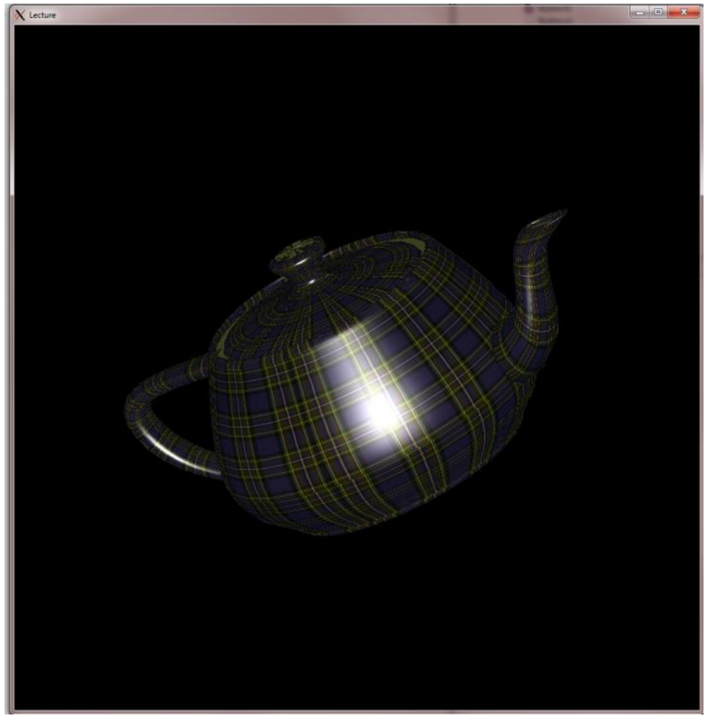
```
#version 130
varying vec3 normal, lightDir, halfVector;
void main() {
            normal = normalize(gl_NormalMatrix*gl_Normal);
    /*  vertex normal to fragment shader  */
            lightDir = normalize(gl_LightSource[0].position.xyz);
    /*  Light Direction Vector to Fragment shader  */
            halfVector = normalize(gl_LightSource[0].halfVector.xyz);
    /*  half Vector to Fragment shader  */
            gl_Position = gl_ModelViewProjectionMatrix*gl_Vertex;
    /*  Projected Position to Fragment shader     */
            gl_TexCoord[0] = gl_MultiTexCoord0;
    /*  texture coordinate to Fragment Shader     */

}
```

Shader code: Fragment

```
#version 130
varying vec3 normal, lightDir, halfVector;
uniform sampler2D tex; //set 2D Texture@Fragment Shader
void main() {
        vec3 n, h;
        float NdotL, NdotH;
        vec4 color = gl_FrontMaterial.ambient * gl_LightSource[0].ambient +
                gl_FrontMaterial.ambient * gl_LightModel.ambient;
        n = normalize(normal);
        NdotL = max(dot(n,lightDir),0.0);
        if (NdotL > 0.0) {
                color += gl_FrontMaterial.diffuse * gl_LightSource[0].diffuse * NdotL;
                h = normalize(halfVector);
                NdotH = max(dot(n,h),0.0);
                color = color*texture2D(tex,gl_TexCoord[0].st);//load Texture Color and compute with Light
                color += gl_FrontMaterial.specular * gl_LightSource[0].specular *
            pow(NdotH, gl_FrontMaterial.shininess);
        }
        gl_FragColor = color;

}
```

Draw Teapot with Texture (Color Channel Change)





(R, G, B)

(B, G, R)

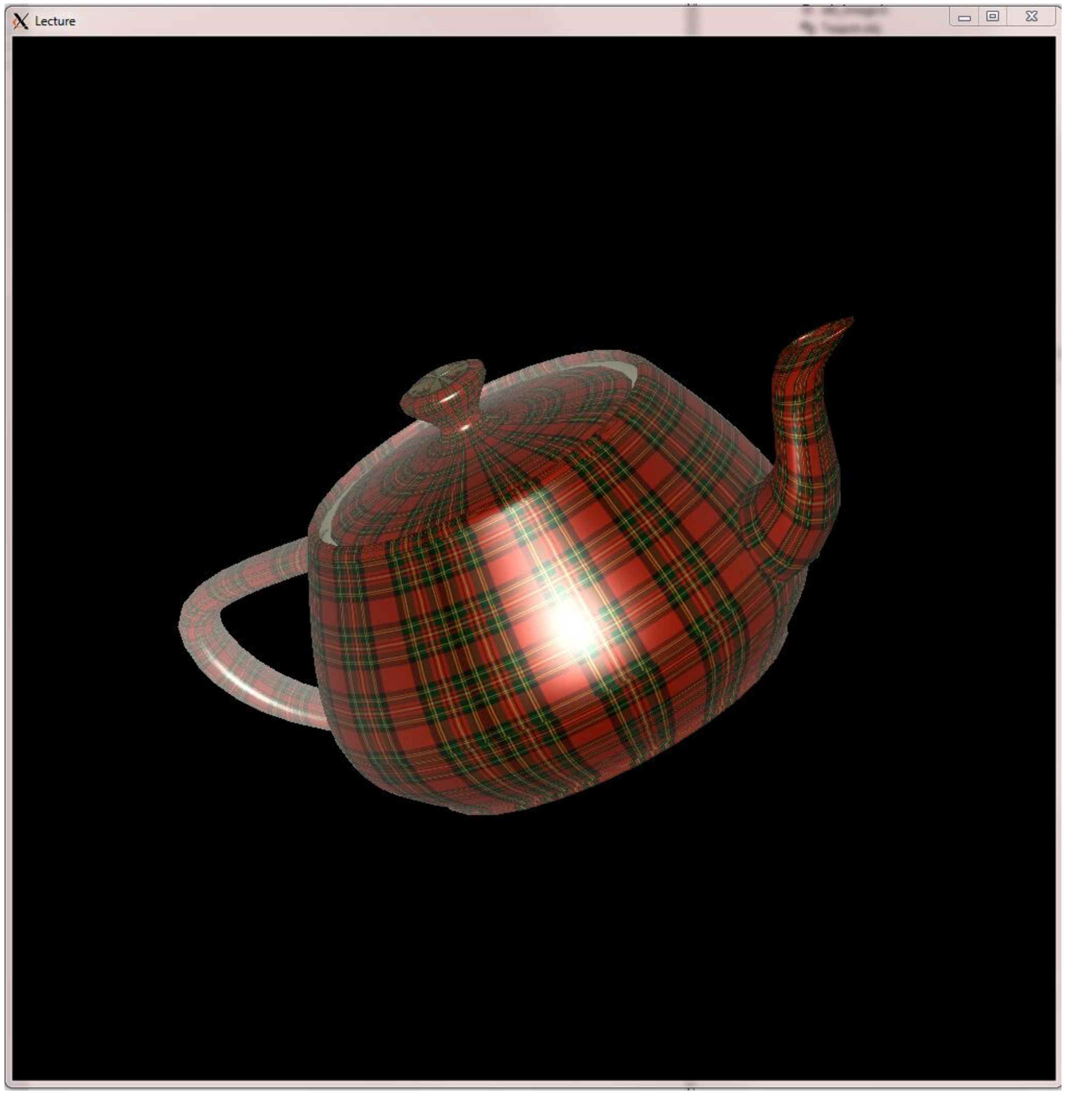
Color Channel Change Coding Exercise

- **Notepad: Shader code 수정**
- **Run program**
  - vglrun ./EXE

Shader code: Fragment

```
#version 130
varying vec3 normal, lightDir, halfVector;
uniform sampler2D tex; //set 2D Texture@Fragment Shader
void main() {
        vec3 n, h;
        float NdotL, NdotH;
        vec4 color = gl_FrontMaterial.ambient * gl_LightSource[0].ambient +
                gl_FrontMaterial.ambient * gl_LightModel.ambient;
        n = normalize(normal);
        NdotL = max(dot(n,lightDir),0.0);
        if (NdotL > 0.0) {
                color += gl_FrontMaterial.diffuse * gl_LightSource[0].diffuse * NdotL;
                h = normalize(halfVector);
                NdotH = max(dot(n,h),0.0);
                color = color*texture2D(tex,gl_TexCoord[0].st).bgra;
                //load Texture Color and compute with Light
                color += gl_FrontMaterial.specular * gl_LightSource[0].specular *
            pow(NdotH, gl_FrontMaterial.shininess);
        }
        gl_FragColor = color;

}
```

Draw Teapot with Foggy

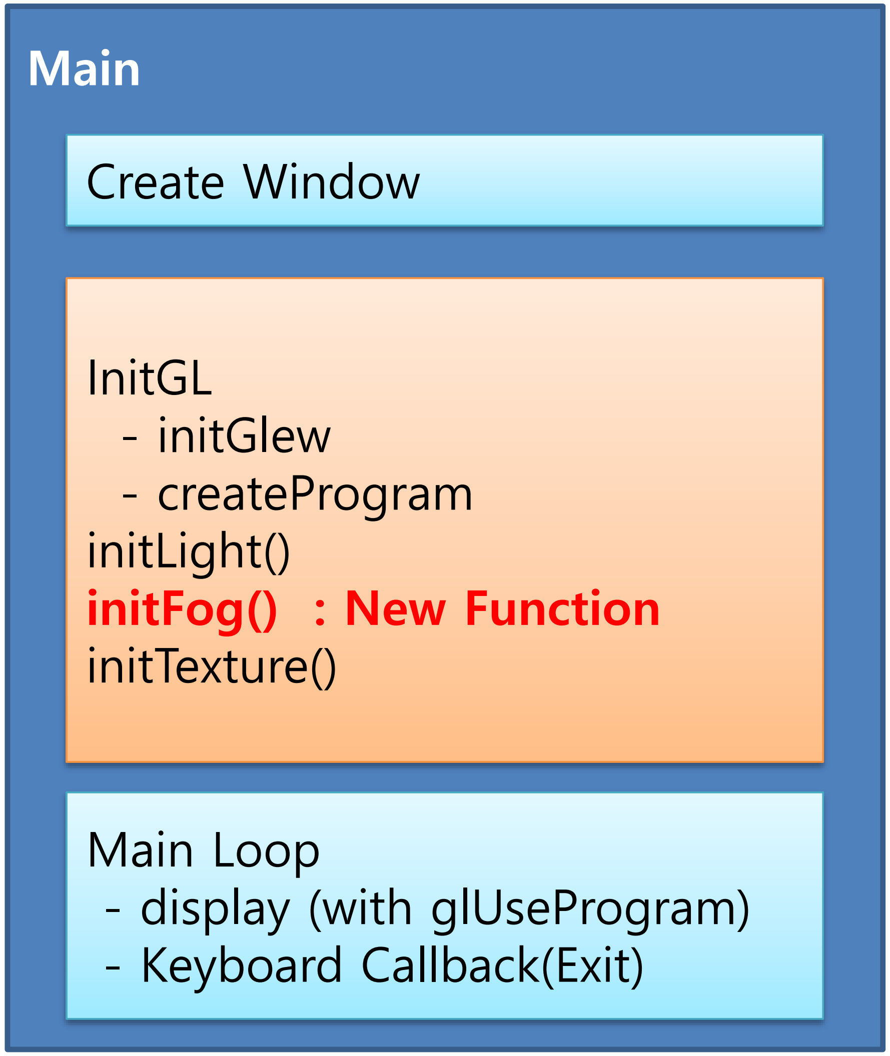


Teapot with Fog Coding Exercise

- **Notepad: Shader code 수정**

- **Compile program**
  - vglrun ./EXE

Program Flow


Main
Create Window
InitGL
- initGlew
- createProgram
initLight()
initFog() : New Function
initTexture()
Main Loop
- display (with glUseProgram)
- Keyboard Callback(Exit)

initFog(): Fog Initialize

```
void initFog(){
        float fog_color[] = {1.0, 1.0, 1.0, 1.0};
        glEnable(GL_FOG);
        glFogfv(GL_FOG_COLOR, fog_color);
        glFogf(GL_FOG_START, 0.48f);
        glFogf(GL_FOG_END, 0.55f);
}
```

Shader code: Vertex

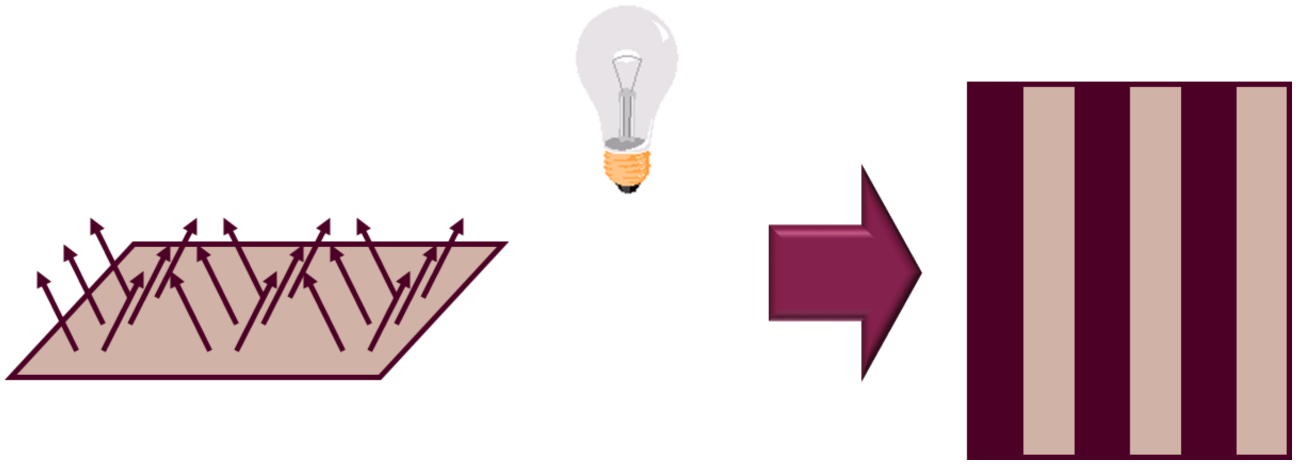
```
#version 130
varying vec3 normal, lightDir, halfVector;
void main() {
            normal = normalize(gl_NormalMatrix*gl_Normal);
     /*  vertex normal to fragment shader  */
            lightDir = normalize(gl_LightSource[0].position.xyz);
     /*  Light Direction Vector to Fragment shader  */
            halfVector = normalize(gl_LightSource[0].halfVector.xyz);
     /*  half Vector to Fragment shader  */
            gl_Position = gl_ModelViewProjectionMatrix*gl_Vertex;
     /*  Projected Position to Fragment shader     */
            gl_TexCoord[0] = gl_MultiTexCoord0;
     /*  texture coordinate to Fragment Shader     */
            gl_ClipVertex = gl_ModelViewMatrix*gl_Vertex;
     /*  vertex position on Clipping Space to Fragment Shader     */
}
```

Shader code: Fragment

```
#version 130
varying vec3 normal, lightDir, halfVector;
uniform sampler2D tex; //set 2D Texture@Fragment Shader
void main() {
            vec4 color;
            /*Compute Light*/
            color = color*texture2D(tex,gl_TexCoord[0].st);

            float z = gl_FragCoord.z / gl_FragCoord.w;                      /*Compute depth*/
            float fogFactor = (gl_Fog.end - z) / (gl_Fog.end - gl_Fog.start);  /*Compute fogfactor*/
            fogFactor = clamp(fogFactor, 0.0, 1.0);

            gl_FragColor = mix(gl_Fog.color, color, fogFactor);              /*mix color with fog*/
             /*mix color with fog: gl_Fog.colr*(1-fogFactor)+color*fogFactor */
}
```

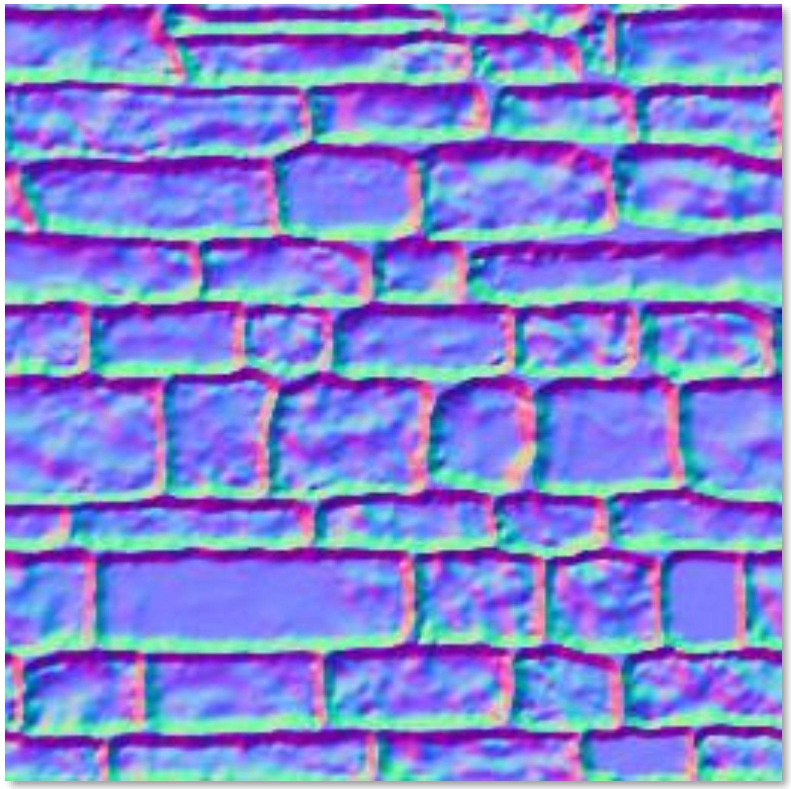
Normal Mapping

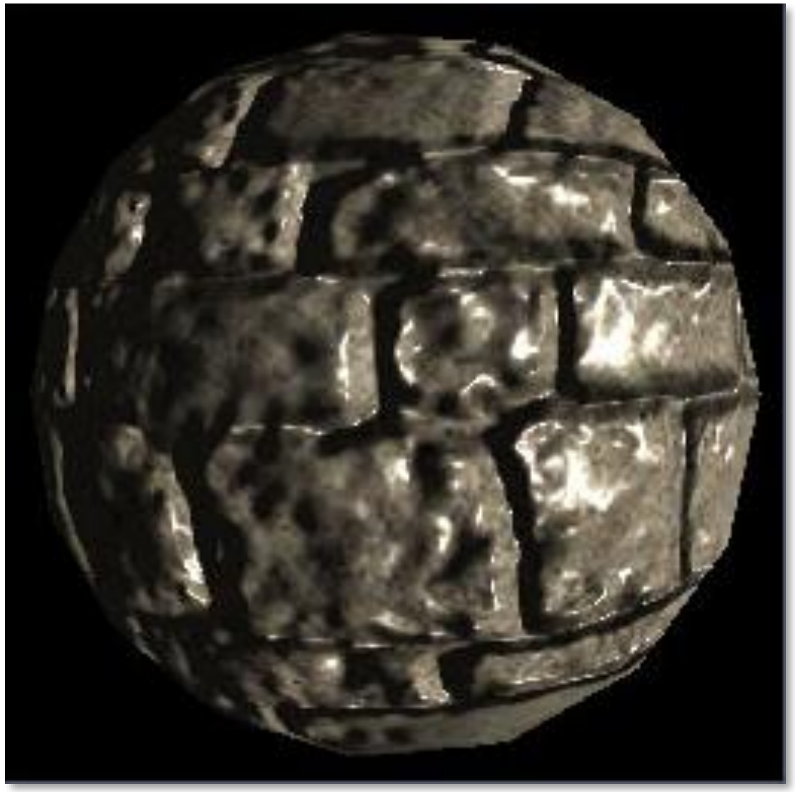
- **Perturbs surface normal vectors**
  - So called 'normal mapping'



- **Normal Calculation : normal = (2*color)-1 *// on each component***
  - normal calculated from normal map in fragment shader
- **TBN Transformation**
  - Computing Tangent Bitangent at Shaders
- **Color Calculation based on Lighting Model**

Normal Mapping Example


[Normal Map]

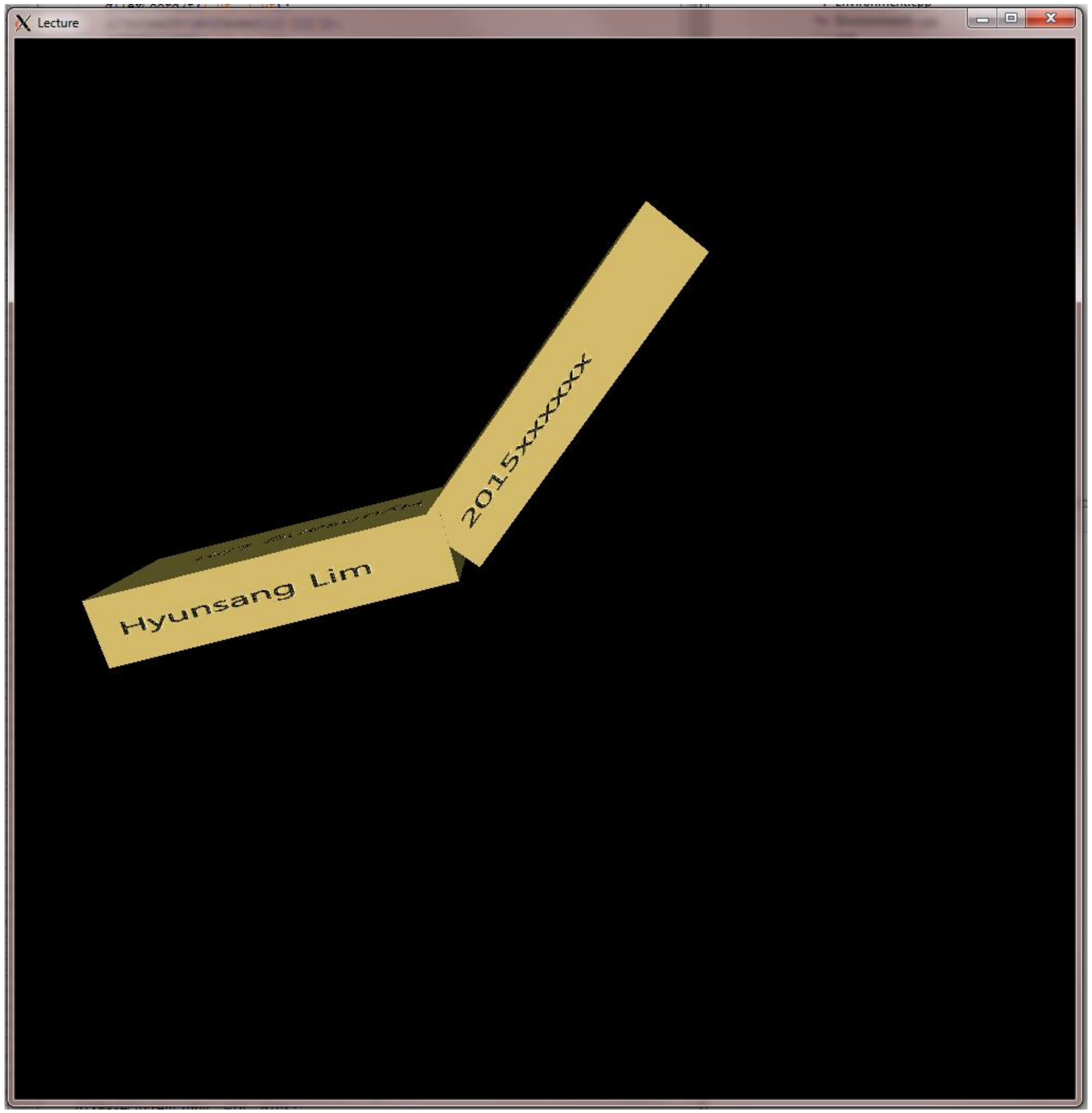

[Normal Mapped Sphere]

Assignment #2

Purpose of Assignment

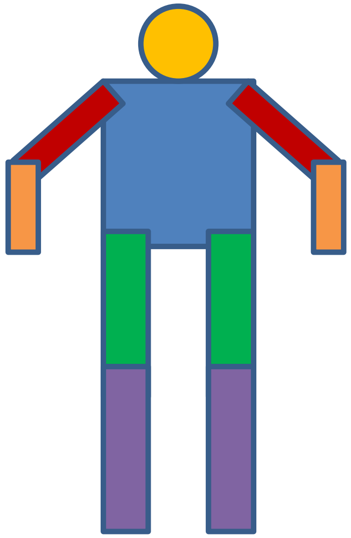
- **Make Robot-arm Program with Shader**
  - Requirements
    1. Texture Mapping
       - Texture Mapping(name, student_ID)
       - Normal Mapping
    2. Phong Lighting Model
       - Ambient, Diffuse, Specular Light with Phong Shading
    3. Using Shaders(Vertex, Fragment)
    4. Run at GPU server

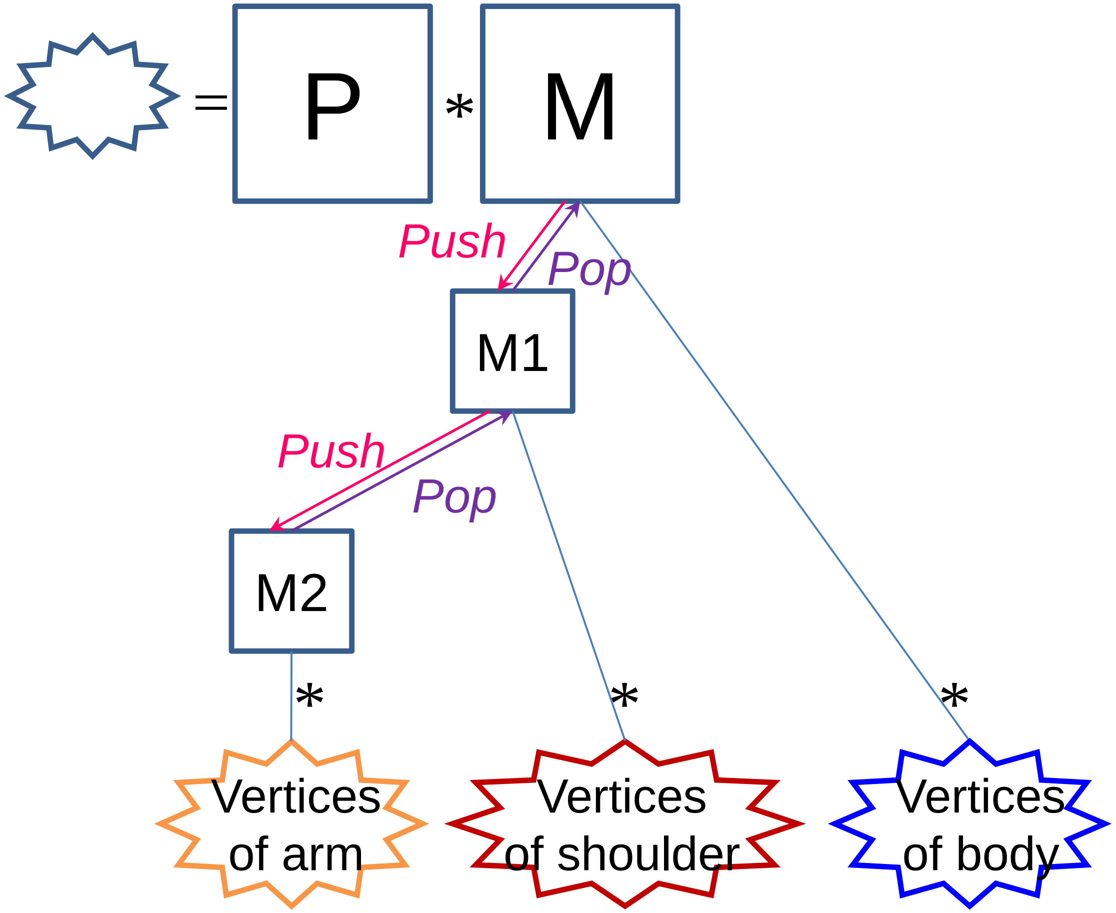
Result Example



Push & Pop

- We can manage the hierarchy by *glPushMatrix()*, *glPopMatrix()*.





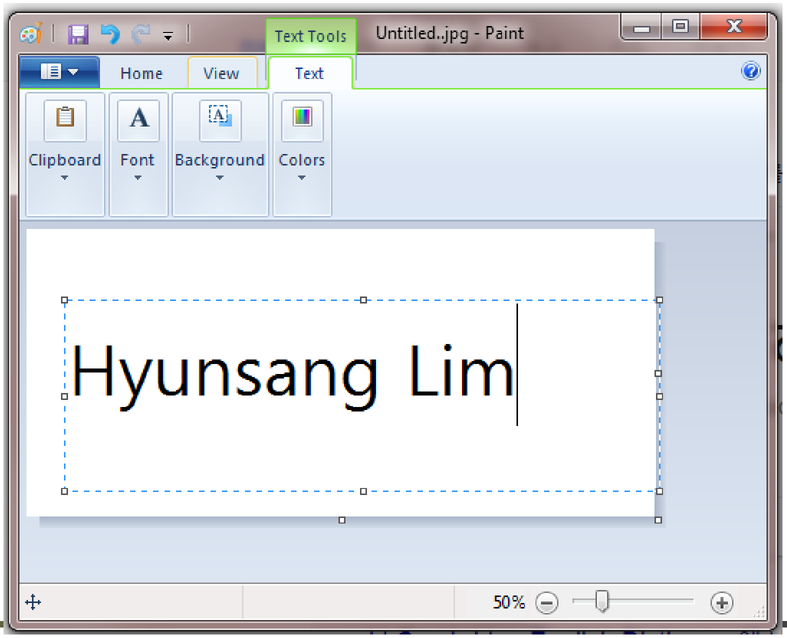
Transform Example : Robot Arm

```
int shoulder = 0, elbow = 0;
void display() {
	/*Initialize Drawing*/
	glPushMatrix();
		glRotatef(20, 1, 0, 1);
		glPushMatrix();
			glTranslatef(-1.0, 0.0, 0.0);
			glRotatef(shoulder, 0.0, 0.0, 1.0);
			glTranslatef(1.0, 0.0, 0.0);

			glPushMatrix();
				glScalef(2.0, 0.4, 1.0);
				glColor3f(1,0,0);
				glutSolidCube(1.0);
			glPopMatrix();

			glTranslatef(1.0, 0.0, 0.0);
			glRotatef(elbow, 0.0, 0.0, 1.0);
			glTranslatef(1.0, 0.0, 0.0);

			glPushMatrix();
				glScalef(2.0, 0.4, 1.0);
				glColor3f(1,1,0);
				glutSolidCube(1.0);
			glPopMatrix();
		glPopMatrix();
	glPopMatrix();
	glXSwapBuffers(dpy, win);
}
```

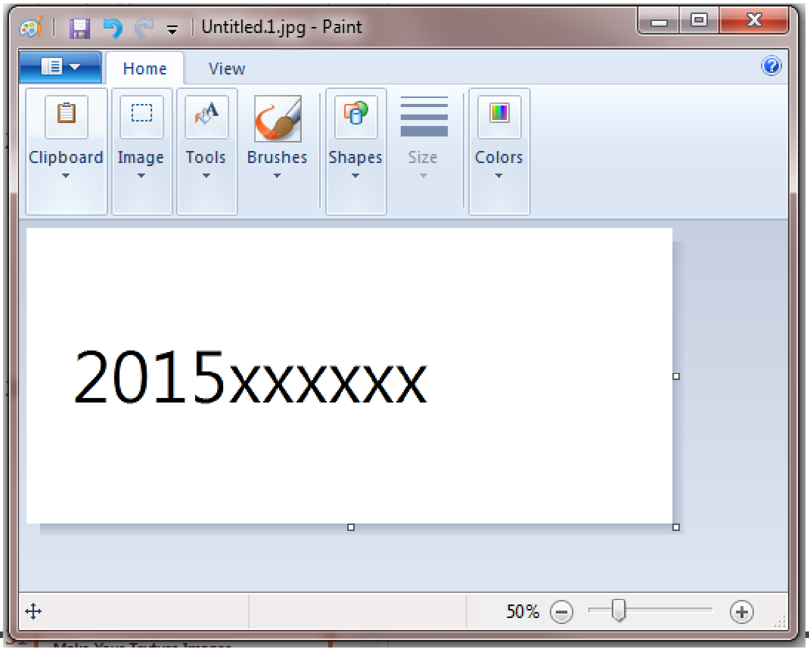
```
void keyPressEvent(char* key_string){
		if(strncmp(key_string, "Up", 2) == 0){
				shoulder = (shoulder+5)%360;
		}else if(strncmp(key_string, "Down", 4) == 0){
				shoulder = (shoulder-5)%360;
		}else if(strncmp(key_string, "Right", 5) == 0){
				elbow = (elbow+5)%360;
		}else if(strncmp(key_string, "Left", 4) == 0){
				elbow = (elbow-5)%360;
		}
}

int main(int argc, char *argv[]) {
	/*CreateWindow*/
	XEvent xev;
	while(1) {
		display();
		XNextEvent(dpy, &xev);
		if(xev.type == KeyPress){
			char *key_string = XKeysymToString(
			XkbKeycodeToKeysym(dpy, xev.xkey.keycode, 0, 0));
			keyPressEvent(key_string);
		}
	}
}
```

Make Your Texture Images

- **Make two color image with your "Paint" program.**
  - Your name
  - Your Student ID
  - We recommend 800*400 size



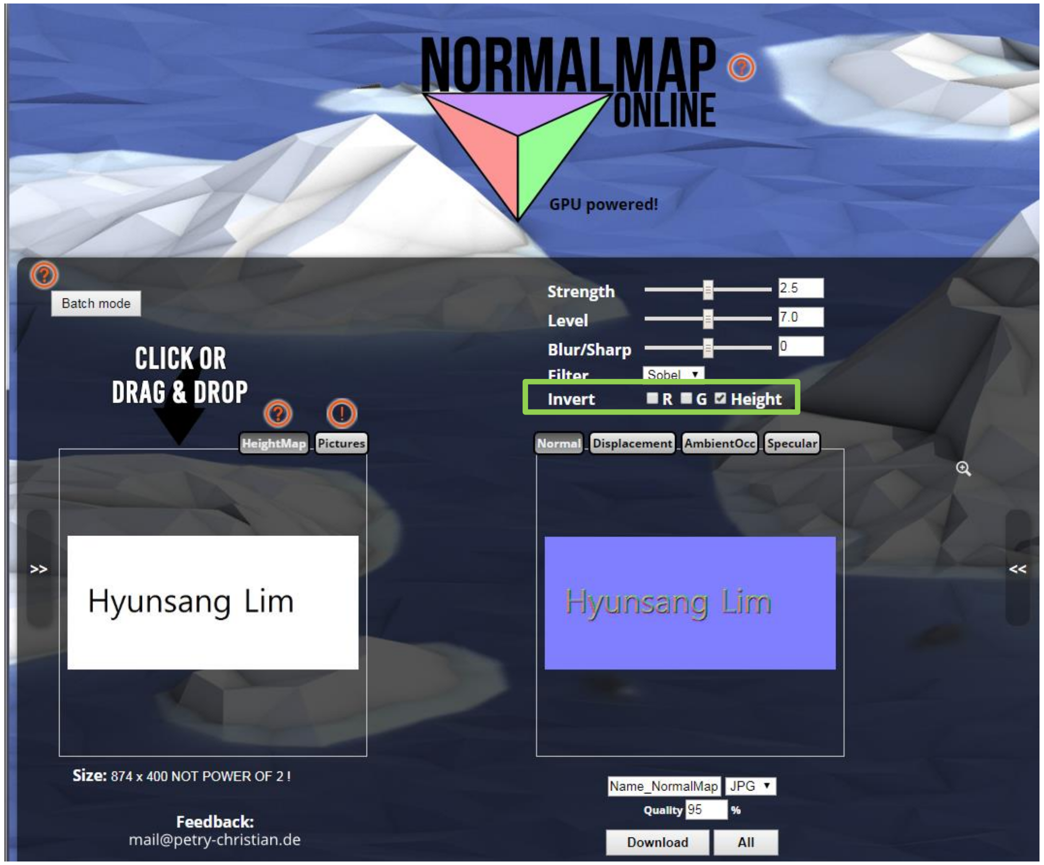
Hyunsang Lim



2015xxxxxx

Make normal image

- You can convert Color image to normal map
  - http://cpetry.github.io/NormalMap-Online/



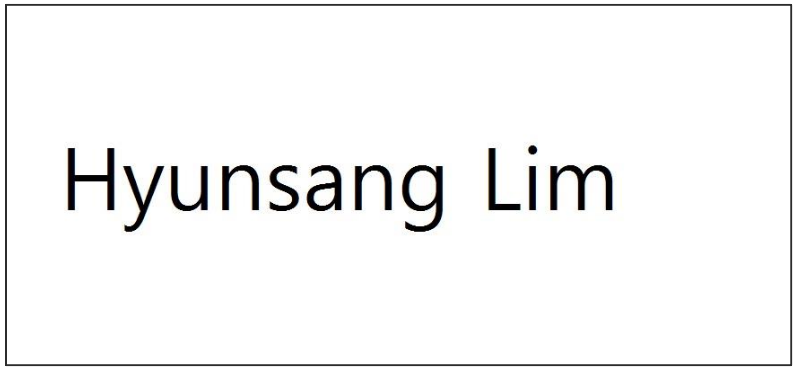
Strength 2.5
Level 7.0
Blur/Sharp 0
Filter Sobel
Invert ☐ R ☐ G ☑ Height

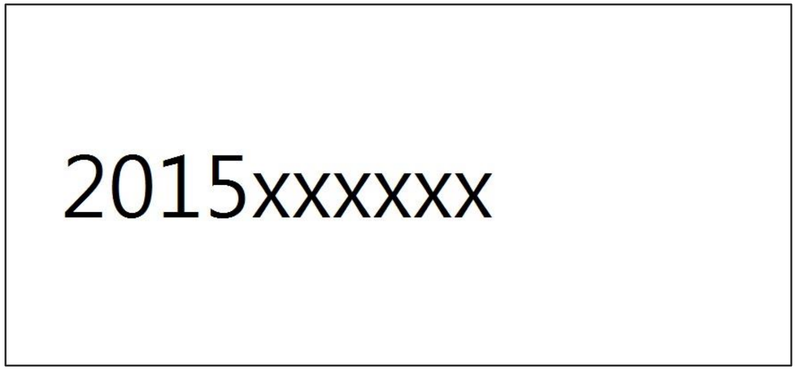
Size: 874 x 400 NOT POWER OF 2 !

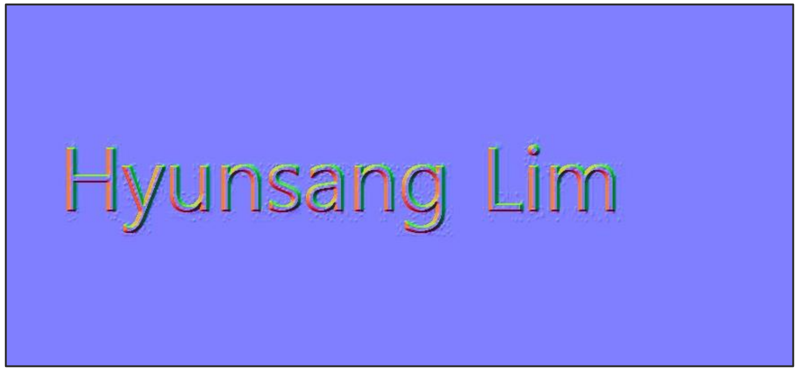
Name_NormalMap JPG
Quality 95 %

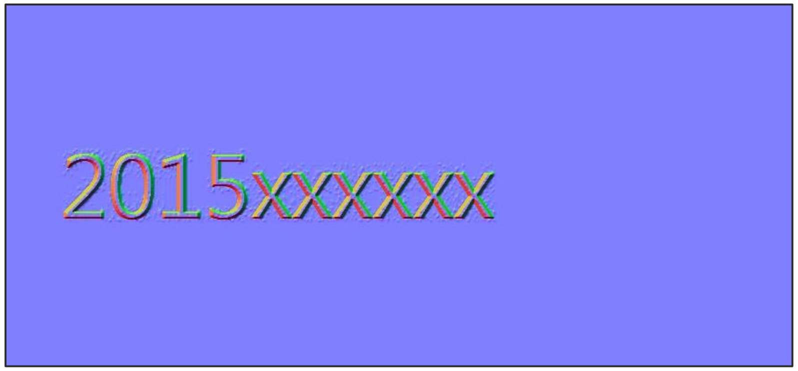
Texture Images Example

- **Transfer Image to your project folder on GPU Server**
  - We'll give file transferring guide









Programming-Hint: Draw Box

- glutSolidCube function has no texture coordinates.
- You should use following modified function.

```
static void drawBox(GLfloat size)
{
          /*initialize vertex & normal value*/
          for (i = 5; i >= 0; i--) {
                    glBegin(GL_QUADS);
                    glNormal3fv(&n[i][0]);
                    glTexCoord2f(0.0f, 0.0f);
                    glVertex3fv(&v[faces[i][0]][0]);
                    glTexCoord2f(1.0f, 0.0f);
                    glVertex3fv(&v[faces[i][1]][0]);
                    glTexCoord2f(1.0f, 1.0f);
                    glVertex3fv(&v[faces[i][2]][0]);
                    glTexCoord2f(0.0f, 1.0f);
                    glVertex3fv(&v[faces[i][3]][0]);
                    glEnd();
          }
}
```

Programming-Hint: TBN Transform

- With this TBN matrix, we can transform normals (extracted from the texture) into model space

```
//fragment Shader
varying normal; //normal from vertex shader
void main(){
        /*..............*/
        vec3 n = normalize(normal);
        vec3 b = normalize(vec3(0,0,1));
        vec3 t = normalize(cross(b, n));
        b = cross(t, n);
        mat3 TBN_Matrix = transpose(mat3(t,b,n));
        /*..............*/

}
```

Programming-Hint: Rendering

```
#version 130
varying vec3 normal, lightDir, halfVector;
uniform sampler2D tex;
uniform sampler2D normal_tex;
void main() {
        vec4 color;
        vec3 n;
        /*Compute TBN Matrix*/
        color = texture2D(tex, glTexCoord[0].st);
        n = (2*texture2D(normal_tex,gl_TexCoord[0].st).rgb-vec3(1.0, 1.0, 1.0))*TBN_Matrix;
        /*Compute Ambient color, Diffuse Color, Specular Color*/

        gl_FragColor = ambientColor+diffuseColor+specularColor;

}
```

Submit the Assignment

- **Submit the zip file @ Blackboard**
  - File name must be “Assignment2_StudentID_Name.zip”
    - Ex. Assignment2_2015000000_박지혁.zip
  - Must include
    - Src file
      - c/c++ and header files
      - Shader files
    - 4 texture images
      - 2 color
      - 2 normal
    - Result running Image file

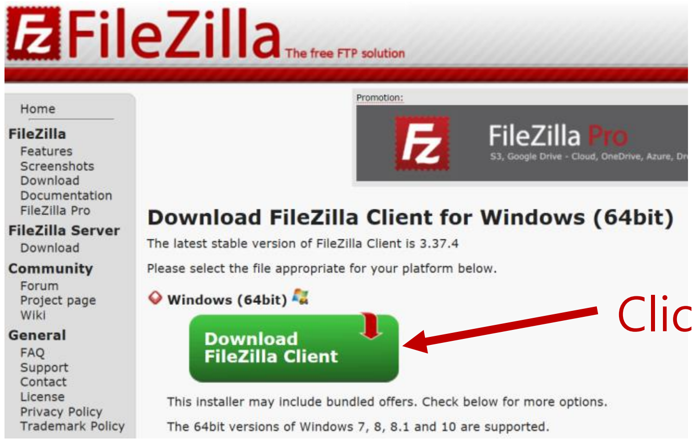
  - Due date: November 4th

Guide: File Transfer to Server

FileZilla

- What is FileZilla
  - FileZilla is the open source cross platform FTP software developed by Tim Kosse

- Download link
  - https://filezilla-project.org/download.php?type=client
  - Recommending to Download with default settings



Download FileZilla Client for Windows (64bit)

The latest stable version of FileZilla Client is 3.37.4

Please select the file appropriate for your platform below.

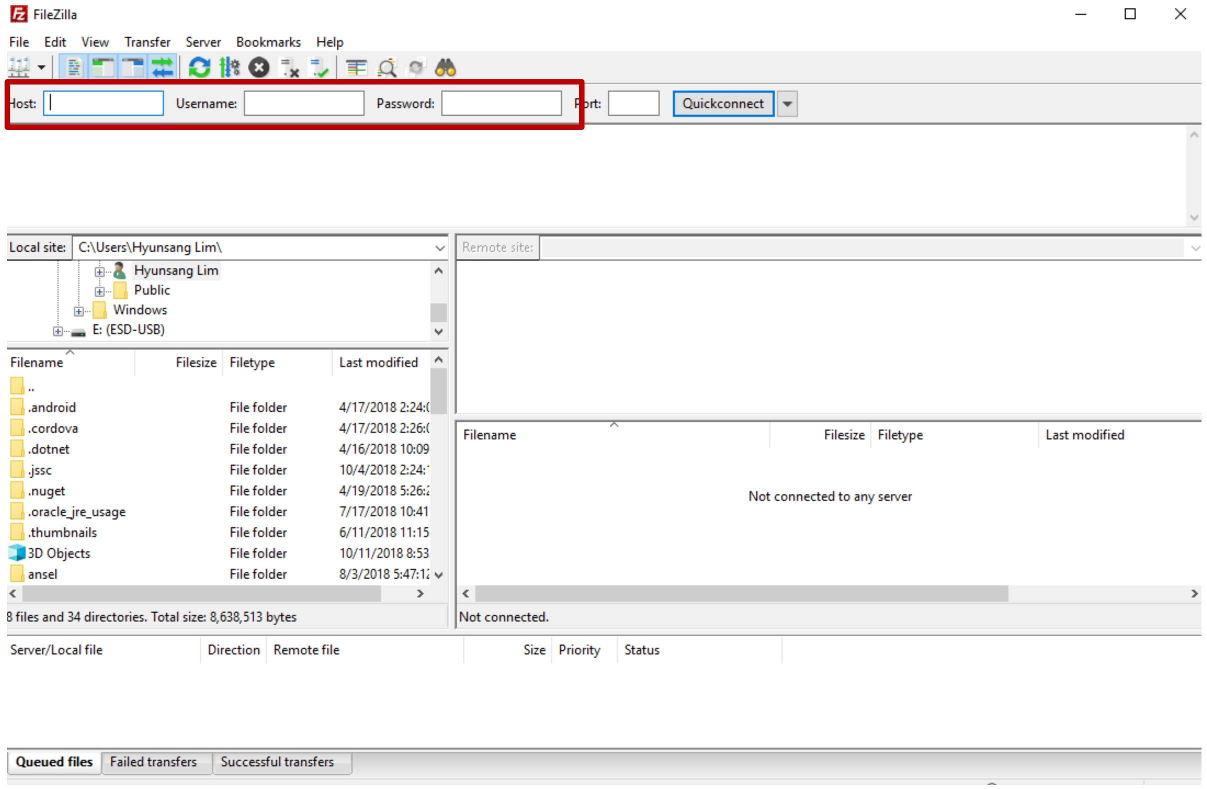
This installer may include bundled offers. Check below for more options.

The 64bit versions of Windows 7, 8, 8.1 and 10 are supported.

Click here

FileZilla Connecting

- Type 'sftp://163.152.20.246' at Host
- Type Username and Password



Local site: C:\Users\Hyunsang Lim\

Hyunsang Lim
Public
Windows
E: (ESD-USB)

Filename	Filesize	Filetype	Last modified
..			
.android		File folder	4/17/2018 2:24:(
.cordova		File folder	4/17/2018 2:26:(
.dotnet		File folder	4/16/2018 10:09
.jssc		File folder	10/4/2018 2:24:
.nuget		File folder	4/19/2018 5:26:
.oracle_jre_usage		File folder	7/17/2018 10:41
.thumbnails		File folder	6/11/2018 11:15
3D Objects		File folder	10/11/2018 8:53
ansel		File folder	8/3/2018 5:47:1

8 files and 34 directories. Total size: 8,638,513 bytes

Filename	Filesize	Filetype	Last modified

Not connected to any server

Not connected.

Server/Local file	Direction	Remote file	Size	Priority	Status

FileZilla File Transfer

- Transferring file by drag & drop file from ③ to ⑤

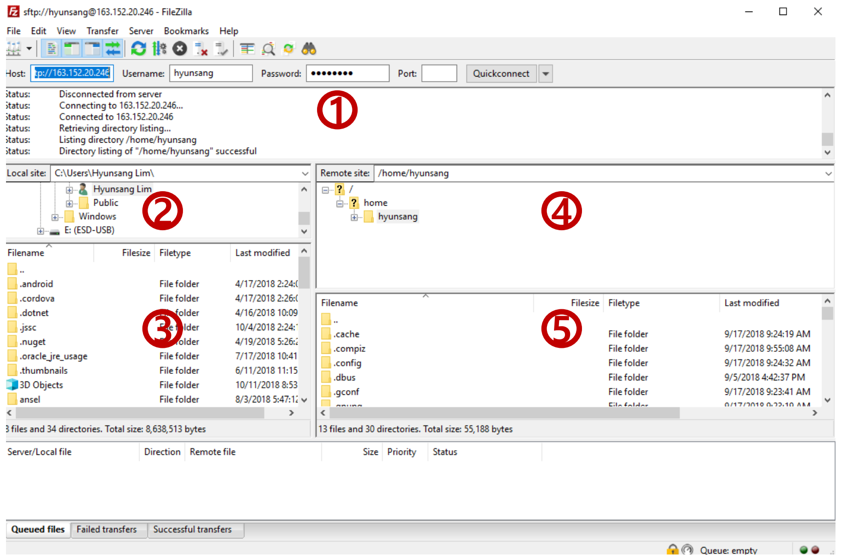
① Representing working states

② Representing folder tree of your PC

③ Representing sub-folders & files in selected folder of you PC

④ Representing folder tree of server

⑤ Representing sub-folders & files in selected folder of server



```
Status:	Disconnected from server
Status:	Connecting to 163.152.20.246...
Status:	Connected to 163.152.20.246
Status:	Retrieving directory listing...
Status:	Listing directory /home/hyunsang
Status:	Directory listing of "/home/hyunsang" successful
```

Local site: C:\Users\Hyunsang Lim\

- Hyunsang Lim
- Public
- Windows
- E: (ESD-USB)

Filename	Filesize	Filetype	Last modified
..			
.android		File folder	4/17/2018 2:24:(
.cordova		File folder	4/17/2018 2:26:(
.dotnet		File folder	4/16/2018 10:09
.jssc		File folder	10/4/2018 2:24:
.nuget		File folder	4/19/2018 5:26:2
.oracle_jre_usage		File folder	7/17/2018 10:41
.thumbnails		File folder	6/11/2018 11:15
3D Objects		File folder	10/11/2018 8:53
ansel		File folder	8/3/2018 5:47:12

3 files and 34 directories. Total size: 8,638,513 bytes

Remote site: /home/hyunsang

- /
  - home
    - hyunsang

Filename	Filesize	Filetype	Last modified
..			
.cache		File folder	9/17/2018 9:24:19 AM
.compiz		File folder	9/17/2018 9:55:08 AM
.config		File folder	9/17/2018 9:24:32 AM
.dbus		File folder	9/5/2018 4:42:37 PM
.gconf		File folder	9/17/2018 9:23:41 AM

13 files and 30 directories. Total size: 55,188 bytes

Server/Local file	Direction	Remote file	Size	Priority	Status