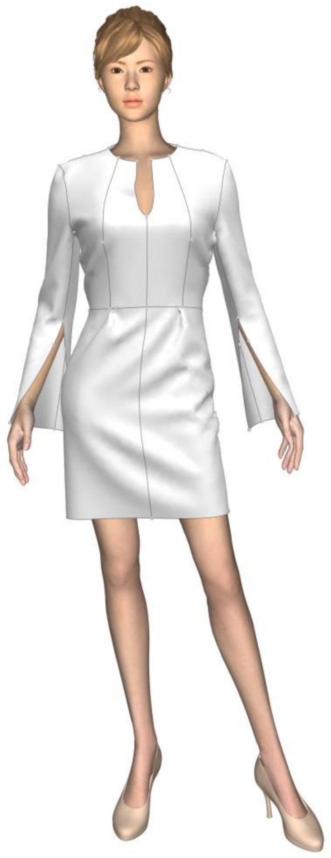
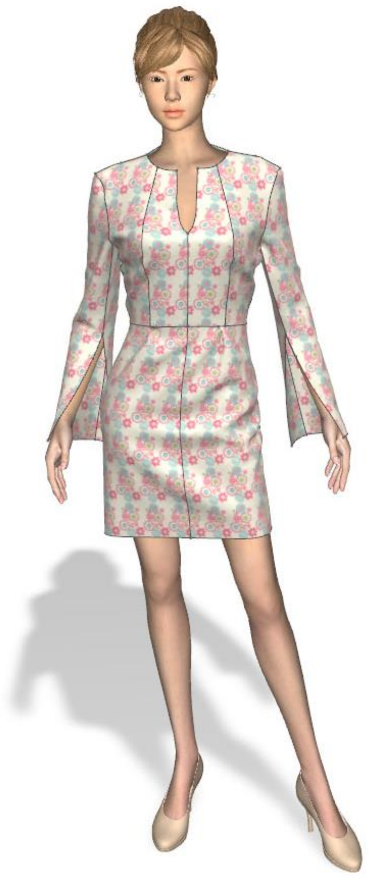
14주차

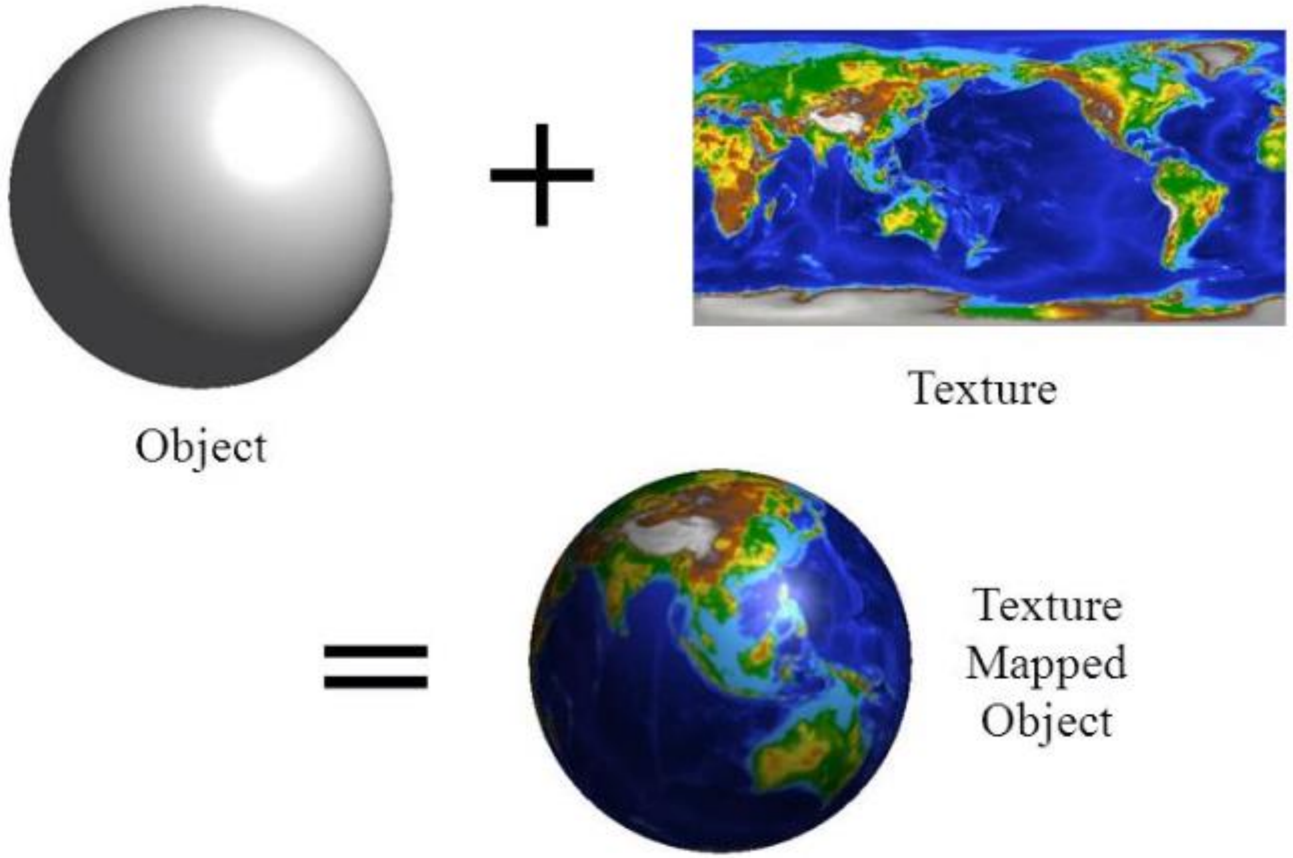
Texture Mapping

What is Texture Mapping?


Without Texture Mapping

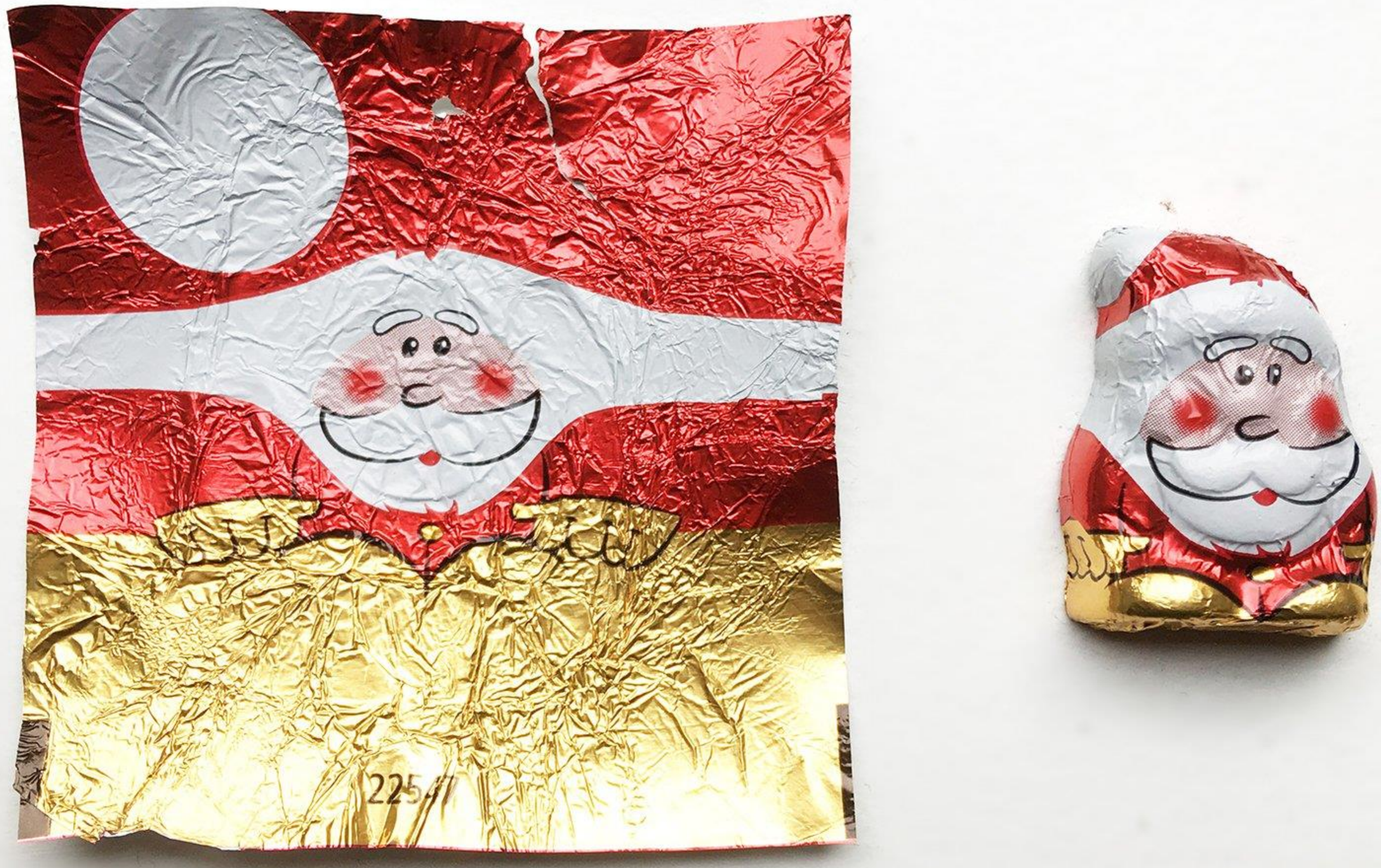

With Texture Mapping

What is Texture Mapping?



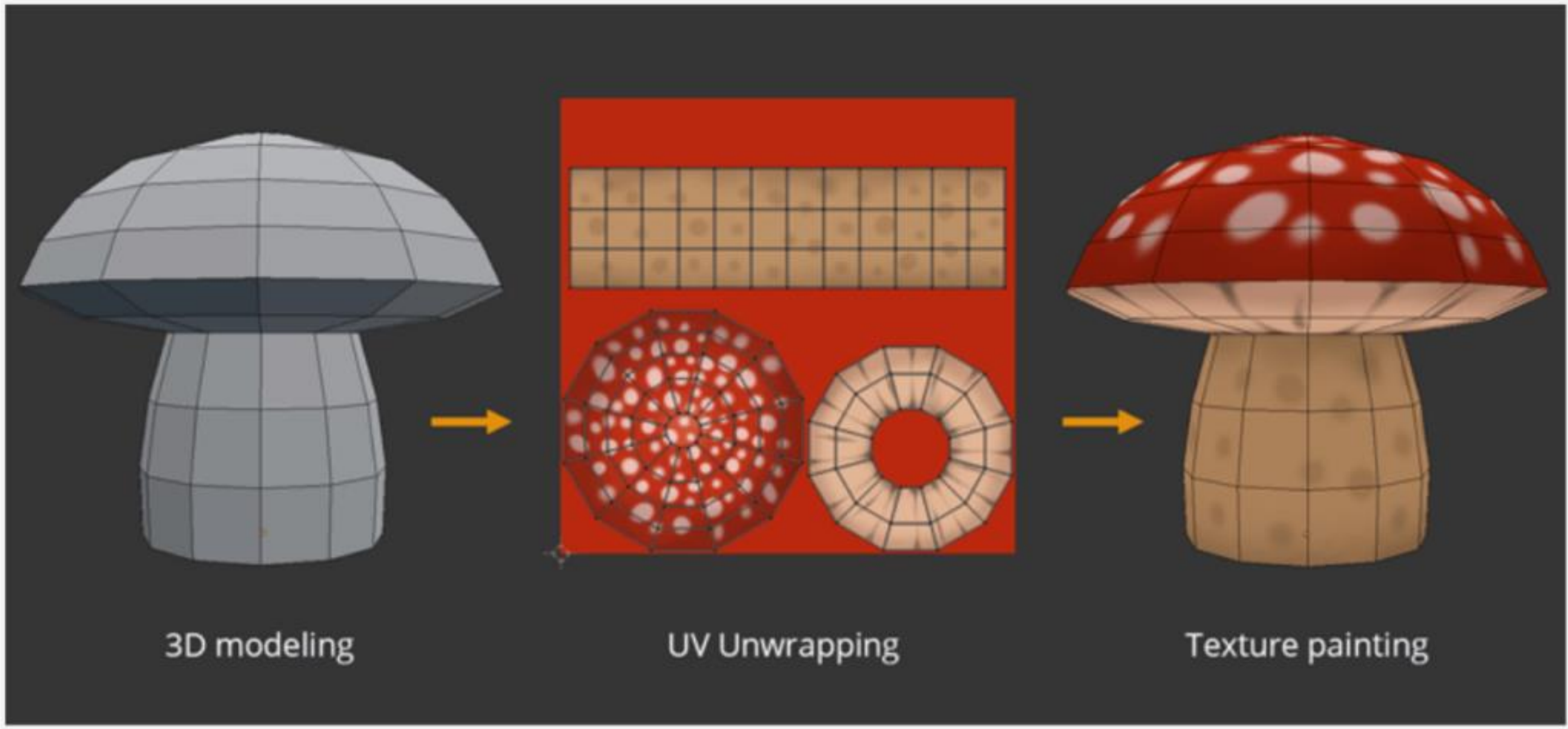
Texture Mapping

- Texture for a face model



Texture Mapping

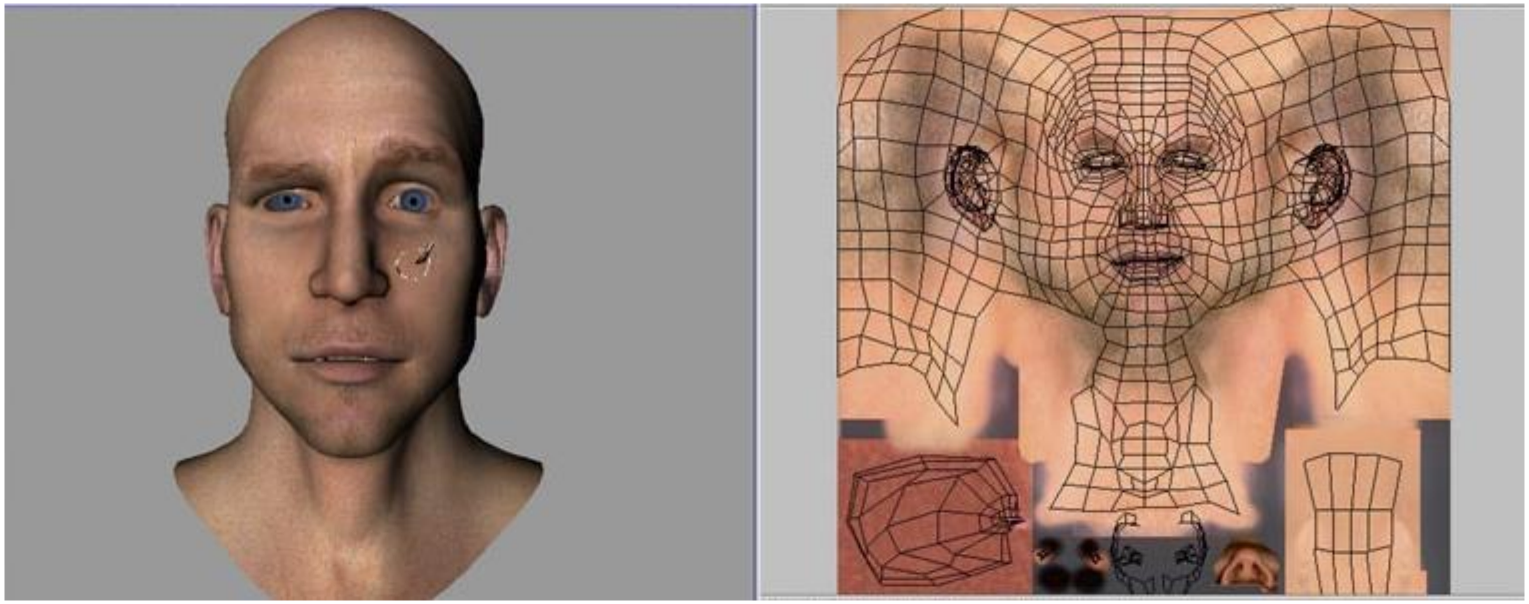
- Texture for a 3D model



The process of UV unwrapping

Texture Mapping

- Texture for a 3D face model

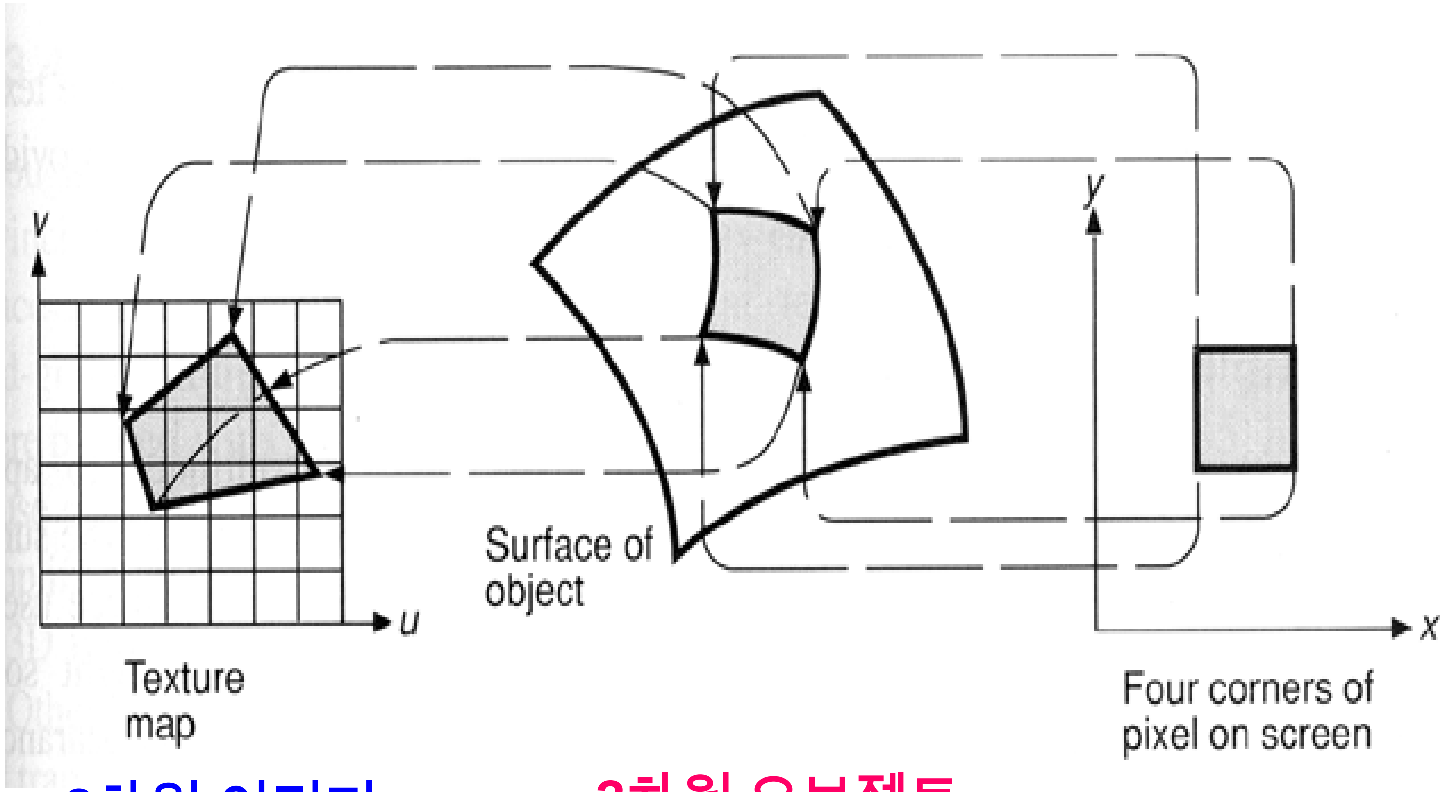


Texture Mapping

OpenGL Texture Coordinates

What is Texture Mapping?

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

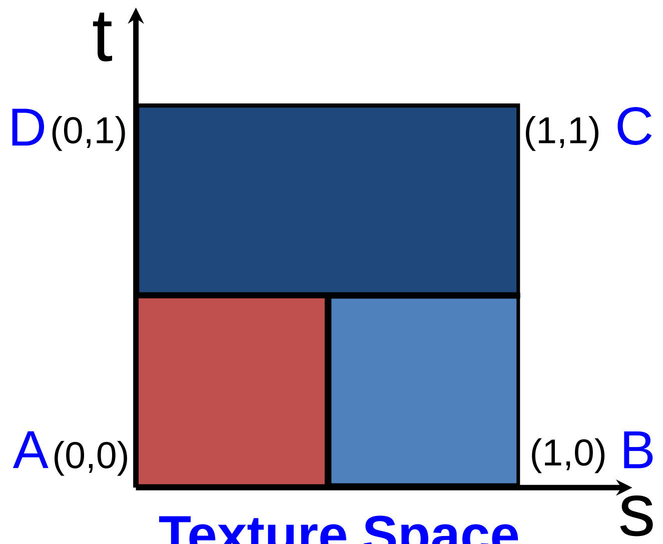


2차원 이미지

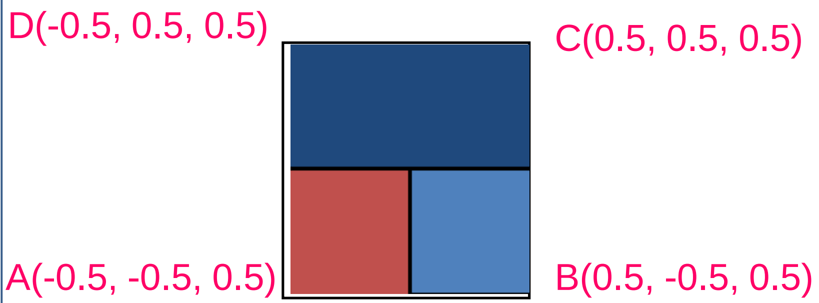
3차원 오브젝트

Texture Coordinates

```
glBegin(GL_POLYGON);
    //A
    glTexCoord2f(0, 0);
    glVertex3f(-0.5, -0.5, 0.5);
    // B
    glTexCoord2f(1, 0);
    glVertex3f(0.5, -0.5, 0.5);
    // C
    glTexCoord2f(1, 1);
    glVertex3f(0.5, 0.5, 0.5);
    // D
    glTexCoord2f(0, 1);
    glVertex3f(-0.5, 0.5, 0.5);
glEnd();
```



Texture Space
2차원 이미지

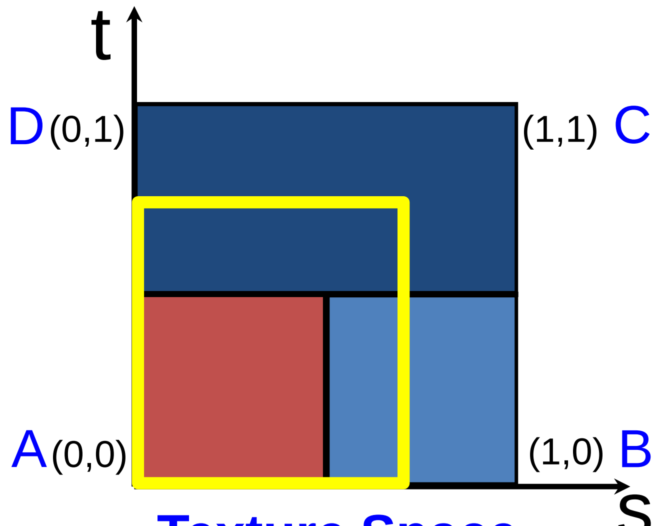


Object Space
3차원 오브젝트

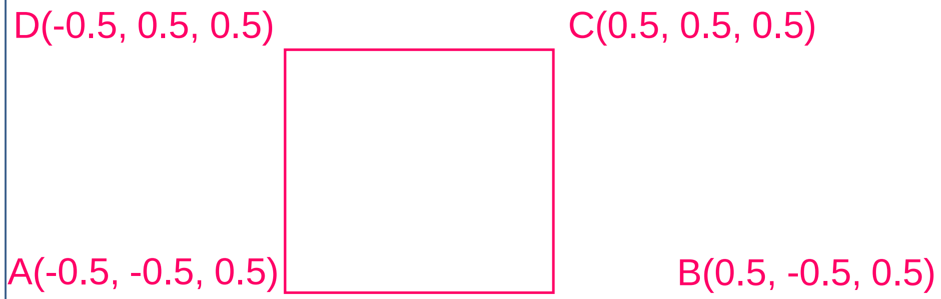
Texture Coordinates

- 텍스처의 일부만 오브젝트에 매핑

```
glBegin(GL_POLYGON);
    glTexCoord2f(0, 0);
    glVertex3f(-0.5, -0.5, 0.5); //A

    glTexCoord2f(0.7, 0);
    glVertex3f(0.5, -0.5, 0.5); //B

    glTexCoord2f(0.7, 0.7);
    glVertex3f(0.5, 0.5, 0.5); //C

    glTexCoord2f(0, 0.7);
    glVertex3f(-0.5, 0.5, 0.5); //D
glEnd();
```



Texture Space
2차원 이미지

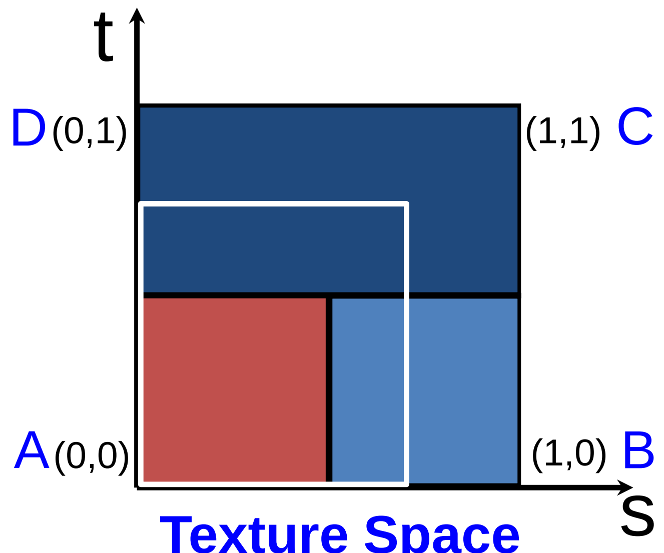


Object Space
3차원 오브젝트

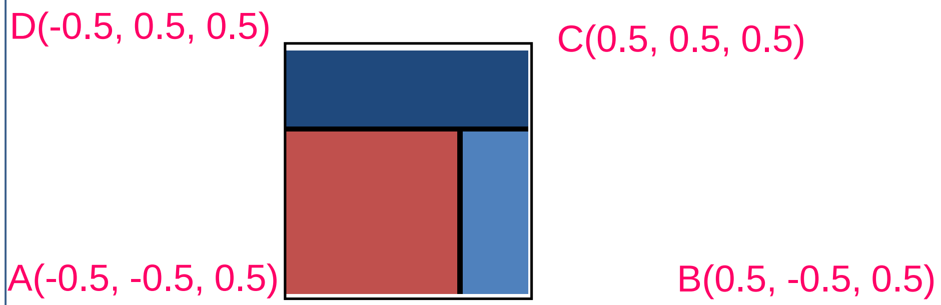
Texture Coordinates

- 텍스처의 일부만 오브젝트에 매핑

```
glBegin(GL_POLYGON);
    glTexCoord2f(0, 0);
    glVertex3f(-0.5, -0.5, 0.5); //A

    glTexCoord2f(0.7, 0);
    glVertex3f(0.5, -0.5, 0.5); //B

    glTexCoord2f(0.7, 0.7);
    glVertex3f(0.5, 0.5, 0.5); //C

    glTexCoord2f(0, 0.7);
    glVertex3f(-0.5, 0.5, 0.5); //D
glEnd();
```



Texture Space
2차원 이미지

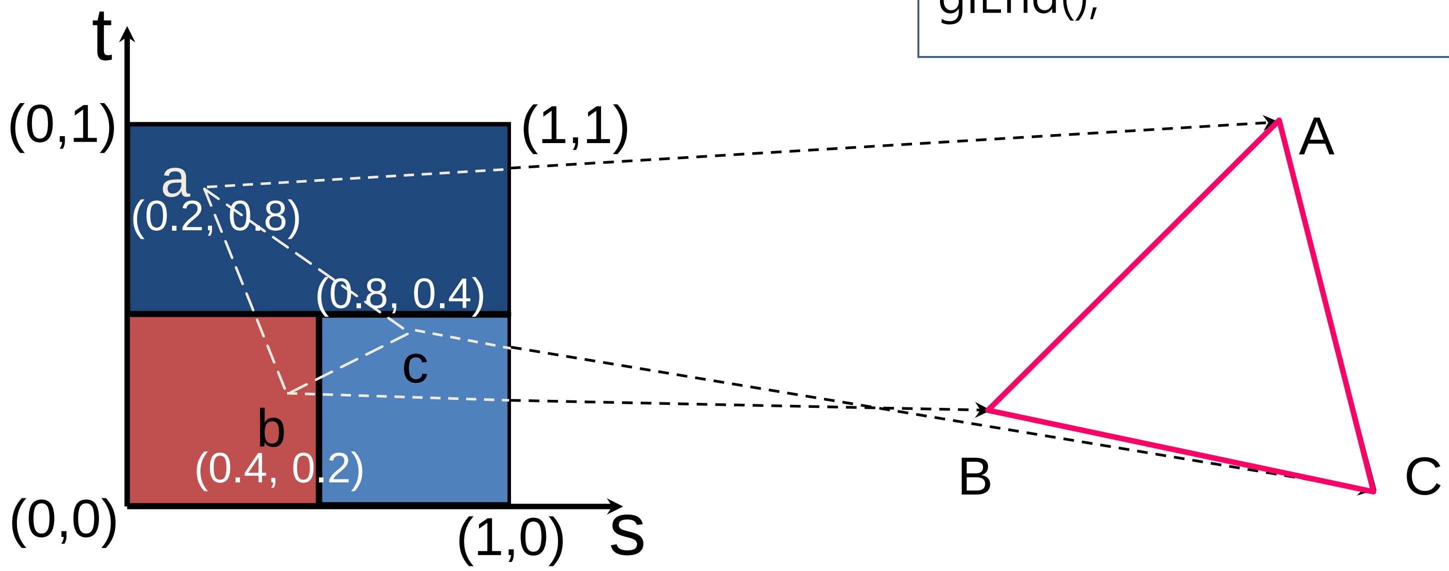


Object Space
3차원 오브젝트

Texture Coordinates

- 텍스처의 일부만 오브젝트에 매핑

```
glBegin(GL_POLYGON);
    glTexCoord2f(0.2, 0.8);
    glVertex3f(x0, y0, z0);  // A

    glTexCoord2f(0.4, 0.2);
    glVertex3f(x1, y1, z1);  // B

    glTexCoord2f(0.8, 0.4);
    glVertex3f(x1, y1, z1);  // B
glEnd();
```



Texture Space
2차원 이미지

Object Space
3차원 오브젝트

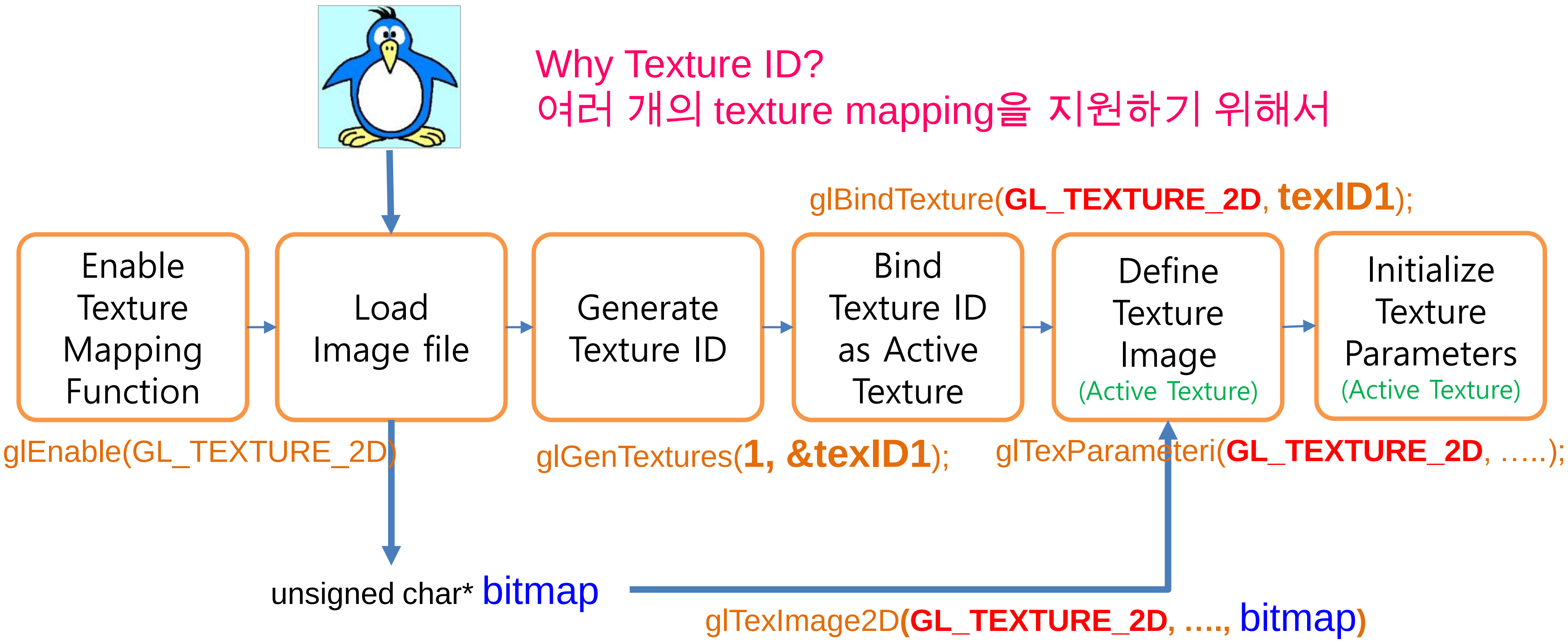
OpenGL Texture Mapping

Initializing OpenGL Texture Mapping

Initializing Texture Mapping

- OpenGL Texture Mapping

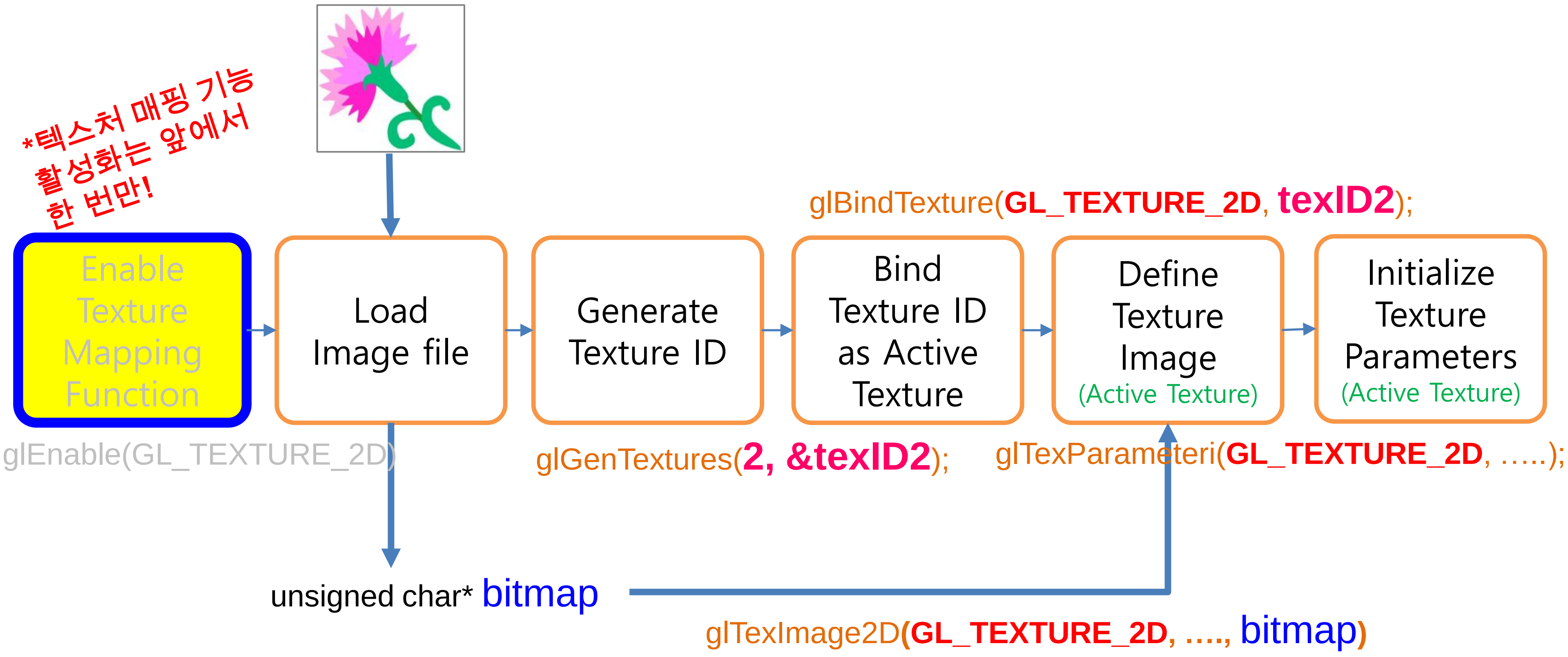
Why Texture ID?
여러 개의 texture mapping을 지원하기 위해서


Enable Texture Mapping Function
Load Image file
Generate Texture ID
Bind Texture ID as Active Texture
Define Texture Image (Active Texture)
Initialize Texture Parameters (Active Texture)

```
glEnable(GL_TEXTURE_2D)
glGenTextures(1, &texID1);
glBindTexture(GL_TEXTURE_2D, texID1);
glTexParameteri(GL_TEXTURE_2D, …..);
unsigned char* bitmap
glTexImage2D(GL_TEXTURE_2D, …., bitmap)
```

Initializing Texture Mapping

- OpenGL Texture Mapping


*텍스처 매핑 기능 활성화는 앞에서 한 번만!
Enable Texture Mapping Function
Load Image file
Generate Texture ID
Bind Texture ID as Active Texture
Define Texture Image (Active Texture)
Initialize Texture Parameters (Active Texture)

glEnable(GL_TEXTURE_2D)

glGenTextures(**2, &texID2**);

glBindTexture(**GL_TEXTURE_2D**, **texID2**);

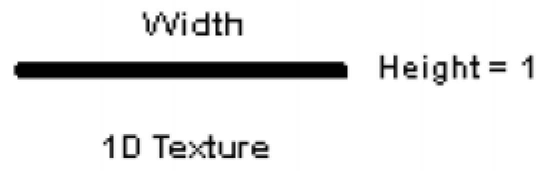
glTexParameteri(**GL_TEXTURE_2D**, …..);

unsigned char* bitmap

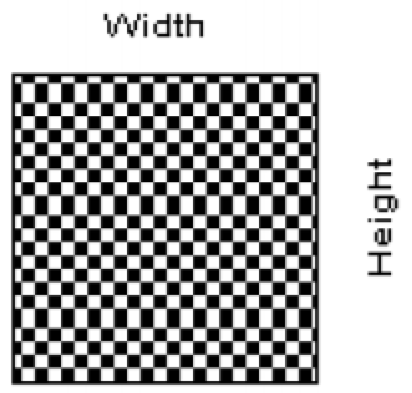
glTexImage2D(**GL_TEXTURE_2D**, ….., bitmap)

Enabling Texture Mapping

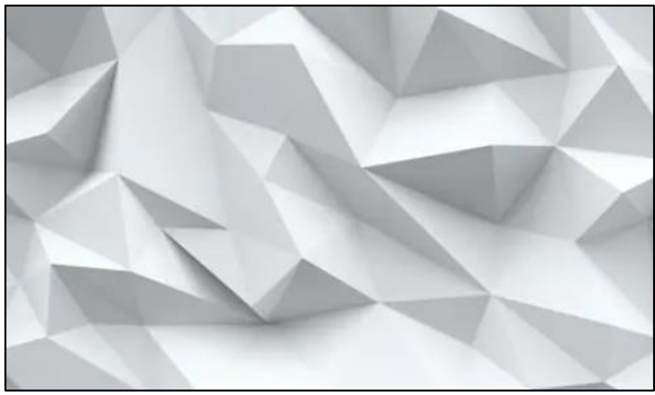
- `glEnable(GL_TEXTURE_*D)`
  - OpenGL supports 1~3 dimensional texture maps:
    - `GL_TEXTURE_1D, GL_TEXTURE_2D, GL_TEXTURE_3D`
  - 3D is only supported by OpenGL 1.2 or above



1D Texture



2D Texture

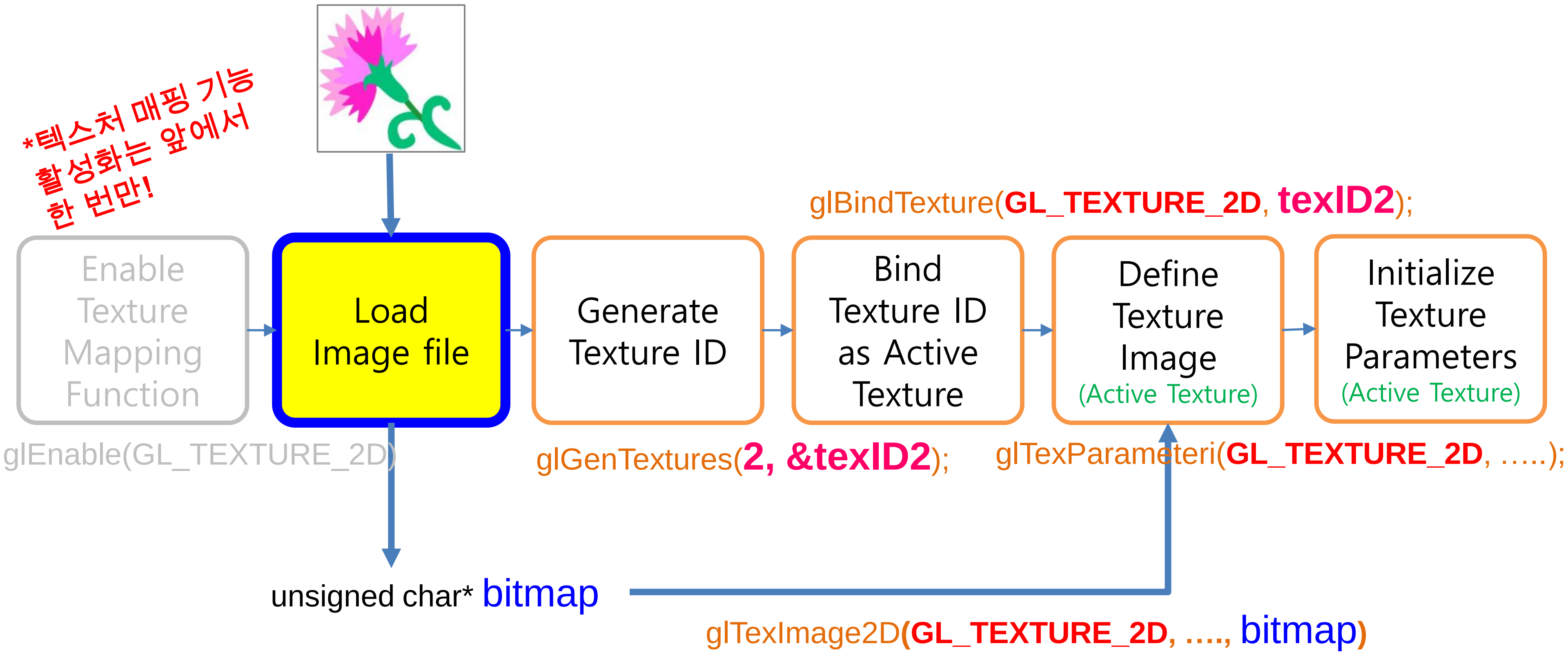


3D Texture

Load Image File

Initializing Texture Mapping

- OpenGL Texture Mapping


*텍스처 매핑 기능
활성화는 앞에서
한 번만!
Enable Texture Mapping Function
Load Image file
Generate Texture ID
Bind Texture ID as Active Texture
Define Texture Image (Active Texture)
Initialize Texture Parameters (Active Texture)
unsigned char* bitmap

```
glEnable(GL_TEXTURE_2D)
glGenTextures(2, &texID2);
glBindTexture(GL_TEXTURE_2D, texID2);
glTexImage2D(GL_TEXTURE_2D, ...., bitmap)
glTexParameteri(GL_TEXTURE_2D, .....);
```

code: load .bmp image file

bmp 파일만 로딩 가능

```
unsigned char* loadBMP(char* fname)
{
    FILE* file = fopen(fname, "rb");        // file open with the option of "read" and "binary"
    if (!file) { cout << "Image file could not be opened " << endl; return NULL; }
    if (fread(header, 1, 54, file) != 54) { // param: buffer, 1 byte, count, file
        cout << "Not a correct BMP file₩n";
        return NULL;
    }
    if (header[0] != 'B' || header[1] != 'M') {  // 2 bytes
        cout << "Not a correct BMP file₩n";
        return NULL;
    }

    // Read ints from the byte array
    dataPos = *(int*)&(header[0x0A]);   // 10
    width = *(int*)&(header[0x12]);     // 18
    height = *(int*)&(header[0x16]);    // 22
    imageSize = *(int*)&(header[0x22]);// 34
    cout << "width = " << width << " height = " << height << endl;
```

offset	
BM	0
Size of BMP file	2
Reserved	6
Reserved	8
Starting address	10
Size of header	14
width	18
height	22
.....	
Image size	34

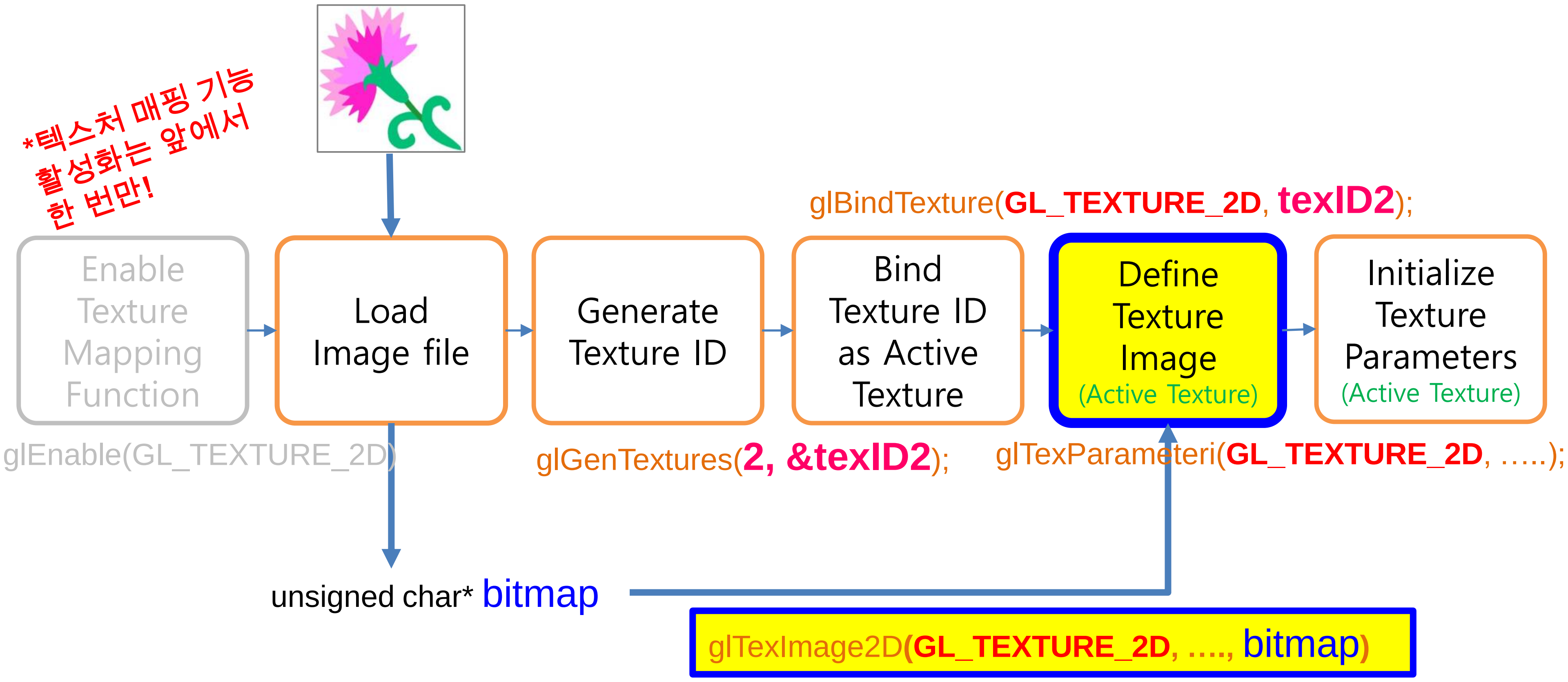
code: load .bmp image file

```
    // Some BMP files are misformatted, guess missing information
    // 3 : one byte for each Red, Green and Blue component
    if (imageSize == 0)    imageSize = width * height * 3;

    // The BMP header is done that way
    if (dataPos == 0)      dataPos = 54;

    // Create a buffer
    unsigned char* bitmap = new unsigned char[imageSize];

    // Read the actual data from the file into the buffer
    fread(bitmap, 1, imageSize, file);

    //Everything is in memory now, the file can be closed
    fclose(file);

    return bitmap;
}
```

Define Texture Image

Initializing Texture Mapping

- OpenGL Texture Mapping


*텍스처 매핑 기능 활성화는 앞에서 한 번만!
Enable Texture Mapping Function
Load Image file
Generate Texture ID
Bind Texture ID as Active Texture
Define Texture Image (Active Texture)
Initialize Texture Parameters (Active Texture)
unsigned char* bitmap

```
glEnable(GL_TEXTURE_2D)
glGenTextures(2, &texID2);
glBindTexture(GL_TEXTURE_2D, texID2);
glTexImage2D(GL_TEXTURE_2D, ...., bitmap)
glTexParameteri(GL_TEXTURE_2D, .....);
```

Defining a Texture

Give the image data for this texture

- `glTexImage2D( target, level, internalFormat, w, h, border, format, type, bitmap );`
  - **`target:`** type of texture, e.g. `GL_TEXTURE_2D`
  - **`level:`** `level-of-detail level,` used for mipmapping (will be discussed later) `It must be 0 if you don't want to use mip-map.`
  - **`internalFormat:`** specifies the number of color components in the texture, e.g. GL_RGB, GL_RGBA
  - **`w, h:`** `image` width and height, image width & height <u>**must be power of 2**</u>
  - **`border:`** specifies the width of the border. (must be either 0 or 1)

    0(no border), 1(user border)
  - **`format, type:`** format and type of the image data

    format(GL_RGB, GL_BGR_EXT, ..) : imput image(bmp)'s color order

    type(GL_UNSIGNED_BYTE) : The data type of the value stored in each color channel of the texture.
  - **`bitmap:`** a pointer to bitmap array, the BMP file you have loaded

- Example:

```
glTexImage2D( GL_TEXTURE_2D, 0, GL_RGB,
    512, 512, 0, GL_BGR_EXT, GL_UNSIGNED_BYTE, bitmap);
```

Defining a Texture

Give the image data for this texture

-
  ```
  glTexImage2D( target, level, internalFormat,
                w, h, border, format, type, bitmap );
  ```

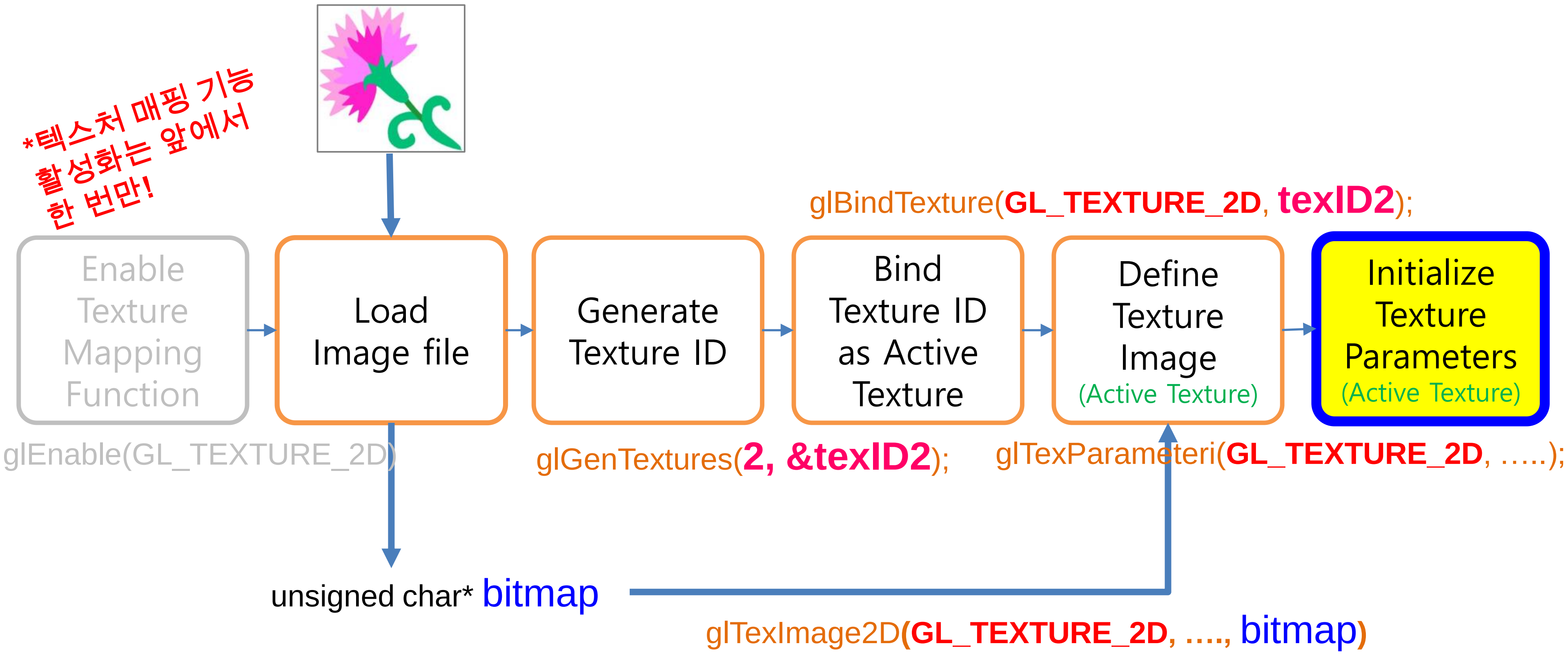
•**텍스처 관련 파라미터** (텍스처 설정):
target, level, internalFormat, border

•**원본 이미지 파일 관련 파라미터** (원본 이미지 데이터 설정):
w, h, format, type, bitmap

Initialize Texture Mapping Parameters

Initializing Texture Mapping

- OpenGL Texture Mapping

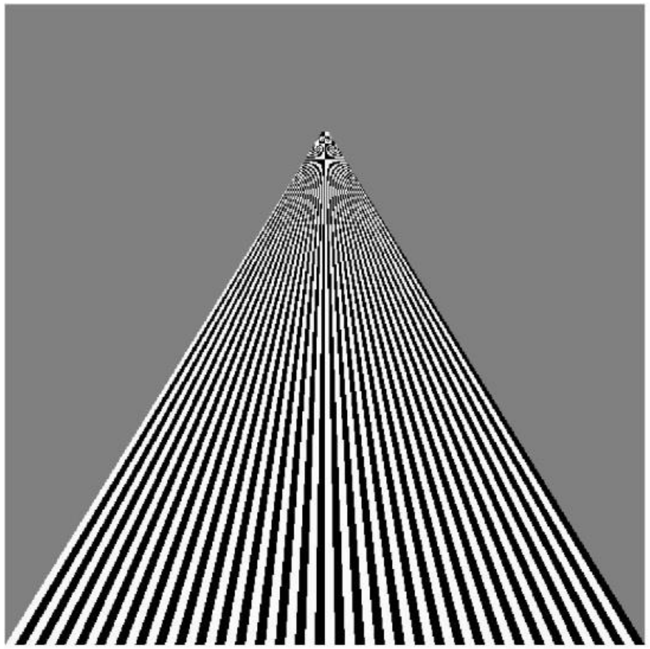

*텍스처 매핑 기능
활성화는 앞에서
한 번만!
Enable Texture Mapping Function
Load Image file
Generate Texture ID
Bind Texture ID as Active Texture
Define Texture Image (Active Texture)
Initialize Texture Parameters (Active Texture)
unsigned char* bitmap

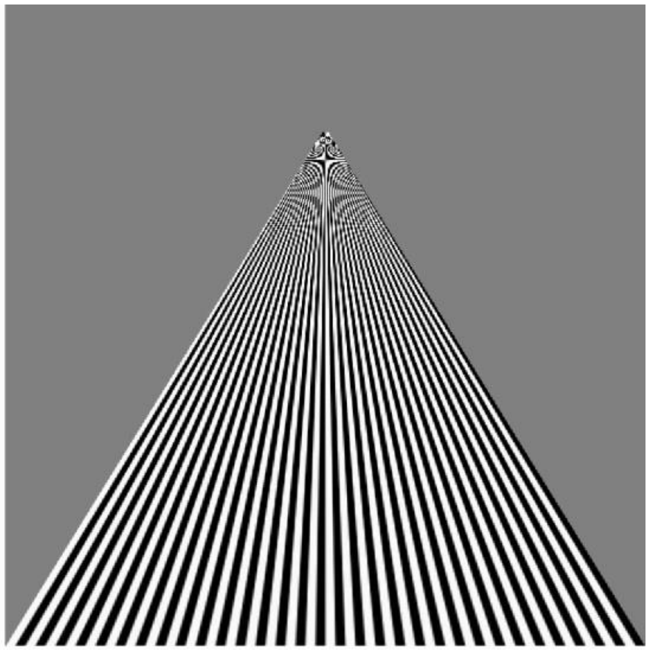
```
glEnable(GL_TEXTURE_2D)
glGenTextures(2, &texID2);
glBindTexture(GL_TEXTURE_2D, texID2);
glTexParameteri(GL_TEXTURE_2D, …..);
glTexImage2D(GL_TEXTURE_2D, …., bitmap)
```

Filter Modes

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

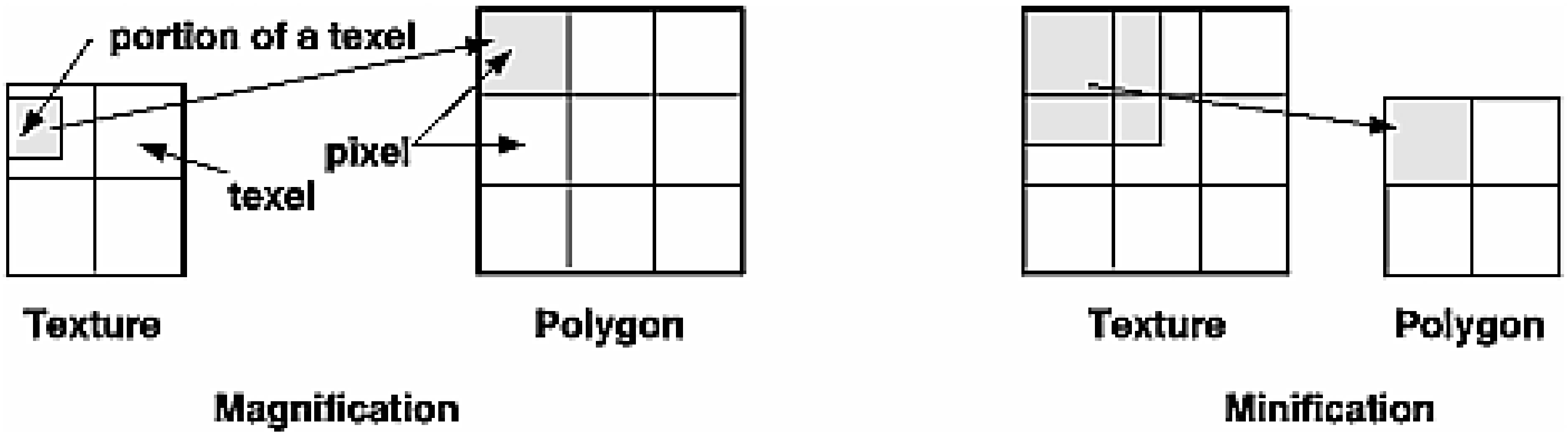
```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```


With GL_NEAREST


With GL_LINEAR

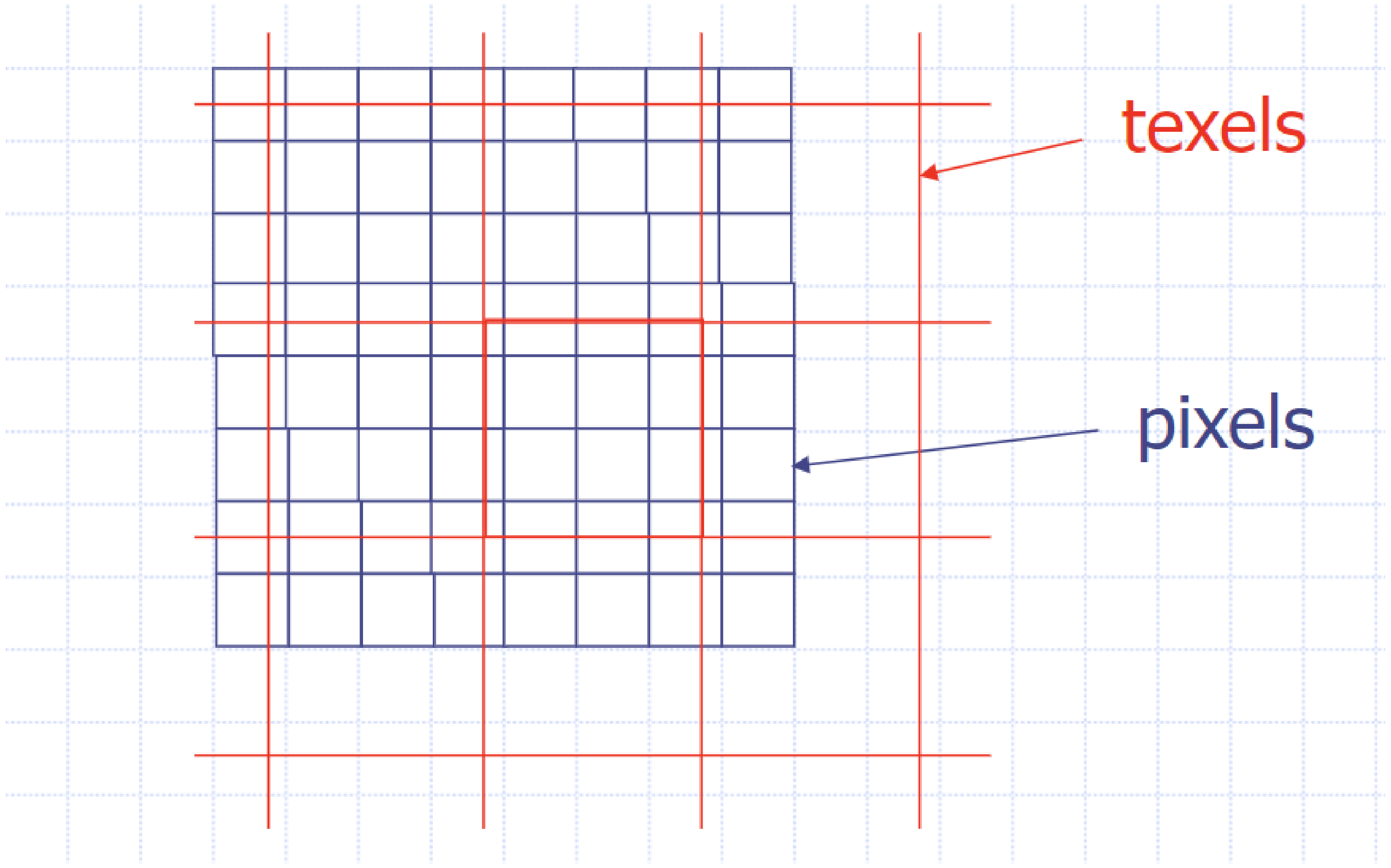
Filter Modes

- A single pixel on the screen can correspond to anything from a tiny portion of a texel (magnification)
- A single pixel on the screen can correspond to a large collection of texels (minification)



Filter Modes

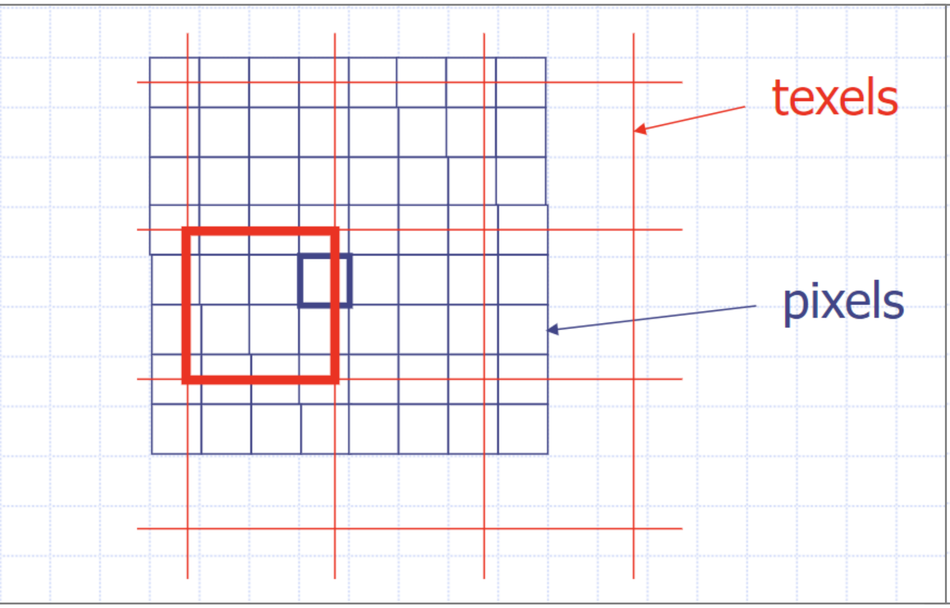
- The alignment is probably not exact.



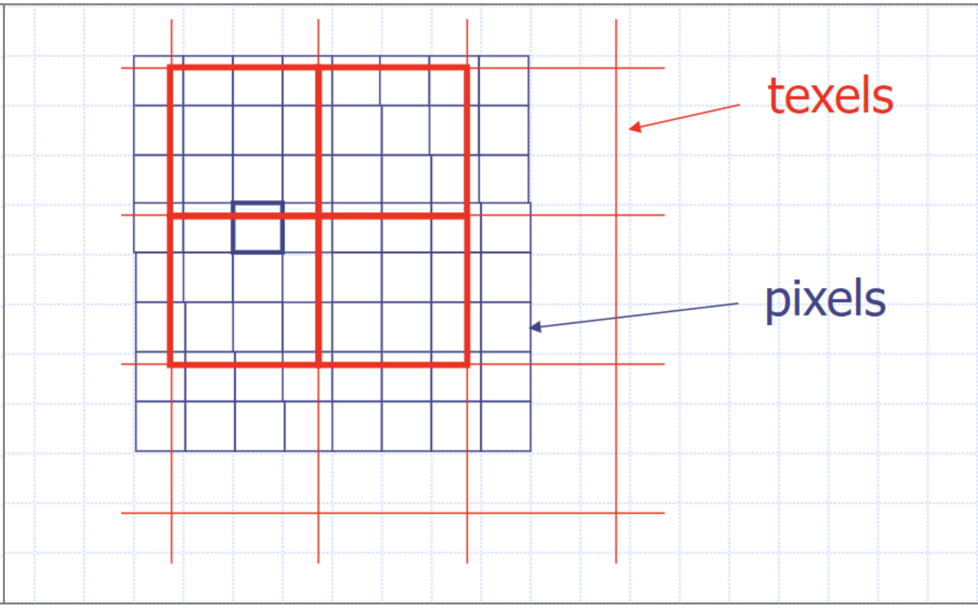
Nearest vs. Interpolation

- Find the nearest texel.
- Interpolate the colors of the nearest four texels.

Find the nearest texel.



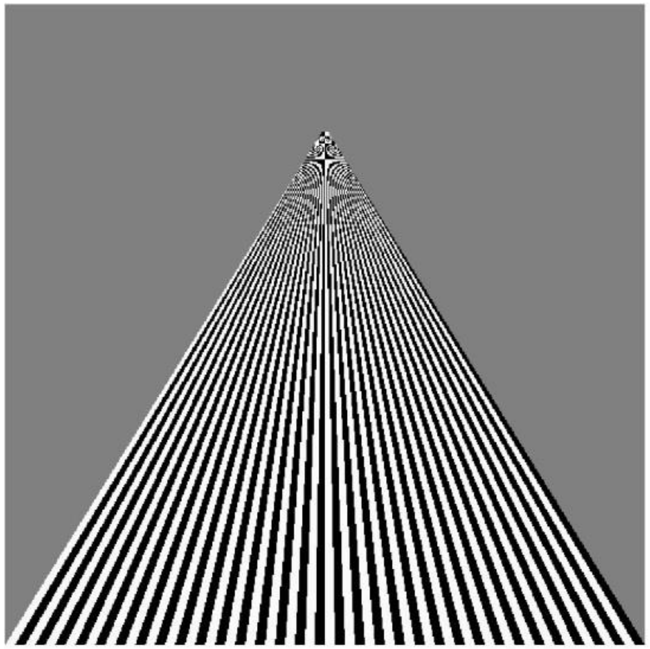
Interpolate the nearest texels

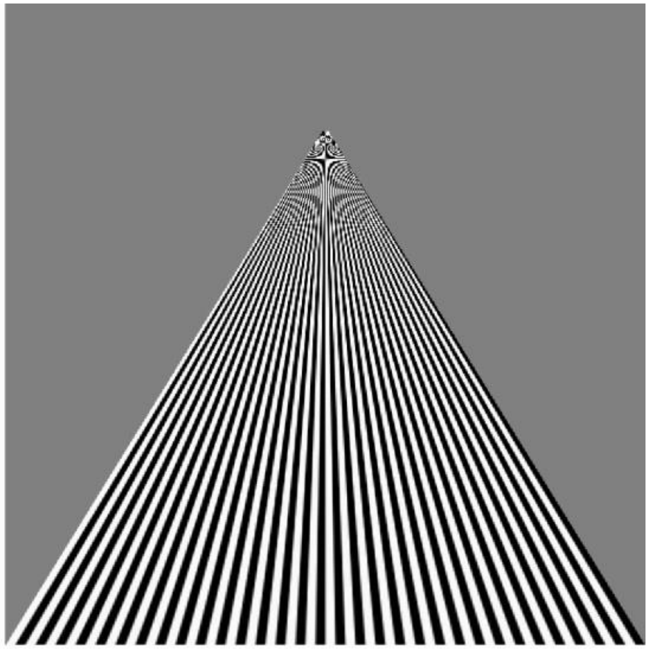


Filter Modes

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

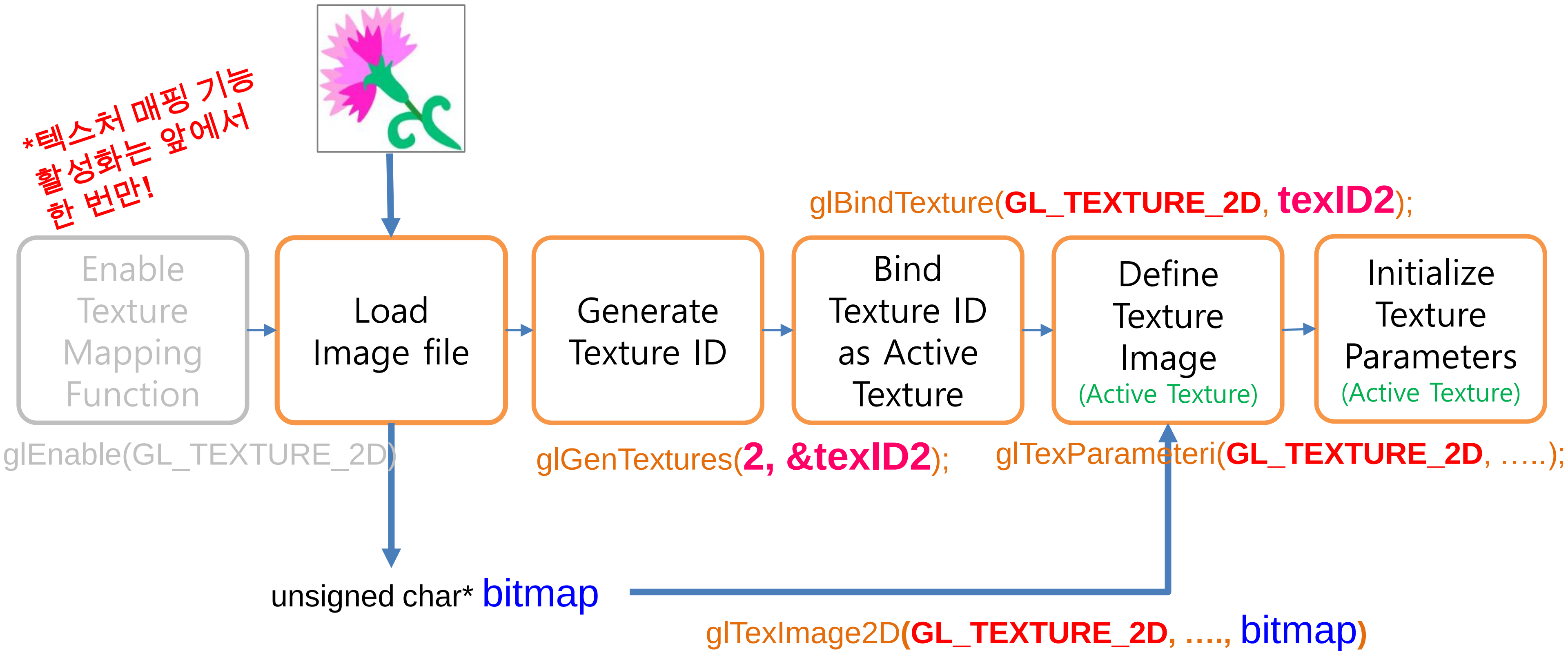
```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```


With GL_NEAREST


With GL_LINEAR

Initializing Texture Mapping

- OpenGL Texture Mapping

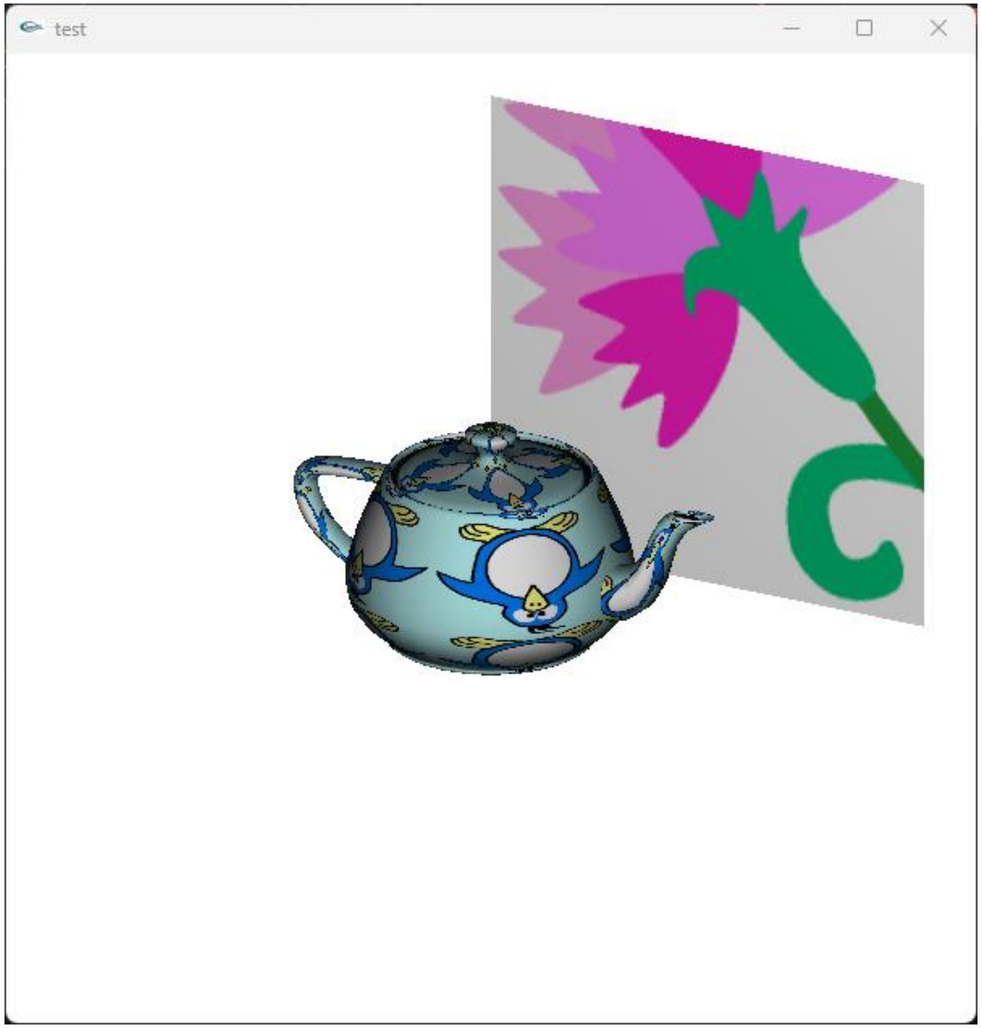


*텍스처 매핑 기능
활성화는 앞에서
한 번만!
Enable Texture Mapping Function
glEnable(GL_TEXTURE_2D)
Load Image file
unsigned char* bitmap
Generate Texture ID
glGenTextures(2, &texID2);
Bind Texture ID as Active Texture
glBindTexture(GL_TEXTURE_2D, texID2);
Define Texture Image (Active Texture)
glTexImage2D(GL_TEXTURE_2D, …., bitmap)
Initialize Texture Parameters (Active Texture)
glTexParameteri(GL_TEXTURE_2D, …..);

OpenGL Texture Mapping Code

Displaying Texture Mapped Objects

- Two textures
  - carnation.bmp
  - penguin.bmp



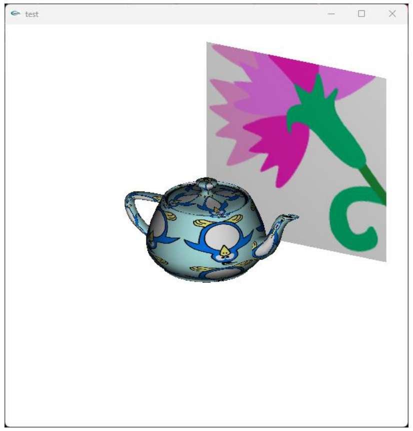
Displaying Texture Mapped Objects

- OpenGL Texture Mapping

glBindTexture(GL_TEXTURE_2D, **texID1**);

$$\begin{tikzcd}\boxed{\text{Bind Texture ID 1}} \arrow[r] & \boxed{\text{Draw Rectangle with Texture Coordinates}}\end{tikzcd}$$

```
glBegin(GL_POLYGON);
   glTexCoord2f(0.0, 0.0);
   glVertex3f(-0.5, -0.5, -1);
   glTexCoord2f(0.0, 0.7);
   glVertex3f(-0.5,  0.5, -1);
   glTexCoord2f(0.7, 0.7);
   glVertex3f( 0.5,  0.5, -1);
   glTexCoord2f(0.7, 0.0);
   glVertex3f( 0.5, -0.5, -1);
glEnd();
```



glBindTexture(GL_TEXTURE_2D, **texID2**);

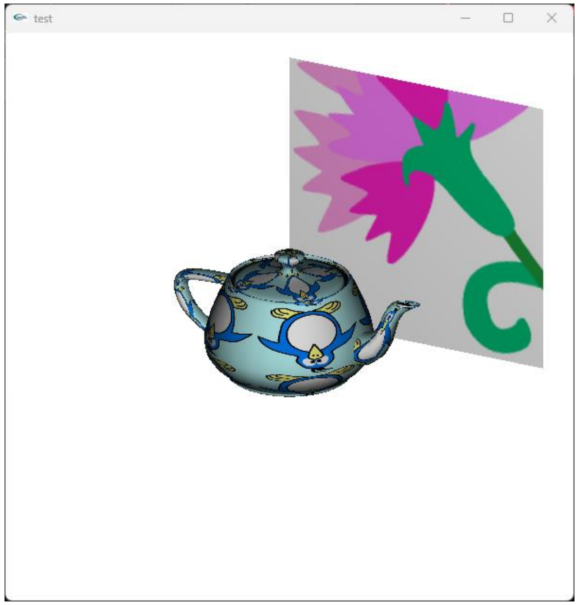
$$\begin{tikzcd}\boxed{\text{Bind Texture ID 2}} \arrow[r] & \boxed{\text{Draw Teapot with Texture Coordinates}}\end{tikzcd}$$

Texture 좌표 줄 필요 없음
Glut 함수 내부에서 처리

```
glutSolidTeapot(0.3);
```

Displaying Texture Mapped Objects

- Texture mapping
  - carnation.bmp
  - penguin.bmp



```
// Teapot
glBindTexture(GL_TEXTURE_2D, texID2);
glutSolidTeapot(0.3);                // penguin

// Rectangle
glBindTexture(GL_TEXTURE_2D, texID1);
glBegin(GL_POLYGON);                 // carnation
        glTexCoord2f(0.0, 0.0); glVertex3f(-0.5, -0.5, -1);
        glTexCoord2f(0.0, 0.7); glVertex3f(-0.5,  0.5, -1);
        glTexCoord2f(0.7, 0.7); glVertex3f( 0.5,  0.5, -1);
        glTexCoord2f(0.7, 0.0); glVertex3f( 0.5, -0.5, -1);
glEnd();
```

Initializing Texture Mapping

```
GLuint texID1, texID2;   // texture ID

int initTexture()
{
      unsigned char* bitmap;

      glEnable(GL_TEXTURE_2D);

      // TEXTURE ID 1
      bitmap = loadBMP((char*)"carnation.bmp");  // Load BMP image file
      if (bitmap == NULL) {
            cout << "file open error" << endl;
            return -1;
      }
      glGenTextures(1, &texID1);                                // Generate texture ID
      glBindTexture(GL_TEXTURE_2D, texID1);          // Bind texture ID
      glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height,      // Define texture image
                         0, GL_BGR_EXT, GL_UNSIGNED_BYTE, bitmap);
      // Initialize texture mapping parameters
      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
      glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
```

Initializing Texture Mapping

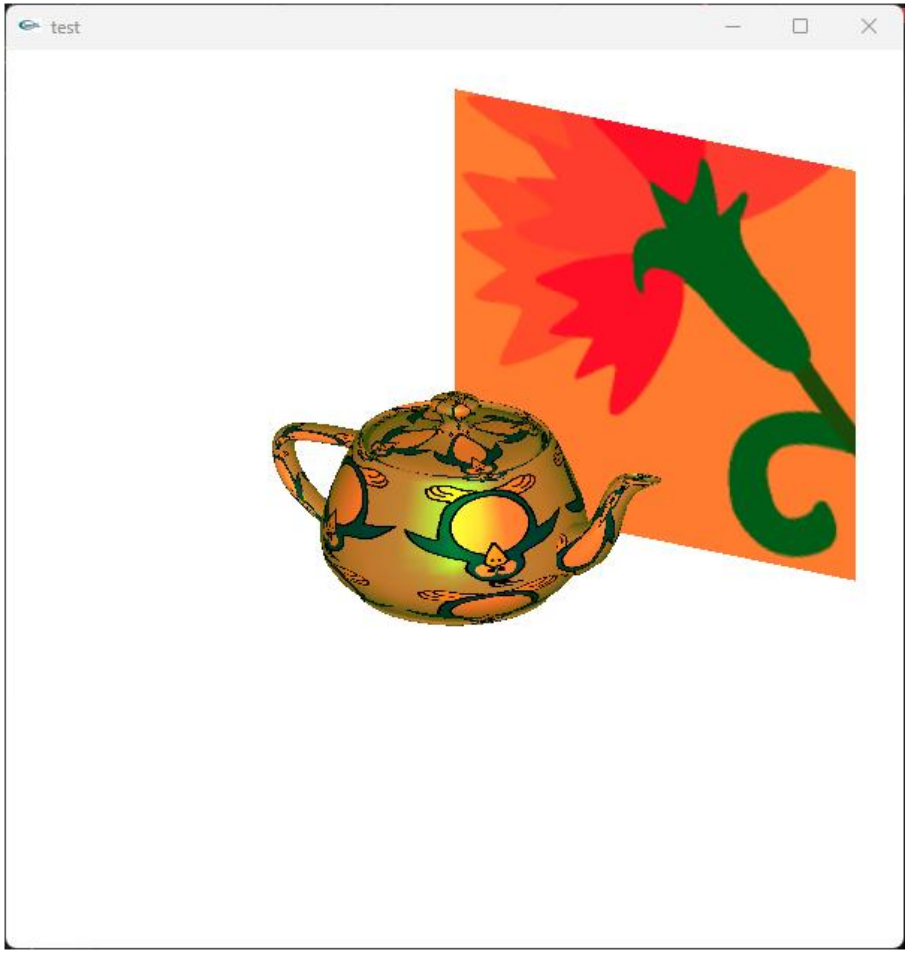
```
    // TEXTURE ID 2
    bitmap = loadBMP((char*)"penguin.bmp");    // Load BMP image file
    if (bitmap == NULL) {
        cout << "file open error" << endl;
        return -1;
    }
    glGenTextures(2, &texID2);                 // Generate texture ID
    glBindTexture(GL_TEXTURE_2D, texID2);   // Bind texture ID
    glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height,     // Define texture image
                      0, GL_BGR_EXT, GL_UNSIGNED_BYTE, bitmap);
    // Initialize texture mapping parameters
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
    return 0;
}

int main(int argc, char** argv)
{
    ….
    initTexture();
    ….
}
```

Texture Mapping & Lighting

Texture Mapping & Lighting

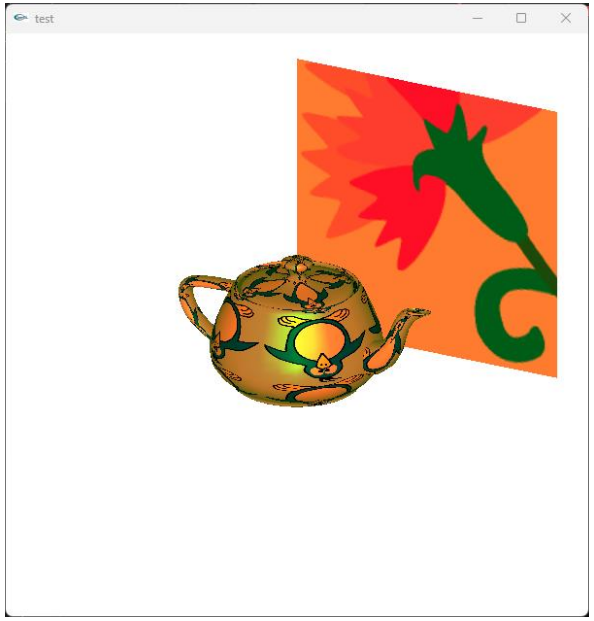
- Texture mapping + Lighting (지난 시간 코드)

```
GLfloat ambient[] = { 0.0, 0.0, 0.0, 1.0 };
GLfloat diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat specular[] = { 1.0, 1.0, 1.0, 1.0 };

GLfloat mat_a[] = { 0.1, 0.1, 0.1, 1.0 };
GLfloat mat_d[] = { 1, 0.2, 0.2, 1.0 };
GLfloat mat_s[] = { 0, 1, 0, 1.0 };
GLfloat low_sh[] = { 50.0 };
GLfloat material_emission[] = { 0.3,0.3,0,1 };

GLfloat position[] = { 0, 0, 2, 1 };
bool dir = true;
GLuint texID1, texID2; // texture ID

void display()
{
        glClear(GL_COLOR_BUFFER_BIT
                | GL_DEPTH_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        // View Volume
        glOrtho(-1, 1, -1, 1, 1, 30);
```

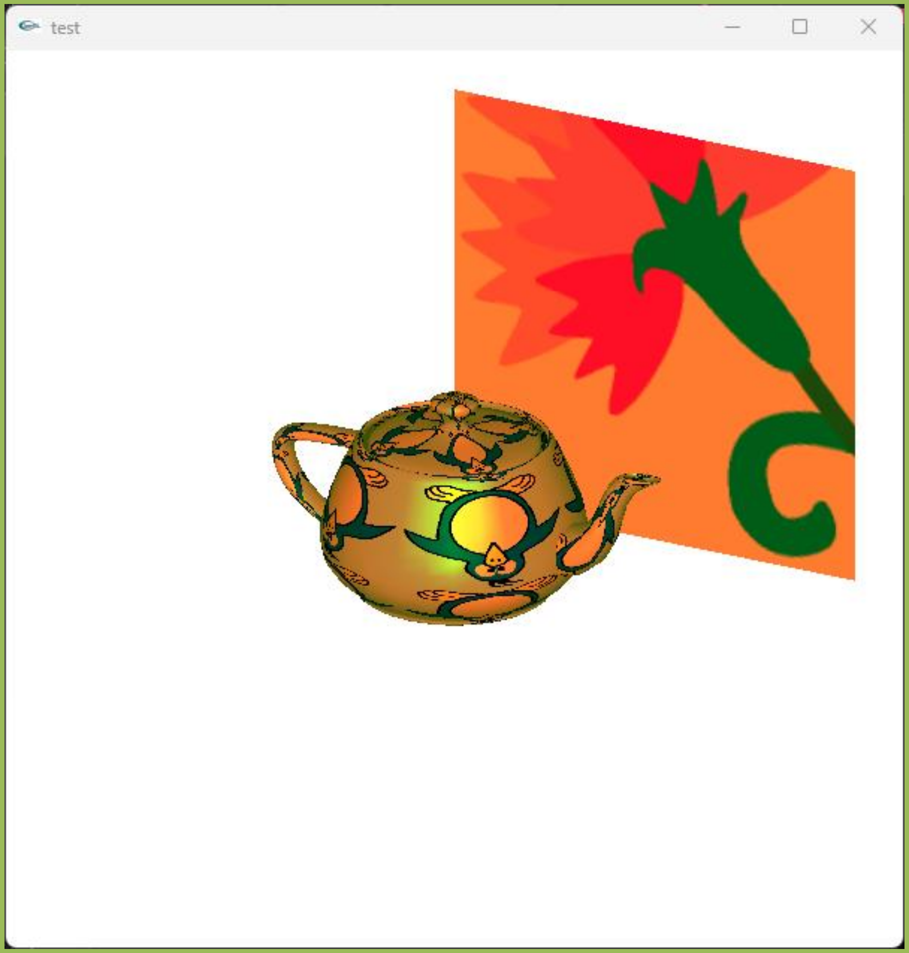


Texture Mapping & Lighting & Material

```
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
// Camera
gluLookAt(1, 1, 2, 0, 0, 0, 0, 1, 0);

// Light
glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
glLightfv(GL_LIGHT0, GL_SPECULAR, specular);

// Material
glMaterialfv(GL_FRONT, GL_AMBIENT, mat_a);
glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_d);
glMaterialfv(GL_FRONT, GL_SPECULAR, mat_s);
glMaterialfv(GL_FRONT, GL_SHININESS, low_sh);
glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION, material_emission);

// Teapot
glBindTexture(GL_TEXTURE_2D, texID2);
glutSolidTeapot(0.3);

// Rectangle
glBindTexture(GL_TEXTURE_2D, texID1);
glBegin(GL_POLYGON);
    glTexCoord2f(0.0, 0.0); glVertex3f(-0.5, -0.5, -1);
    glTexCoord2f(0.0, 0.7); glVertex3f(-0.5,  0.5, -1);
    glTexCoord2f(0.7, 0.7); glVertex3f( 0.5,  0.5, -1);
    glTexCoord2f(0.7, 0.0); glVertex3f( 0.5, -0.5, -1);
glEnd();
glFlush();
}
```



OpenGL Texture Mapping Example Program



Texture Mapping(1)

```
#include <iostream>
#include <math.h>
#include <gl/glut.h>
using namespace std;

#define WIDTH 600
#define HEIGHT 600

// light
GLfloat ambient[] = { 0.0, 0.0, 0.0, 1.0 };
GLfloat diffuse[] = { 1.0, 1.0, 1.0, 1.0 };
GLfloat specular[] = { 1.0, 1.0, 1.0, 1.0 };

// material
GLfloat mat_a[] = { 0.1, 0.1, 0.1, 1.0 };
GLfloat mat_d[] = { 1, 0.2, 0.2, 1.0 };
GLfloat mat_s[] = { 0, 1, 0, 1.0 };
GLfloat low_sh[] = { 50.0 };
GLfloat material_emission[] = { 0.3,0.3,0,1 };

// light position
GLfloat position[] = { 0, 0, 2, 1 };
// texture ID
GLuint texID1, texID2;
```

```
void display()
{
        glClear(GL_COLOR_BUFFER_BIT
                | GL_DEPTH_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        // View Volume
        glOrtho(-1, 1, -1, 1, 1, 30);

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        // Camera
        gluLookAt(1, 1, 2, 0, 0, 0, 0, 1, 0);

        // Light
        glLightfv(GL_LIGHT0, GL_AMBIENT, ambient);
        glLightfv(GL_LIGHT0, GL_DIFFUSE, diffuse);
        glLightfv(GL_LIGHT0, GL_SPECULAR, specular);

        // Material
        glMaterialfv(GL_FRONT, GL_AMBIENT, mat_a);
        glMaterialfv(GL_FRONT, GL_DIFFUSE, mat_d);
        glMaterialfv(GL_FRONT, GL_SPECULAR, mat_s);
        glMaterialfv(GL_FRONT, GL_SHININESS, low_sh);
        glMaterialfv(GL_FRONT_AND_BACK,GL_EMISSION,
                material_emission);
```

Texture Mapping(2)

```
    // Teapot
    glBindTexture(GL_TEXTURE_2D, texID2);
    glutSolidTeapot(0.3);

    // Rectangle
    glBindTexture(GL_TEXTURE_2D, texID1);
    glBegin(GL_POLYGON);
        glTexCoord2f(0.0, 0.0);
        glVertex3f(-0.5, -0.5, -1);
        glTexCoord2f(0.0, 0.7);
        glVertex3f(-0.5,  0.5, -1);
        glTexCoord2f(0.7, 0.7);
        glVertex3f( 0.5,  0.5, -1);
        glTexCoord2f(0.7, 0.0);
        glVertex3f( 0.5, -0.5, -1);
    glEnd();

    glFlush();
}
```

```
// Data read from the header of the BMP file
// Each BMP file begins by a 54-bytes header
unsigned char header[54];
// Position in the file where the actual data begins
unsigned int dataPos;
// image width and height
unsigned int width, height;
// imageSize = width*height*3
unsigned int imageSize;

unsigned char* loadBMP(char* fname)
{
    FILE* file = fopen(fname, "rb");  // file open
    if (!file) {
        cout << "Image file could not be opened "
            << endl; return NULL;
    }
    if (fread(header, 1, 54, file) != 54) {
        // read header: if not 54 bytes read, problem
        cout << "Not a correct BMP file₩n";
        return NULL;
    }
    if (header[0] != 'B' || header[1] != 'M') {  // 2 bytes
        cout << "Not a correct BMP file₩n";
        return NULL;
    }
```

Texture Mapping(3)

```
    // Read ints from the byte array
    dataPos = *(int*)&(header[0x0A]);
    width = *(int*)&(header[0x12]);
    height = *(int*)&(header[0x16]);
    imageSize = *(int*)&(header[0x22]);
    cout << "width = " << width << " height = " << height << endl;

    // Some BMP files are misformatted, guess missing information
    // 3 : one byte for each Red, Green and Blue component
    if (imageSize == 0)     imageSize = width * height * 3;

    // The BMP header is done that way
    if (dataPos == 0)       dataPos = 54;

    // Create a buffer
    unsigned char* bitmap = new unsigned char[imageSize];

    // Read the actual data from the file into the buffer
    fread(bitmap, 1, imageSize, file);

    // Everything is in memory now, the file can be closed
    fclose(file);

    return bitmap;
}
```

Texture Mapping(4)

```
int initTexture()
{
	unsigned char* bitmap;

	glEnable(GL_TEXTURE_2D);

	bitmap = loadBMP((char*)"carnation.bmp");
	if (bitmap == NULL) {
		cout << "file open error" << endl;
		return -1;
	}
	cout << "width=" << width << " height=" << height << endl;

	glGenTextures(1, &texID1);
	glBindTexture(GL_TEXTURE_2D, texID1);
	glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height,
		0, GL_BGR_EXT, GL_UNSIGNED_BYTE, bitmap);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);

	bitmap = loadBMP((char*)"penguin.bmp");
	if (bitmap == NULL) {
		cout << "file open error" << endl;
		return -1;
	}
	glGenTextures(2, &texID2);
	glBindTexture(GL_TEXTURE_2D, texID2);
	glTexImage2D(GL_TEXTURE_2D, 0, GL_RGB, width, height,
		0, GL_BGR_EXT, GL_UNSIGNED_BYTE, bitmap);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_LINEAR);
	glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_LINEAR);
	return 0;
}
```

Texture Mapping(5)

```
int main(int argc, char** argv)
{
	glutInit(&argc, argv);
	glutInitDisplayMode(GLUT_RGBA | GLUT_DEPTH | GLUT_SINGLE);
	glutInitWindowPosition(100, 100);
	glutInitWindowSize(WIDTH, HEIGHT);
	glutCreateWindow("test");

	glClearColor(1, 1, 1, 0);
	glEnable(GL_DEPTH_TEST);
	glEnable(GL_LIGHTING);
	glEnable(GL_LIGHT0);

	initTexture();

	glutDisplayFunc(display);
	glutMainLoop();
	return 0;
}
```

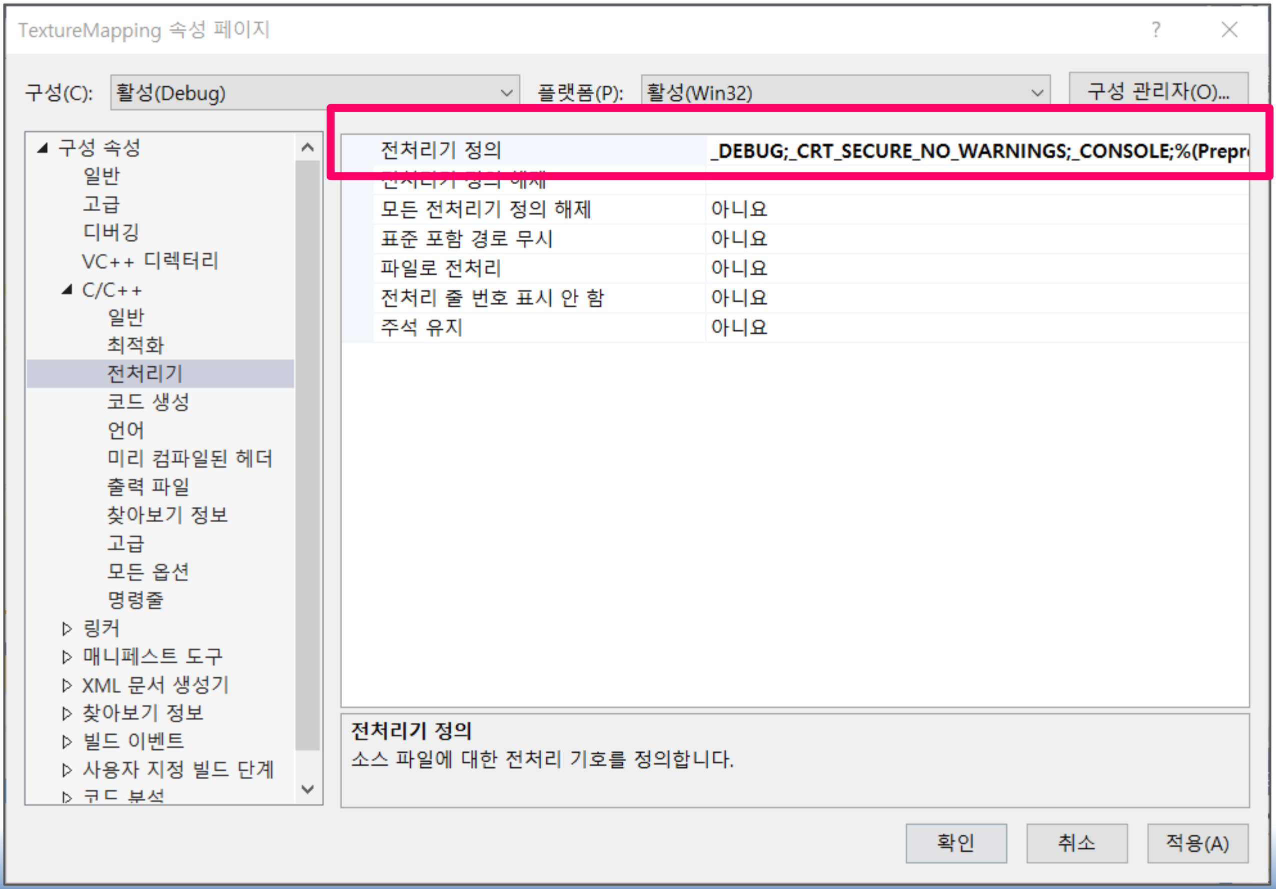
Compile Error

- fopen

1>D:\Code\TextureMapping\TextureMapping\texture.cpp(29,1): error C4996: 'fopen': This function or variable may be unsafe. Consider using fopen_s instead. **To disable deprecation, use _CRT_SECURE_NO_WARNINGS**. See online help for details.

- Fix error !

>상단메뉴
>프로젝트
>속성



TextureMapping 속성 페이지

구성(C): 활성(Debug) 플랫폼(P): 활성(Win32) 구성 관리자(O)...

전처리기 정의	_DEBUG;_CRT_SECURE_NO_WARNINGS;_CONSOLE;%(Prepr
전처리기 정의 해제	
모든 전처리기 정의 해제	아니요
표준 포함 경로 무시	아니요
파일로 전처리	아니요
전처리 줄 번호 표시 안 함	아니요
주석 유지	아니요

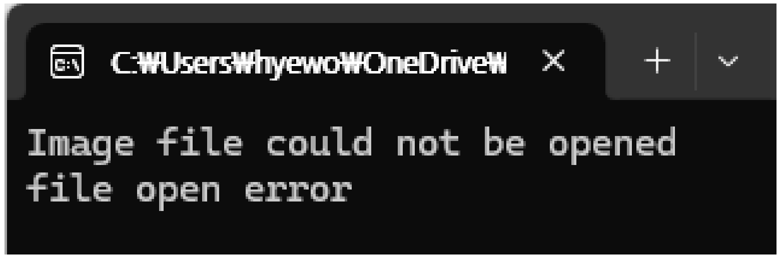
전처리기 정의
소스 파일에 대한 전처리 기호를 정의합니다.

Image File Error

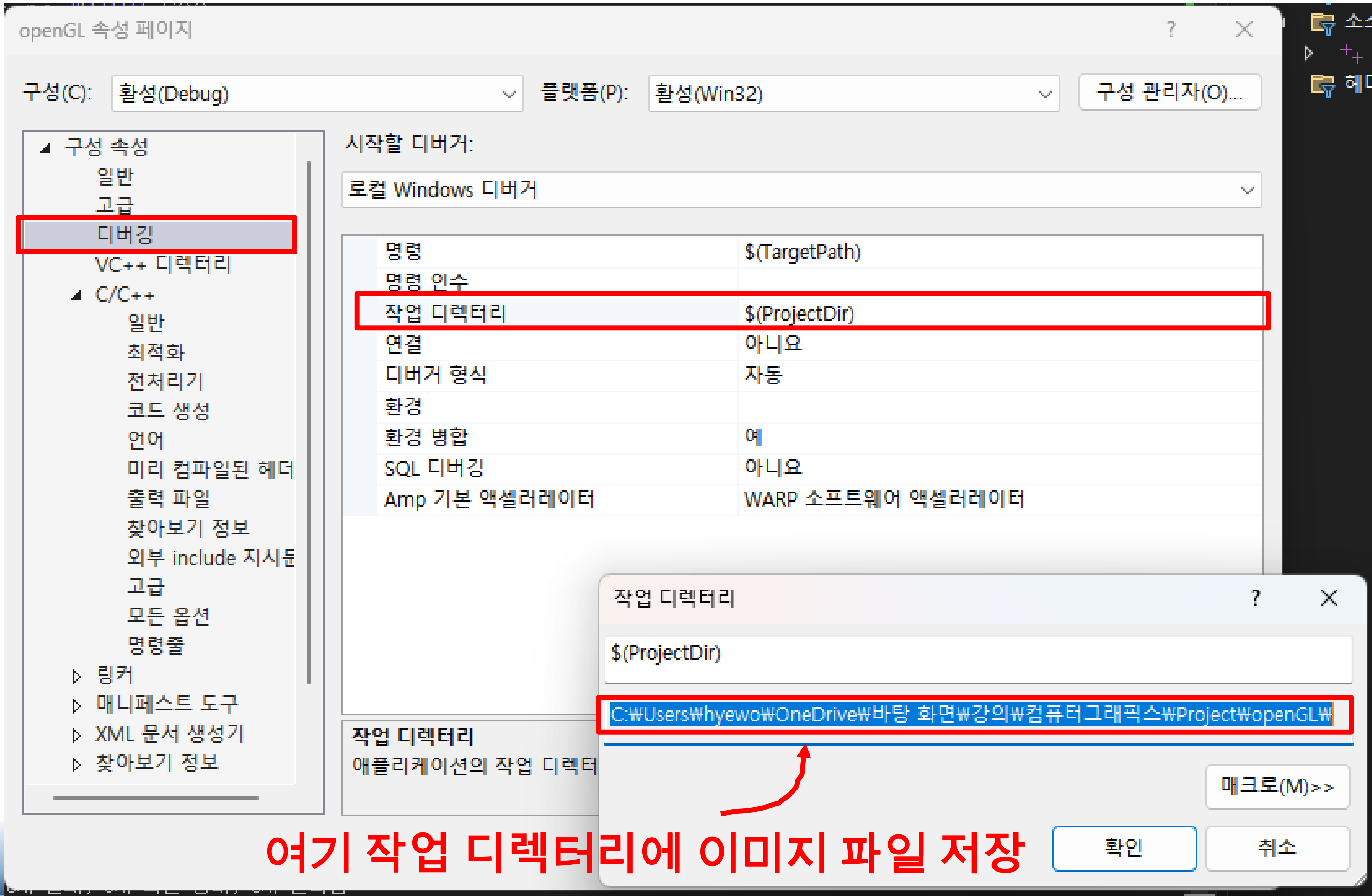
- 작업 디렉토리에 이미지 파일 저장

- Fix error !

>상단메뉴
>프로젝트
>속성



```
C:₩Users₩hyewo₩OneDrive₩
Image file could not be opened
file open error
```



openGL 속성 페이지

구성(C): 활성(Debug) 플랫폼(P): 활성(Win32) 구성 관리자(O)...

시작할 디버거: 로컬 Windows 디버거

명령	$(TargetPath)
명령 인수	
작업 디렉터리	$(ProjectDir)
연결	아니요
디버거 형식	자동
환경	
환경 병합	예
SQL 디버깅	아니요
Amp 기본 액셀러레이터	WARP 소프트웨어 액셀러레이터

작업 디렉터리
애플리케이션의 작업 디렉터

작업 디렉터리
$(ProjectDir)
C:₩Users₩hyewo₩OneDrive₩바탕 화면₩강의₩컴퓨터그래픽스₩Project₩openGL₩

여기 작업 디렉터리에 이미지 파일 저장

Thank you !

학생 여러분,
한 학기 동안 공부하느라 고생 많았습니다.
재미있는 신나는 컴퓨터 그래픽스 공부였기를…

Loading BMP image file