OpenGL Homework 4

CS 550000 Computer Graphics
May 17, 2017
CGV Lab, NTHUCS

Outline

- Goal
- Demo
- Grading
- Submission
- Reminders

Outline

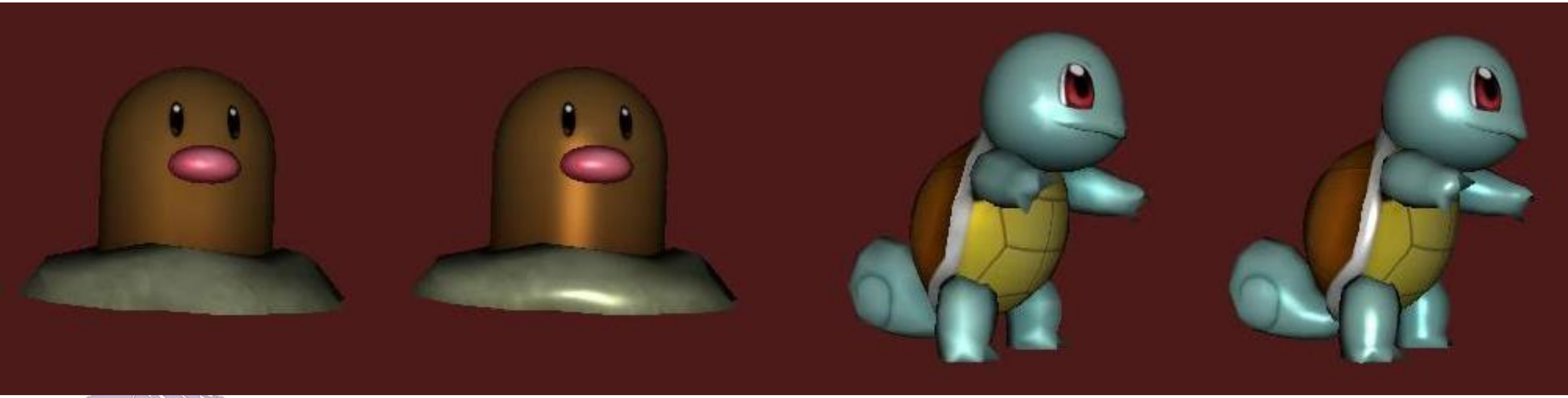
- Goal
- Demo
- Grading
- Submission
- Reminders

Goal (1/2)

- Render a texture-mapped 3D model properly on the screen.

Goal (2/2)

- Render a perpixel-lighted 3D model properly on the screen.



Goal

- Step by step

  1) Start from your previous project.

  2) Download texture models (*.obj, *.mtl, *.bmp).

  3) Ask questions on iLMS.

  4) Finish texture mapping.

  5) Finish per-pixel lighting.

(optional) Find other interesting textured model

Learn to parse the model/material data into the format which can be read by your project.

Outline

- Goal
- Demo
- Grading
- Submission
- Reminders

DEMO

Outline

- Goal
- Demo
- Grading
- Submission
- Reminders

Grading (100)

- Render Texture-Mapped Model (50)

- Correct texture binding (20)

(should be able to display or switch different models)

- Texture filter mode (10)

{magnify, minify} x {linear, nearest}

- Texture wrap mode (10)

{wrap_s, wrap_t} x {clamp, repeat}

- Mipmap (10)

- Lighting (30)

- Vertex shader lighting (10)

Directional, point, and spot light

- Perpixel Lighting (20)

Directional, point, and spot light

Callback Function (10)

- Mouse & keyboard

‘h’ for help menu

Report (10)

- Express your work.

* How to operate your program

* Implementation and problems

* Other efforts you have done

* Screenshots

Keyboard Look-Up Table

‘h’	– show help menu	
‘z’ / ‘Z’	– change to previous model	
‘x’ / ‘X’	– change to next model	
‘t’ / ‘T’	– trigger texture mapping	ON / OFF
‘M’	– switch MAG_FILTER between GL_LINEAR / GL_NEAREST	
‘m’	– switch MIN_FILTER between GL_LINEAR / GL_NEAREST	
‘w’ / ‘W’	– switch TEXTURE_WRAP between GL_REPEAT / GL_CLAMP_TO_EDGE	
‘1’	– trigger Directional light	ON / OFF
‘2’	– trigger Point light	ON / OFF
‘3’	– trigger Spot light	ON / OFF
‘v’	– switch vertex / fragment shader lighting	

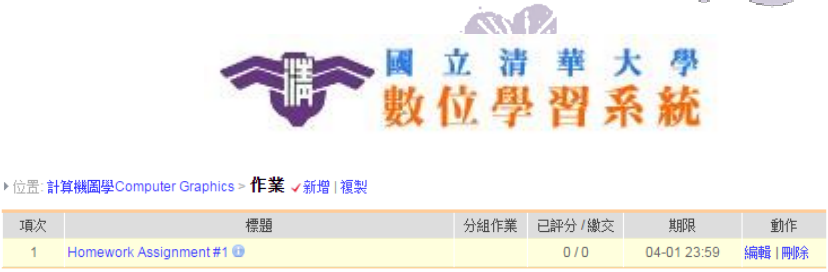
Drag left button	– Transform model
‘R’ / ‘r’	– Auto rotate model around y-axis

Outline

- Goal
- Demo
- Grading
- Submission
- Reminders

Submission

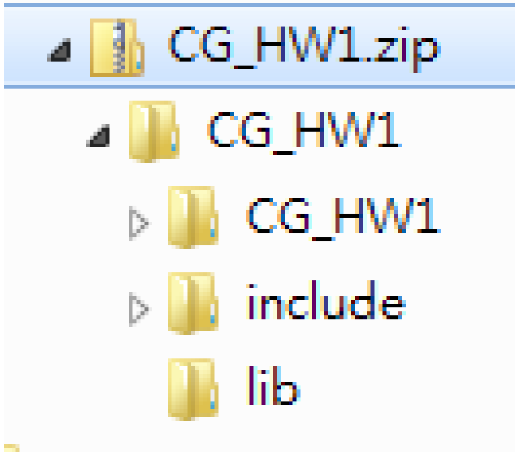
- Due date: May 31th, 2017
- Submit your project to iLMS.
- Filename: HW4_XXXXXXXXX.zip
- Put both "lib" and "include" folder in your zip file



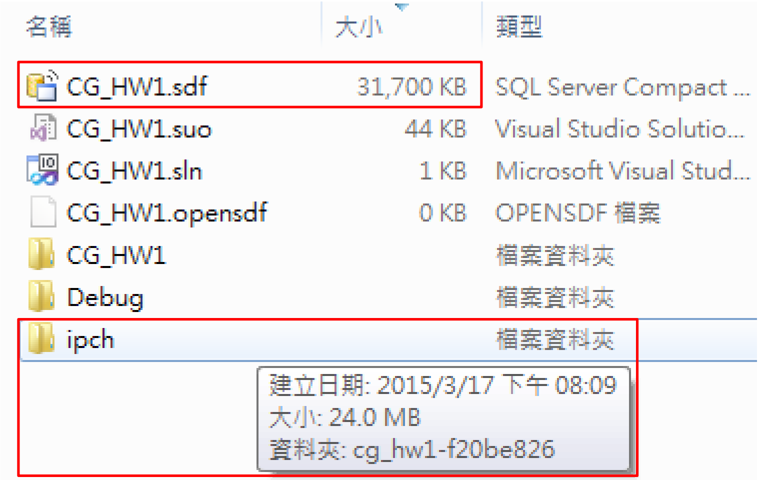
國立清華大學
數位學習系統

位置: 計算機圖學Computer Graphics > 作業 ✓新增 | 複製

項次	標題	分組作業	已評分 / 繳交	期限	動作
1	Homework Assignment #1		0 / 0	04-01 23:59	編輯 | 刪除



CG_HW1.zip
- CG_HW1
  - CG_HW1
  - include
  - lib



名稱	大小	類型
CG_HW1.sdf	31,700 KB	SQL Server Compact ...
CG_HW1.suo	44 KB	Visual Studio Solutio...
CG_HW1.sln	1 KB	Microsoft Visual Stud...
CG_HW1.opensdf	0 KB	OPENSDF 檔案
CG_HW1		檔案資料夾
Debug		檔案資料夾
ipch		檔案資料夾

建立日期: 2015/3/17 下午 08:09
大小: 24.0 MB
資料夾: cg_hw1-f20be826

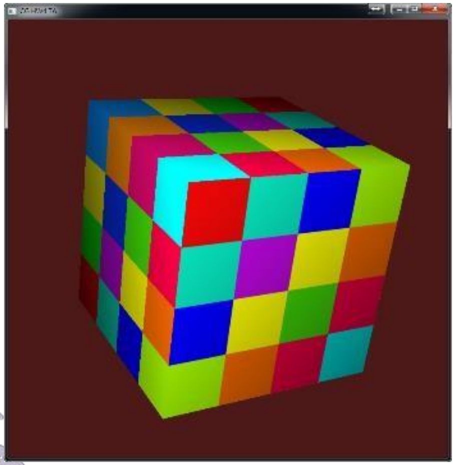
*** Remove "ipch" folder and ".sdf" file. ***

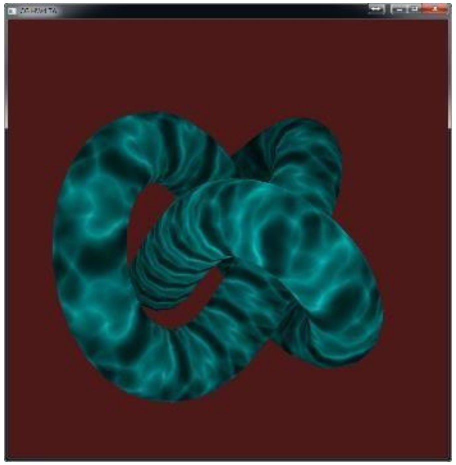
Outline

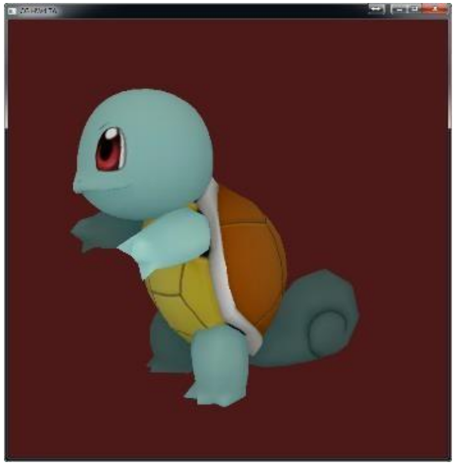
- Goal
- Demo
- Grading
- Submission
- Reminders

Reminders

- You MUST use these critical cases:
  - cube.obj, texturedknot.obj, Zenigame.obj







- Late submission is accepted, DO NOT give up!
- Ask and share information through iLMS

API for Textures

GL_CLAMP GL_TRUE

GL_GENERATE_MIPMAP GL_TEXTURE_MAG_FILTER

GL_TEXTURE_WRAP_S glTexImage2D() glEnable() glBindTexture()

GL_RGB GL_TEXTURE_2D GL_REPEAT

GL_LINEAR GL_NEAREST

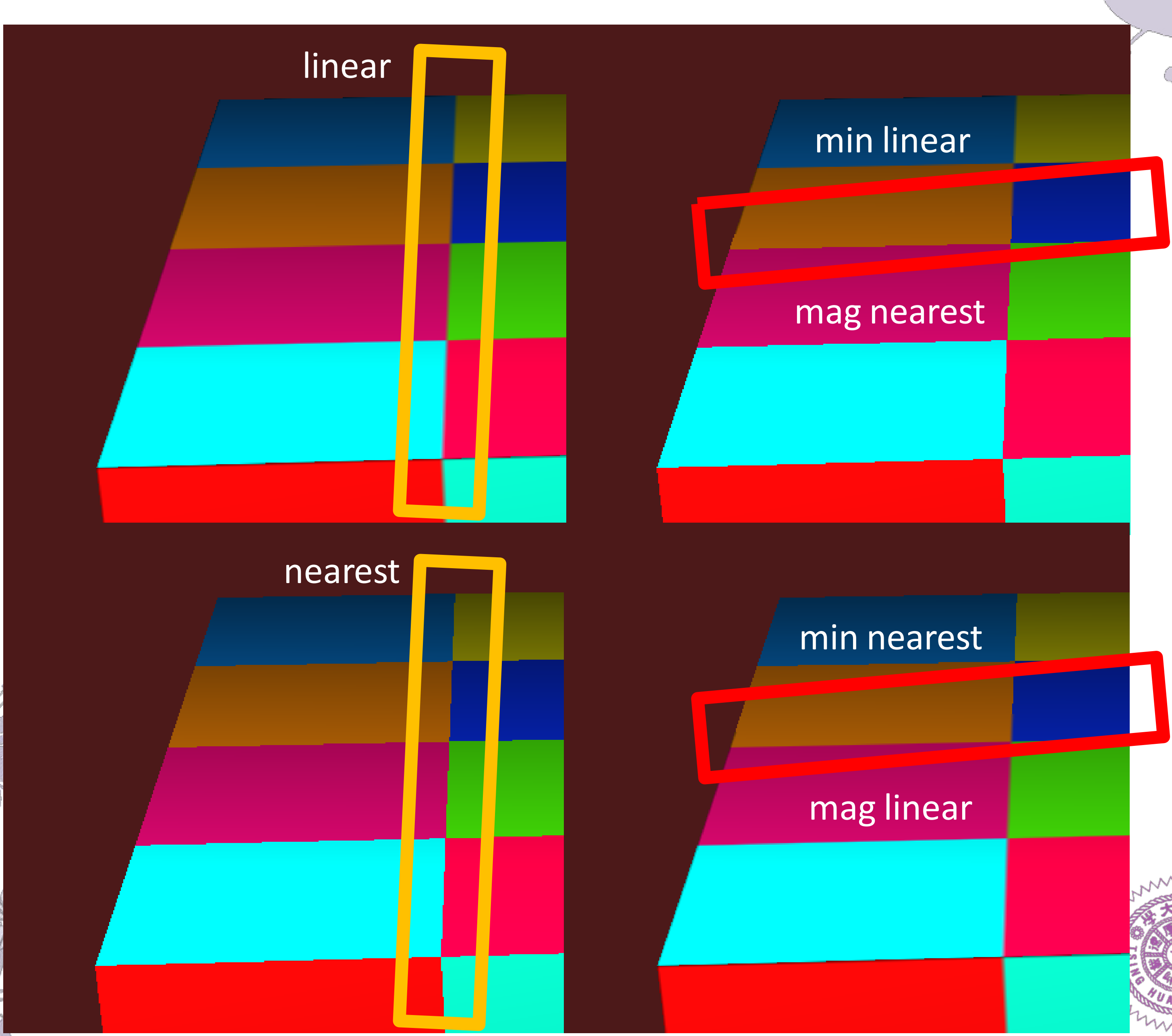
GL_TEXTURE_MAX_LEVEL GL_UNSIGNED_BYTE

glTexParameteri() glGenTextures()

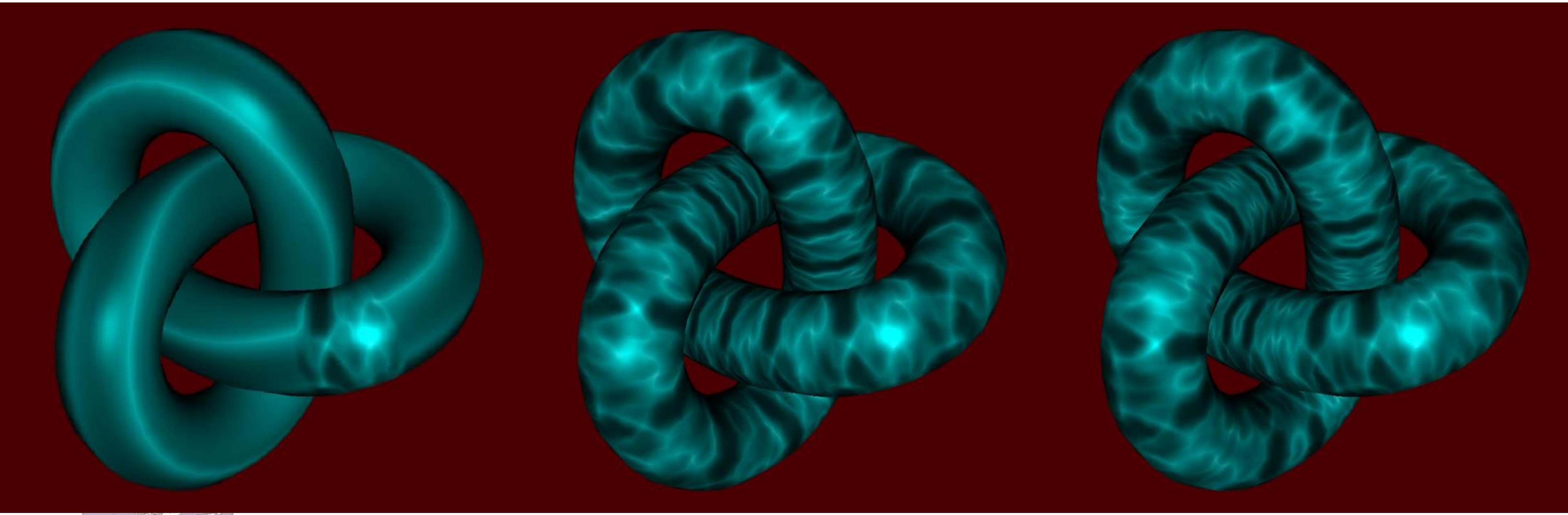
GL_FLOAT GL_TEXTURE_WRAP_T

glEnableVertexAttribArray()

Linear & Nearest Filter



Clamp & Repeat Wrapping

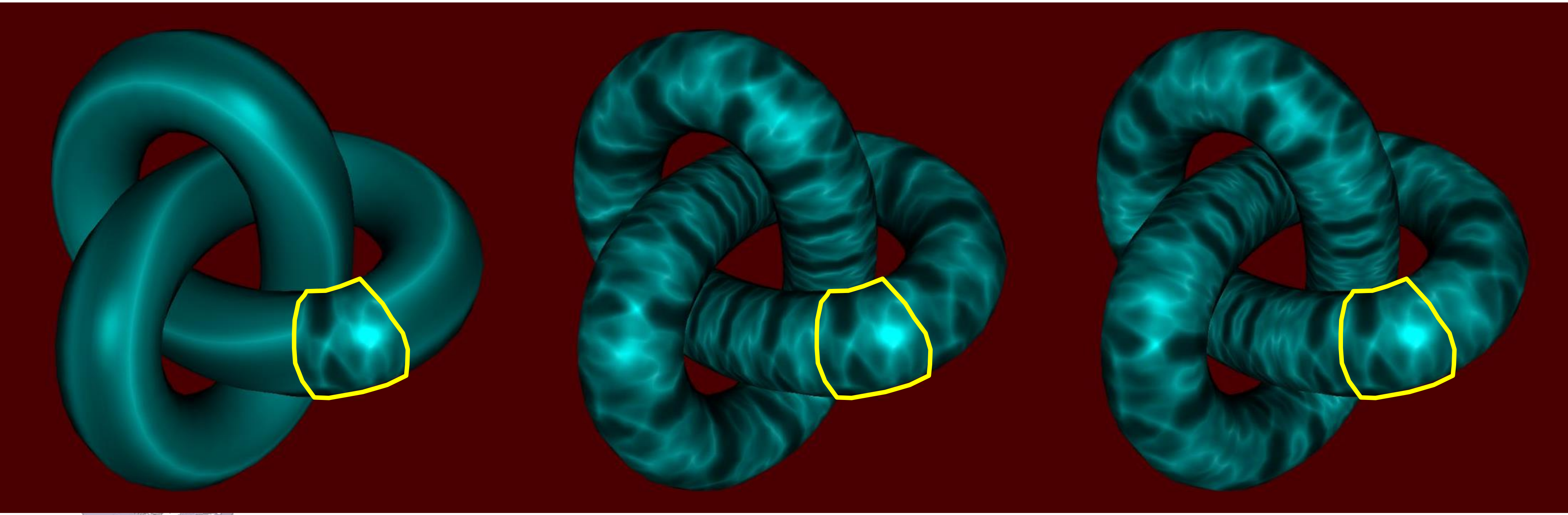


GL_CLAMP_TO_EDGE

GL_REPEAT

GL_MIRRORED_REPEAT

Clamp & Repeat Wrapping

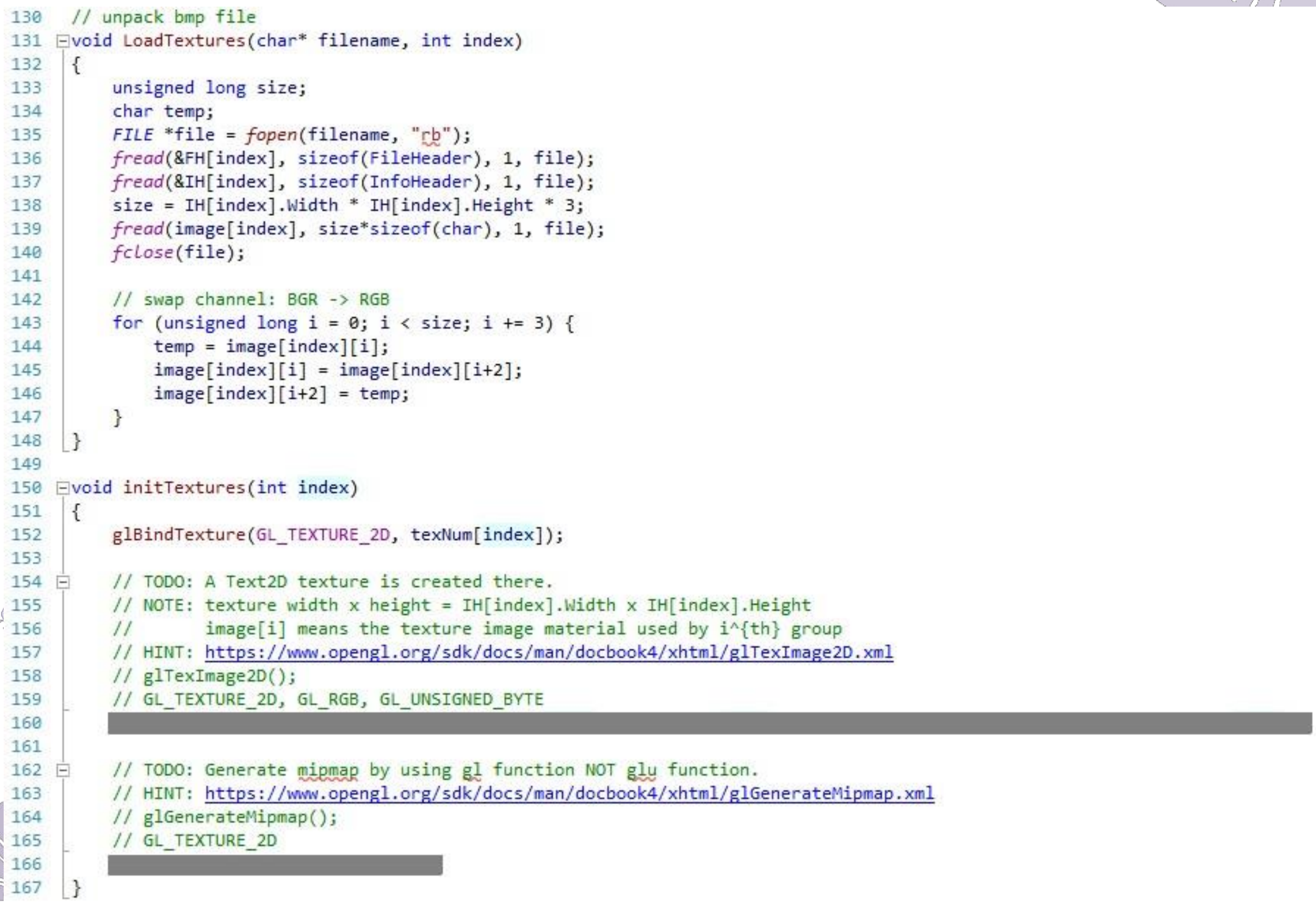


GL_CLAMP_TO_EDGE

GL_REPEAT

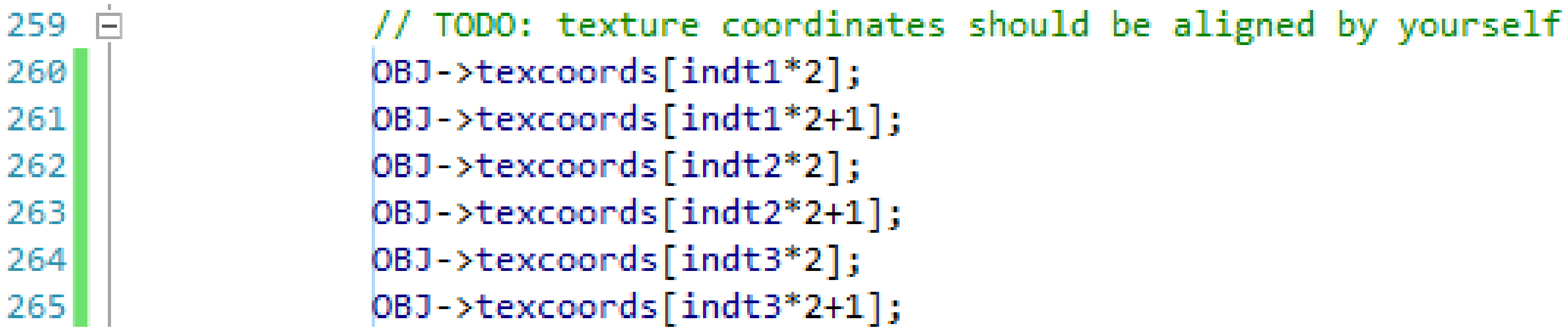
GL_MIRRORED_REPEAT

TODOs



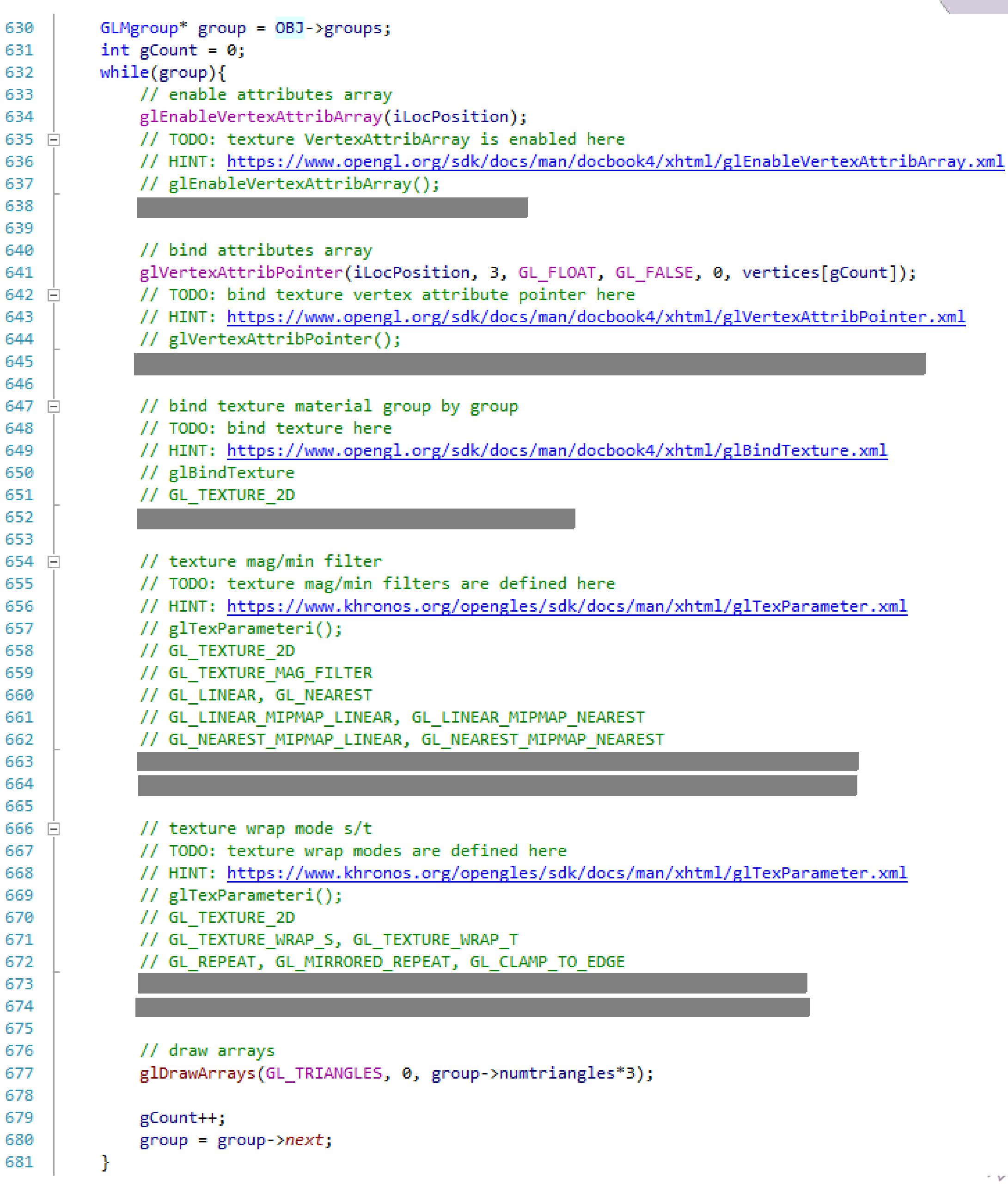
```
// unpack bmp file
void LoadTextures(char* filename, int index)
{
    unsigned long size;
    char temp;
    FILE *file = fopen(filename, "rb");
    fread(&FH[index], sizeof(FileHeader), 1, file);
    fread(&IH[index], sizeof(InfoHeader), 1, file);
    size = IH[index].Width * IH[index].Height * 3;
    fread(image[index], size*sizeof(char), 1, file);
    fclose(file);

    // swap channel: BGR -> RGB
    for (unsigned long i = 0; i < size; i += 3) {
        temp = image[index][i];
        image[index][i] = image[index][i+2];
        image[index][i+2] = temp;
    }
}

void initTextures(int index)
{
    glBindTexture(GL_TEXTURE_2D, texNum[index]);

    // TODO: A Text2D texture is created there.
    // NOTE: texture width x height = IH[index].Width x IH[index].Height
    //       image[i] means the texture image material used by i^{th} group
    // HINT: https://www.opengl.org/sdk/docs/man/docbook4/xhtml/glTexImage2D.xml
    // glTexImage2D();
    // GL_TEXTURE_2D, GL_RGB, GL_UNSIGNED_BYTE
    [redacted]

    // TODO: Generate mipmap by using gl function NOT glu function.
    // HINT: https://www.opengl.org/sdk/docs/man/docbook4/xhtml/glGenerateMipmap.xml
    // glGenerateMipmap();
    // GL_TEXTURE_2D
    [redacted]
}
```

TODOs



```
// TODO: texture coordinates should be aligned by yourself
OBJ->texcoords[indt1*2];
OBJ->texcoords[indt1*2+1];
OBJ->texcoords[indt2*2];
OBJ->texcoords[indt2*2+1];
OBJ->texcoords[indt3*2];
OBJ->texcoords[indt3*2+1];
```

TODOs



```
GLMgroup* group = OBJ->groups;
int gCount = 0;
while(group){
    // enable attributes array
    glEnableVertexAttribArray(iLocPosition);
    // TODO: texture VertexAttribArray is enabled here
    // HINT: https://www.opengl.org/sdk/docs/man/docbook4/xhtml/glEnableVertexAttribArray.xml
    // glEnableVertexAttribArray();
    [redacted]

    // bind attributes array
    glVertexAttribPointer(iLocPosition, 3, GL_FLOAT, GL_FALSE, 0, vertices[gCount]);
    // TODO: bind texture vertex attribute pointer here
    // HINT: https://www.opengl.org/sdk/docs/man/docbook4/xhtml/glVertexAttribPointer.xml
    // glVertexAttribPointer();
    [redacted]

    // bind texture material group by group
    // TODO: bind texture here
    // HINT: https://www.opengl.org/sdk/docs/man/docbook4/xhtml/glBindTexture.xml
    // glBindTexture
    // GL_TEXTURE_2D
    [redacted]

    // texture mag/min filter
    // TODO: texture mag/min filters are defined here
    // HINT: https://www.khronos.org/opengles/sdk/docs/man/xhtml/glTexParameter.xml
    // glTexParameteri();
    // GL_TEXTURE_2D
    // GL_TEXTURE_MAG_FILTER
    // GL_LINEAR, GL_NEAREST
    // GL_LINEAR_MIPMAP_LINEAR, GL_LINEAR_MIPMAP_NEAREST
    // GL_NEAREST_MIPMAP_LINEAR, GL_NEAREST_MIPMAP_NEAREST
    [redacted]

    // texture wrap mode s/t
    // TODO: texture wrap modes are defined here
    // HINT: https://www.khronos.org/opengles/sdk/docs/man/xhtml/glTexParameter.xml
    // glTexParameteri();
    // GL_TEXTURE_2D
    // GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T
    // GL_REPEAT, GL_MIRRORED_REPEAT, GL_CLAMP_TO_EDGE
    [redacted]

    // draw arrays
    glDrawArrays(GL_TRIANGLES, 0, group->numtriangles*3);

    gCount++;
    group = group->next;
}
```

Enable vertex attribute array

Assign pointer

Bind texture id

Set texture filter parameter

Set texture wrap parameter

Fragment Shader

```
varying vec2 vv2texCoord;
varying vec4 vv4colour, vv4vertex;

void main()
{
    gl_FragColor = vec4(vv4vertex.zzz/2.0/sqrt(3.0)+0.5, 0.0);
}
```

```
uniform sampler2D us2dtexture;
```

```
vec4 v4texColor = texture2D(us2dtexture, vv2texCoord).bgra;
```

Q
&
A