

COSC 4332 Computer Graphics

OpenGL Lab3 Texture using Soil

Dr. Khaled Rabieh

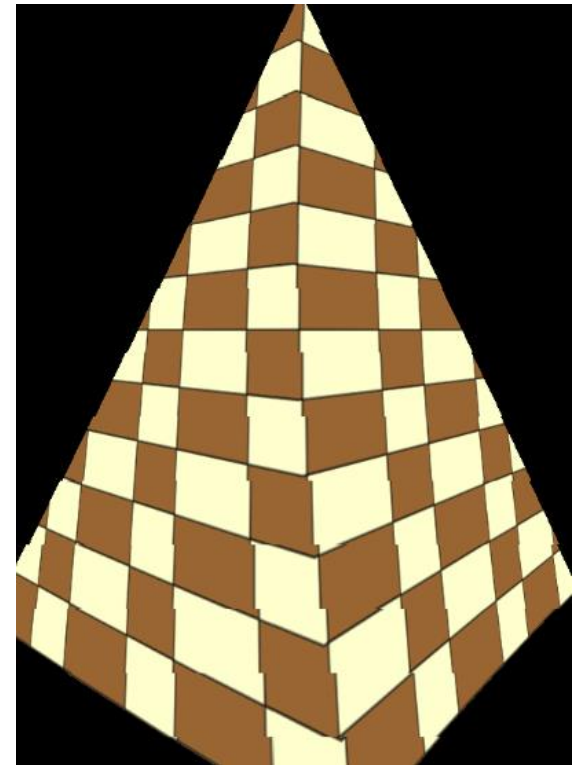
Outline

1. OpenGL

1. Textures Introduction
2. Loading images using Soil

Texture Mapping

- Texture mapping means applying any type of picture on the faces/ surfaces of a model.
- Uses of Texturing
 - Simulating materials
 - Reducing geometric complexity
 - Reflections



OpenGL Texturing Basic Strategy

To generate a texture

1. Generate a texture object
2. Bind the texture to the type of array
 - GL_TEXTURE_1D
 - GL_TEXTURE_2D //we will use images (2D array)
 - GL_TEXTURE_3D
3. Read or generate an image
 - OpenGL does not have built-in images reading library
 - Use an external library (SOIL)

OpenGL Texturing Basic Strategy

4. Enable texturing

5. Specify texture parameters

- Proper mapping function is left to application
- Wrapping or filtering

Textures in OpenGL 1

Binding an image to a Texture object in OPENGGL

- 1-generate a texture
- 2-bind a texture to 2D array
- 3-load an image using SOIL
- 4-map the image to a texture
- 5-free resources

```
GLuint loadtextures(char* filename)
{
    //textures are objects that need to be generated first by calling a function.
    GLuint textureId;
    glGenTextures(1, &textureId);
    // textures have to be bound to apply operations to them.
    //Since images are 2D arrays of pixels, it will be bound to the GL_TEXTURE_2D target.
    glBindTexture(GL_TEXTURE_2D, textureId);
    //Load an image using SOIL
    image = SOIL_load_image(filename, &width, &height, 0, SOIL_LOAD_RGB);
    //Map the image to the texture
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height, 0, GL_RGB, GL_UNSIGNED_BYTE, image);
    //free the memory from the image since no need now for the image
    SOIL_free_image_data(image);
    return textureId;
}
```

Define an Image as a Texture

```
glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, 2, 2, 0, GL_RGB, GL_FLOAT, pixels);
```

Specifies the internal pixel format

Width and height

describe the
format of the
pixels in the array
that will be
loaded and the
final parameter

Pixel array

```
glTexImage2D(GL_TEXTURE_2D, 0, 3, 512, 512, 0, GL_RGB,  
GL_UNSIGNED_BYTE, image);
```

Textures in OpenGL (2)

- Enable texturing
 - in your init method

```
void initRendering() {  
    glEnable(GL_TEXTURE_2D);  
    glEnable(GL_DEPTH_TEST);  
    quad = gluNewQuadric();  
    quad2 = gluNewQuadric();  
    {  
        _textureId = loadtextures("pyramids.jpg");  
        _textureId1 = loadtextures("2.jpg");  
    }  
}
```

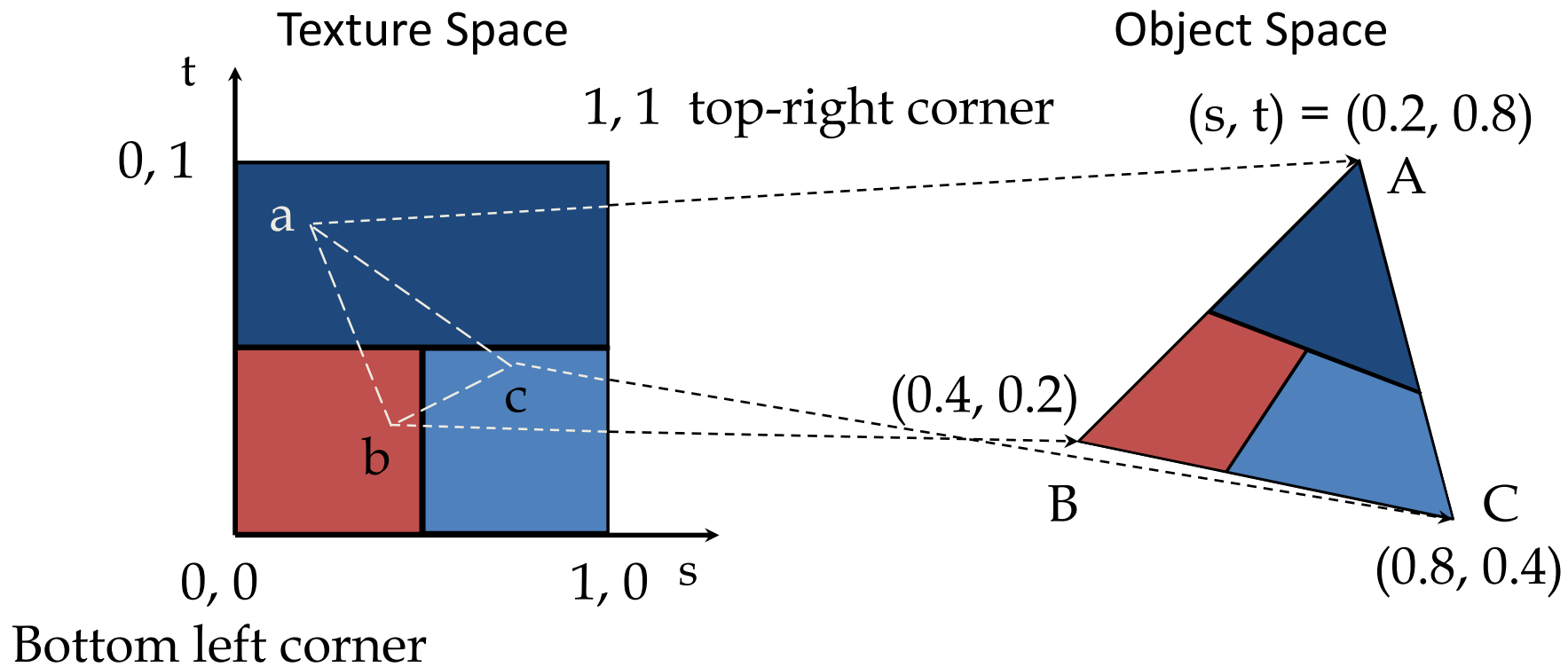
No Texturing in the scene
without ENABLING
texturing

Loading multitextures

Hides background
Gives 3D effect to the scene

Texture Co-ordinates

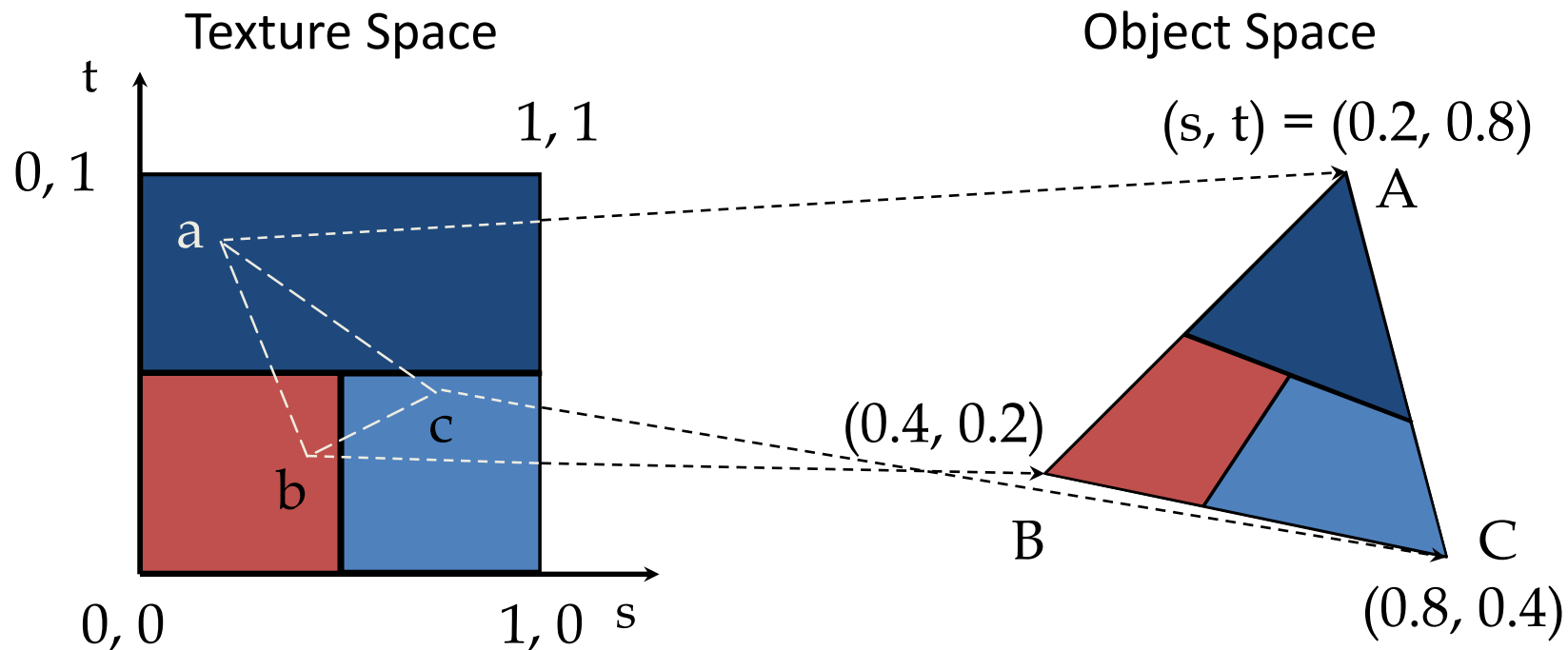
- s, t are used to determine the texture co-ordinates
- For every object vertex co-ordinates, you should specify a corresponding texture co-ordinates



Mapping a Texture

Based on parametric texture coordinates
glTexCoord*() specified at each vertex

```
glBegin(GL_TRIANGLES);  
  
glTexCoord2d(0.5, 1);  
glVertex3f(0.0f, 1.0f, 0.0f);  
glTexCoord2d(0, 0);  
glVertex3f(-1.0f, -1.0f, 1.0f);  
glTexCoord2d(1, 0);  
glVertex3f(1.0f, -1.0f, 1.0f)  
... ..
```



Wrapping Modes

- How the texture should be sampled when a coordinate is outside the range of 0 to 1 ?
 - GL_repeat
 - GL_mirrored_repeat
 - GL_clamp_to_edge
 - GL_clamp_to_border



GL_REPEAT



GL_MIRRORED_REPEAT



GL_CLAMP_TO_EDGE



GL_CLAMP_TO_BORDER

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

Filtering Modes

- Texture coordinates are resolution independent, they won't always match a pixel exactly
- A texture image can be stretched beyond its original size or when it's sized down.

GL_LINEAR: Returns the weighted average of the 4 pixels surrounding the given coordinates.

GL_NEAREST: Returns the pixel that is closest to the coordinates.

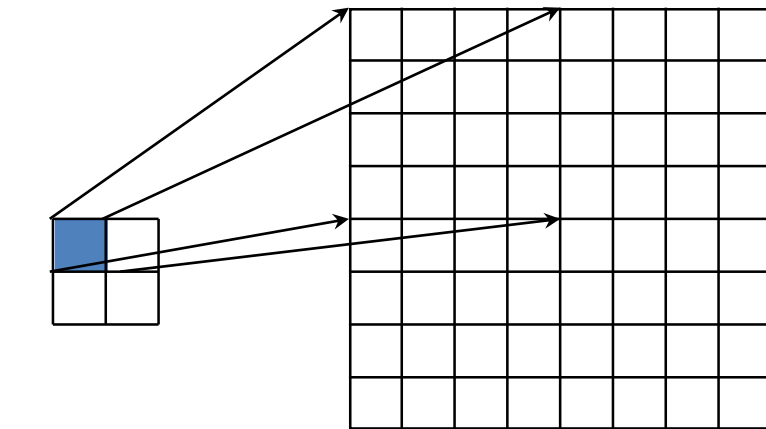


GL_LINEAR



GL_NEAREST

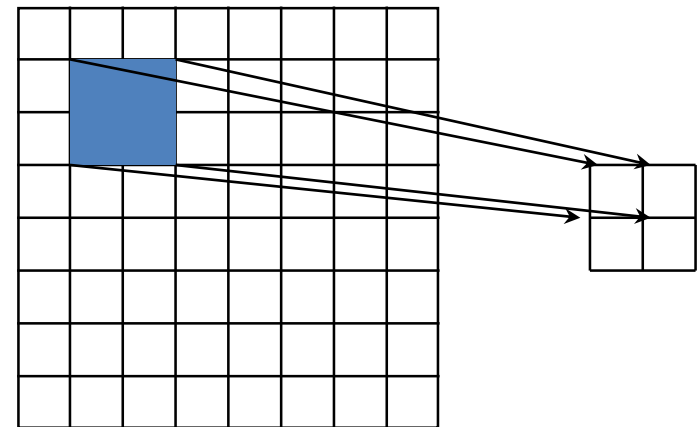
Filtering Modes



Texture

Object

Magnification



Texture

Object

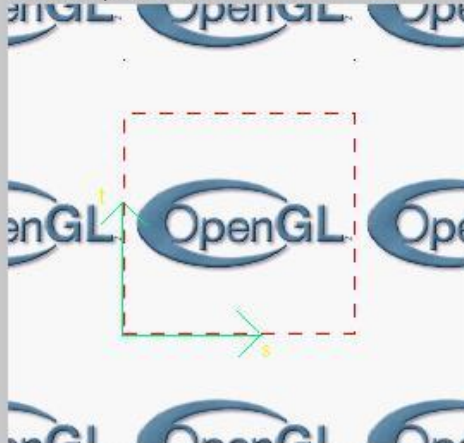
Minification

Tutorial: Texture

Screen-space view



Texture-space view



Command manipulation window

```
GLfloat border_color[ ] = { 1.00 , 0.00 , 0.00 , 1.00 };
GLfloat env_color[ ] = { 0.00 , 1.00 , 0.00 , 1.00 };

glTexParameterfv(GL_TEXTURE_2D, GL_TEXTURE_BORDER_COLOR, border_color);
glTexEnvfv(GL_TEXTURE_ENV, GL_TEXTURE_ENV_COLOR, env_color);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);

glEnable(GL_TEXTURE_2D);
gluBuild2DMipmaps(GL_TEXTURE_2D, 3, w, h, GL_RGB, GL_UNSIGNED_BYTE, image);

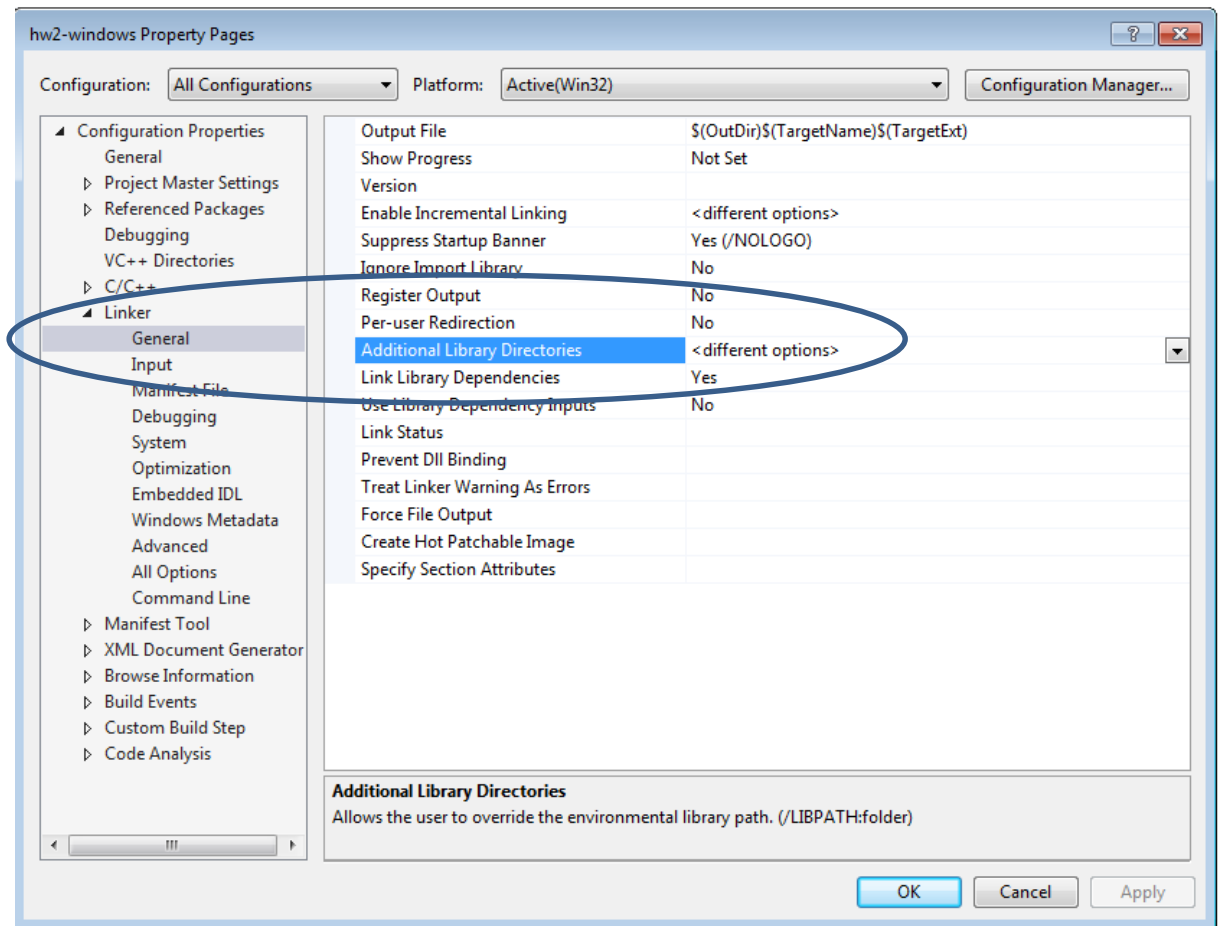
glColor4f( 0.60 , 0.60 , 0.60 , 1.00 );
glBegin(GL_POLYGON);
glTexCoord2f( 0.0 , 0.0 ); glVertex3f( -1.0 , -1.0 , 0.0 );
glTexCoord2f( 1.0 , 0.0 ); glVertex3f( 1.0 , -1.0 , 0.0 );
glTexCoord2f( 1.0 , 1.0 ); glVertex3f( 1.0 , 1.0 , 0.0 );
glTexCoord2f( 0.0 , 1.0 ); glVertex3f( -1.0 , 1.0 , 0.0 );
glEnd();
```

Click on the arguments and move the mouse to modify values.

Soil Library integration with OpenGL

1. Right click on your project in the solution explorer and then click on Properties.

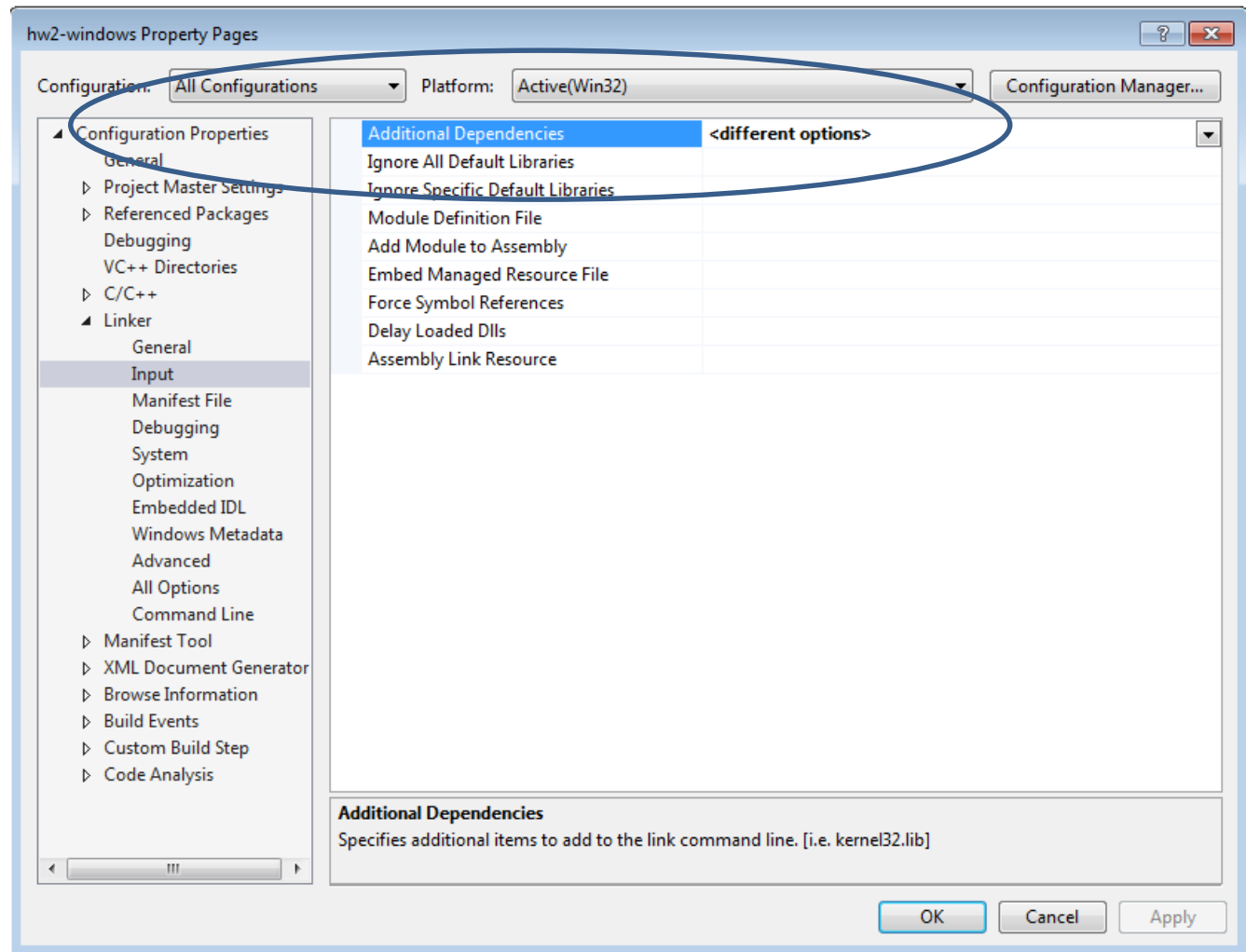
Next open Configuration Properties and then Linker.



Select the directory that contain the main library to Additional Library Directories
Lib Directory and contains libsoil.a library

Soil Library integration with OpenGL

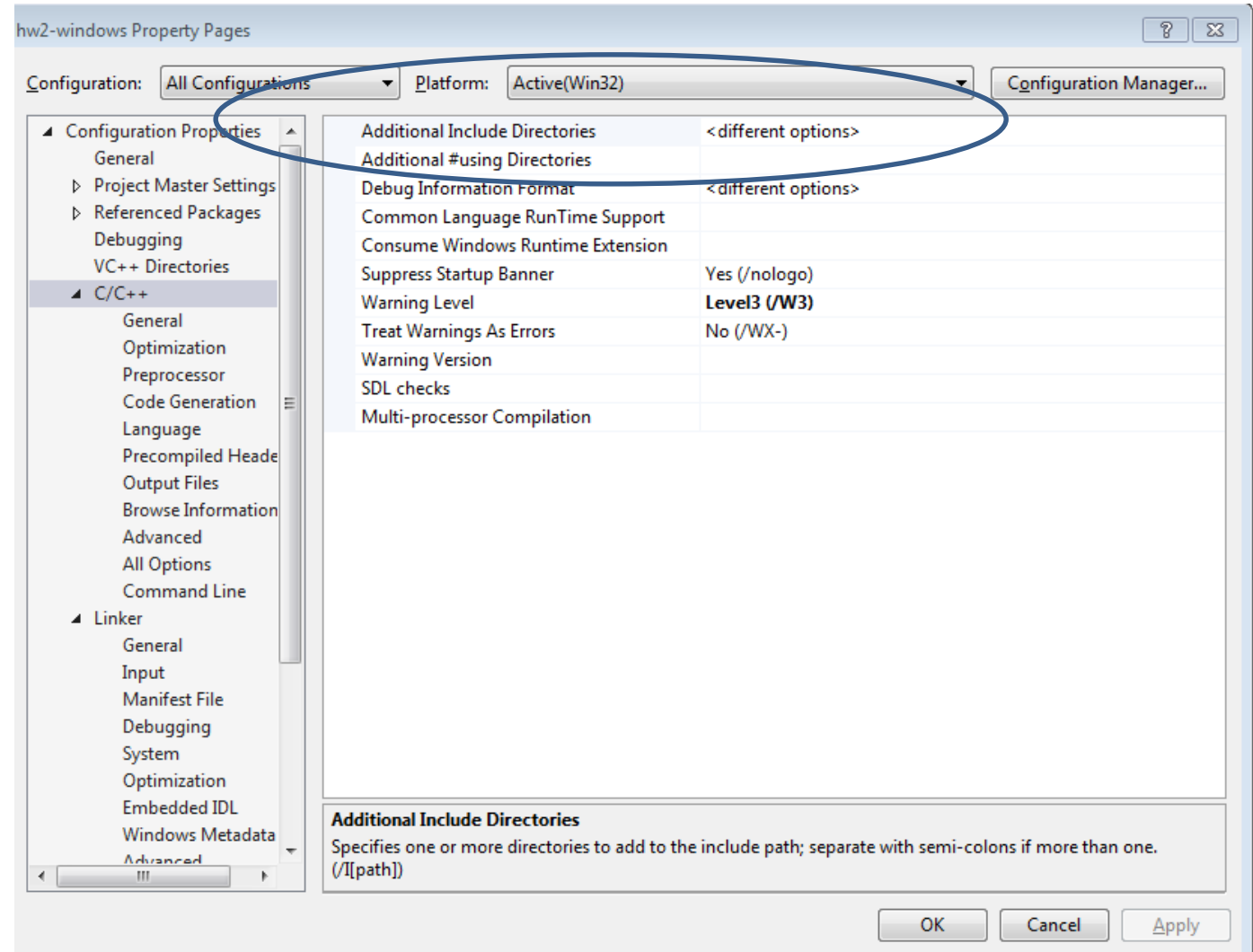
2. Linker -> Input and add the actual library files under Additional Dependencies.



Select the actual library with the name libsoil.a library

Soil Library integration with OpenGL

3. Include the directory containing the src files in C/C++ -> Additional Include Directories.




It is called src directory


Packages


NuGet: hw2-windows x sphertexture.cpp

Browse Installed Updates

Search (Ctrl+E) 🔍 ↻ ☐ Include prerelease

 **glfw** by Marcus Geelnard, Camilla Berglund ✓ v3.2.1
GLFW is an Open Source, multi-platform library for creating windows with OpenGL contexts and receiving input and events.

 **glfw.redist** by Marcus Geelnard, Camilla Berglund ✓ v3.2.1
Redistributable components for for package 'glfw'

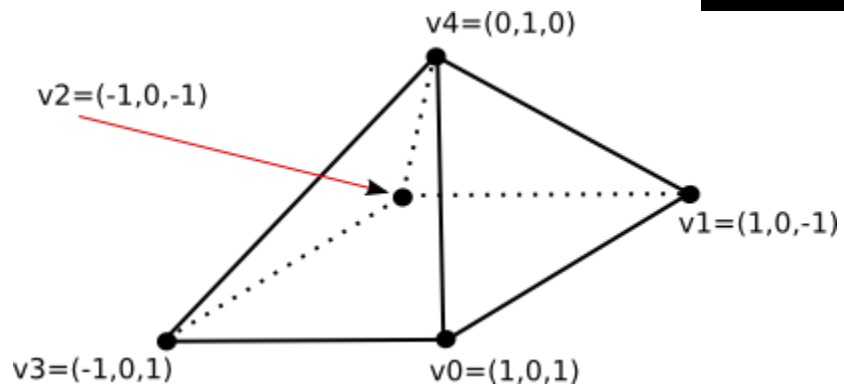
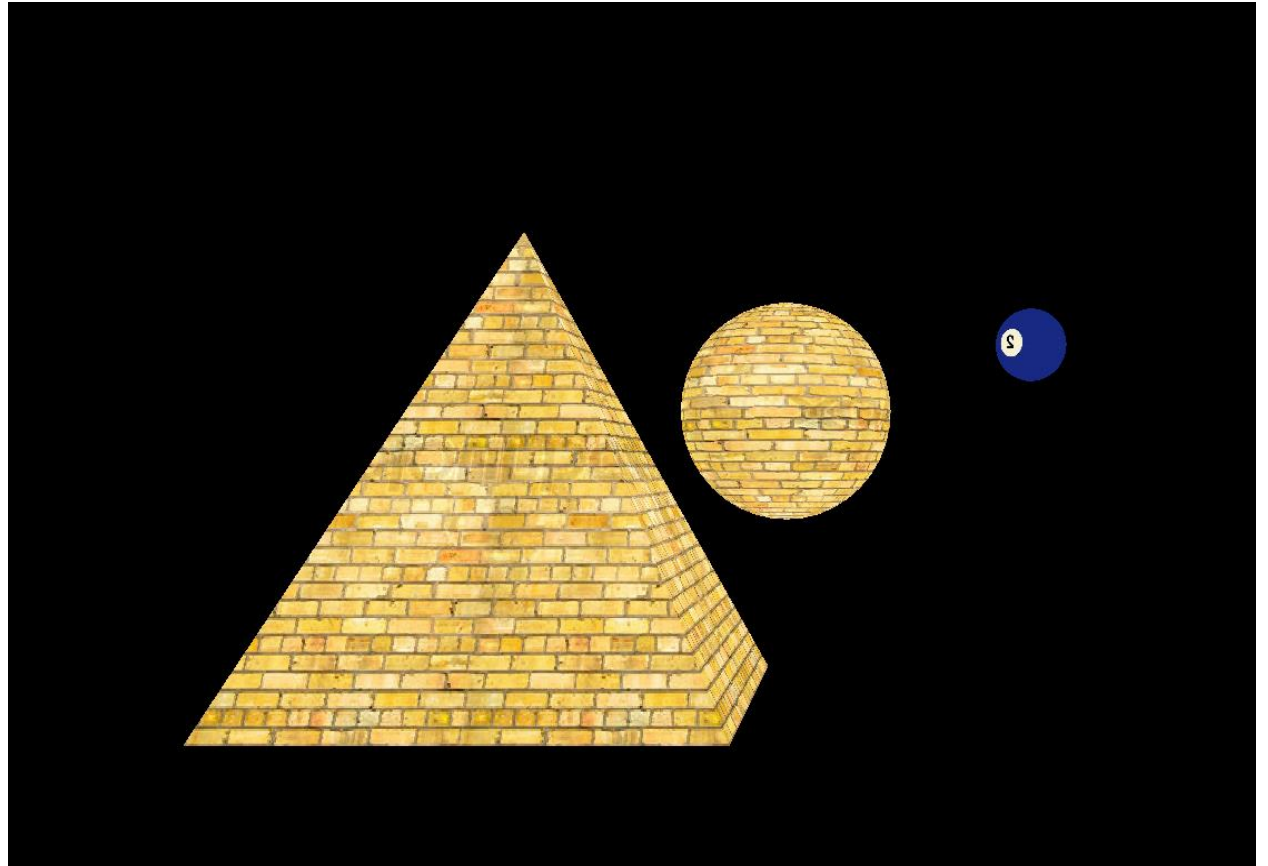
 **soil** by www.lonesock.net, Sean Barrett ✓ v1.16.0
SOIL is a tiny C library used primarily for uploading textures into OpenGL

Donot forget to add soil.h in your project

```
#include <soil.h>
```

Project

Multi-Texture Translation



Thank You



Questions

Khaled Rabieh