《计算机图形学实验》综合实验报告

题目 **OpenGL 实现茶壶的简单纹理贴图**

学 号 **20171060106**

姓 名 吴旭

指导教师 钱文华

日 期 **2022 年 6 月 20 日**

目录

摘要 3
一、实验内容 4
二、开发工具 4
三、　程序设计及基本模块 4
四、关键算法的理论介绍及步骤 5
1、关键算法介绍 5
2、实现步骤 6
五、运行结果 7
六、实验总结及体会 8
参考文献 8
附录 9

摘要

计算机图形学(Computer Graphics，简称 CG)是一种使用数学算法将二维或三维图形转化为计算机显示器的栅格形式的科学。简单地说，计算机图形学的主要研究内容就是研究如何在计算机中表示图形、以及利用计算机进行图形的计算、处理和显示的相关原理与算法。

本次计算机图形学的综合实验我主要进行了一个茶壶的贴图实现，让一些自定义的简单图像以及 bmp 类的图像能通过代码贴到实心茶壶上。通过对 OpenGL 各个强大的函数的使用，我成功让自定义的简单图像能附着在茶壶上，为茶壶增添了许多好看的花纹。在这过程中还对光照的函数进行了一些了解从而让茶壶看上去能有光照的样子。本次的期末综合实验是对之前所做实验的一次总结，也是对之前实验中所学到知识的一次总结运用，这次的综合实验不仅让我对 OpenGL 各个函数更加了解，而且也通过自身的亲自实践对图形学这门学科有了更深的认识。

关键词：OpenGL 茶壶 纹理贴图 光照

一、实验内容

利用 Codeblock, OpenGL，实现三维图形渲染，三维图形可以是 OpenGL 库中的各种三维图形，除了简单的立方体外还可以是球体、圆环、茶壶等。渲染过程中既可以使用自定义的一些简单花纹也可以是各种图片所产生的的纹理贴图。渲染完成后可以通过键盘实现对茶壶的平移、旋转等控制，以便从不同角度观察茶壶的渲染情况。

二、开发工具

Codeblock、OpenGL 等工具。

三、程序设计及基本模块

1、设计目标：

（1） 一个可以实现纹理贴图的茶壶；

（2） 茶壶具有光照效果；

（3） 可以通过键盘对茶壶进行基本的三维变换控制；

（4） 三维图形映射效果良好。

2、基本模块

本次实验只需一个图形显示界面，通过这个界面可以实现实验所要求的各个功能。

四、关键算法的理论介绍及步骤

1、关键算法介绍

BITMAPFILEHEADER、BITMAPINFOHEADER：以二进制读取图像获取 bitmap 图像的文件头。

fopen("D:rb.bmp", "rb")：以二进制只读的方式打开存储在 D 盘根目录下的文件。

GL_TEXTURE_2D：加载位图。

glTexImage2D：生成 2D 纹理函数使之贴合到茶壶上。

glTexEnvf(GL_TEXTURE_ENV,GL_TEXTURE_ENV_MODE,GL_MODULATE)：设置纹理的光照模式。

glutSolidTeapot(3)：画实线框茶壶并设置大小为 3。

case 'o': {bWire = !bWire; break; }：当按下键盘的 O 键时进行实体茶壶和线框茶壶的切换。

glLightfv(GL_LIGHT0, GL_POSITION, light_pos)：确定光源位置。

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);

glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT)：对纹理进行 S 方向和 T 方向的渲染

2、实现步骤

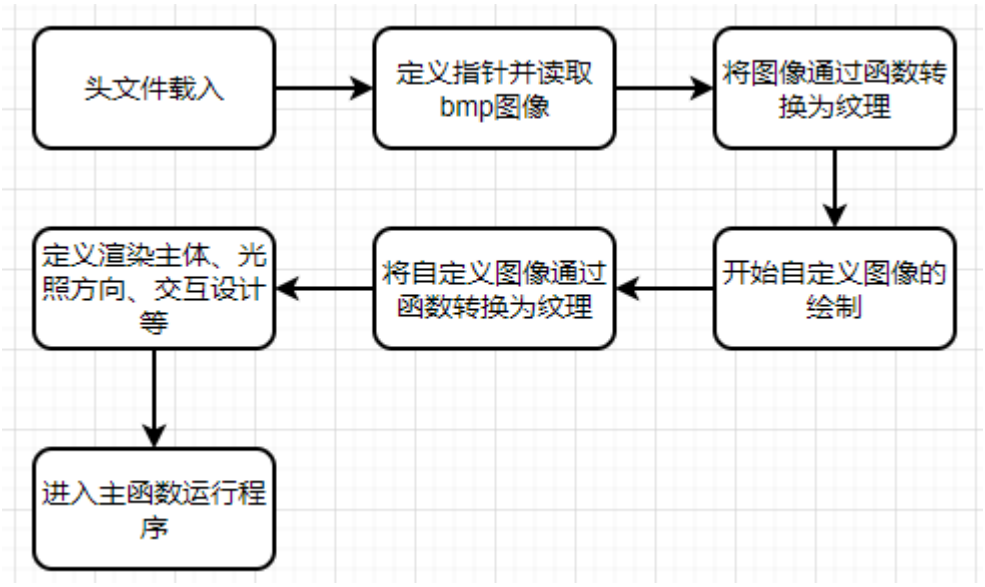

图 1 实现流程图

五、运行结果

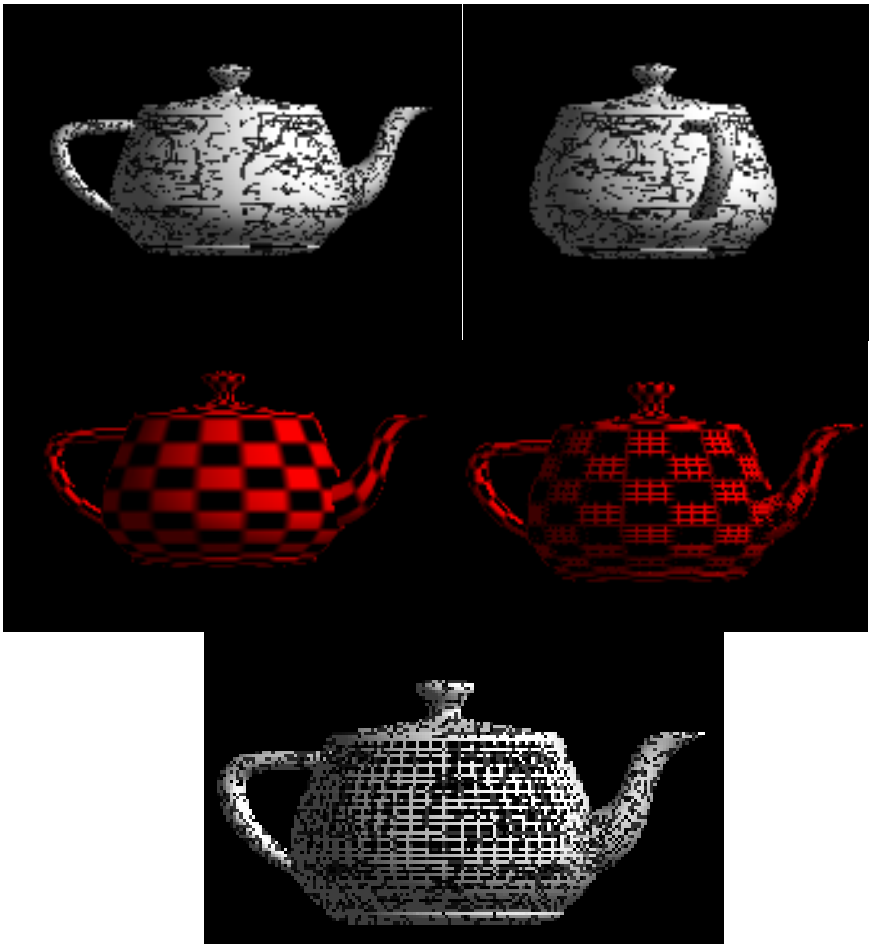

图 2 渲染结果

实验中用于贴图的图像：

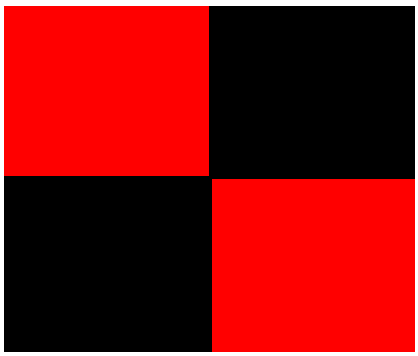

图 3 纹理贴图

六、实验总结及体会

1、首先对本次实验中存在的不足进行总结。经过一个学期的学习我明白了 OpenGL 是个强大的开源图像库，但因为自身编程能力比较弱所以在实现时大多还是对网上现成的资料进行理解和修改从而达到自己的目的，后期确实还需提升自己的逻辑思维和编程能力才能更好地进行今后的学习。

2、对本次实验中学到的东西进行总结。通过本次实验，第一点是让我对 bmp 图像有了一定了解，它没有进行数据压缩因此内部的色彩信息是直接以二进制的形式暴露在外，为生成纹理函数提供了很多便利，但很多图像都软件都不支持 bmp 图像的上传，因此不算好找；第二点是让我认识到了 OpenGL 本身强大的渲染功能，通过 glTexImage1D 、glTexImage2D 可以轻松地将一些图像渲染到三维的图像上，再结合光照函数的简单运用，最终的效果也还是可以的；第三点是在三维图形的视角变换时运用了正投影和透视投影等，在理论课上对投影的变换相对来说我觉得还是稍显复杂，但得益于 OpenGL 强大的库，只需要直接调用函数，搞清楚相应参数的含义并直接输入就可以了。

参考文献

[1] Donald Hearn. 计算机图形学（第四版）[M] . 电子工业出版社，2014.

[2] ZJU_fish1996. [opengl]茶壶与纹理[DB/OL] . CSDN，2016.

[3] V.Scott,Gordon John.Clevenger. 计算机图形学编程[M] . 人民邮电出版社，2020.

附录

代码：

```
#include <windows.h>

#include <stdlib.h>

#include <stdio.h>

#include <GL/glut.h>

#define BITMAP_ID 0x4D42

#define Height 16

#define Width 16

GLubyte image[Height][Width][3];

float fTranslate;

float fRotate;

float fScale        = 1.0f;
```

```
int status = 0;

int status2 = 1;

bool bPersp = false;

bool bAnim = false;

bool bWire = false;

int wHeight = 0;

int wWidth = 0;

GLuint texture[3];

void Draw_Leg();

unsigned char *LoadBitmapFile(char *filename, BITMAPINFOHEADER *bitmapInfoHeader)

{

    FILE *filePtr;

                BITMAPFILEHEADER bitmapFileHeader;

    unsigned char    *bitmapImage;

    int    imageIdx = 0;

    unsigned char    tempRGB;

    filePtr = fopen("D:rb1.bmp", "rb");
```

```
    if (filePtr == NULL) {
        printf("file not open\n");
        return NULL;
    }
    fread(&bitmapFileHeader, sizeof(BITMAPFILEHEADER), 1, filePtr);
    if (bitmapFileHeader.bfType != BITMAP_ID) {
        fprintf(stderr, "Error in LoadBitmapFile: the file is not a bitmap file\n");
        return NULL;
    }
    fread(bitmapInfoHeader, sizeof(BITMAPINFOHEADER), 1, filePtr);
    fseek(filePtr, bitmapFileHeader.bfOffBits, SEEK_SET);
    bitmapImage = new unsigned char[bitmapInfoHeader->biSizeImage];
    if (!bitmapImage) {
        fprintf(stderr, "Error in LoadBitmapFile: memory error\n");
        return NULL;
    }
    fread(bitmapImage, 1, bitmapInfoHeader->biSizeImage, filePtr);
```

```
        if (bitmapImage == NULL) {
                fprintf(stderr, "Error in LoadBitmapFile: memory error\n");
                return NULL;
        }
           for (imageIdx = 0;imageIdx < bitmapInfoHeader->biSizeImage; imageIdx
+= 3) {
                tempRGB = bitmapImage[imageIdx];
                bitmapImage[imageIdx] = bitmapImage[imageIdx + 2];
                bitmapImage[imageIdx + 2] = tempRGB;
        }
          fclose(filePtr);
        return bitmapImage;
}
void texload(int i, char *filename)
{
        BITMAPINFOHEADER bitmapInfoHeader;
        unsigned char*      bitmapData;
```

```
        bitmapData = LoadBitmapFile(filename, &bitmapInfoHeader);

        glBindTexture(GL_TEXTURE_2D, texture[i]);

        glTexParameteri(GL_TEXTURE_2D,                GL_TEXTURE_MAG_FILTER,
GL_NEAREST);

        glTexParameteri(GL_TEXTURE_2D,                GL_TEXTURE_MIN_FILTER,
GL_NEAREST);



        glTexImage2D(GL_TEXTURE_2D,
                0,                GL_RGB,                bitmapInfoHeader.biWidth,
                bitmapInfoHeader.biHeight,        0,
                GL_RGB,
                GL_UNSIGNED_BYTE,
                bitmapData);
}
void generateTex()
{
        for (int i = 0; i < Height; i++) {
                for (int j = 0; j < Width; j++) {
```

```
                int x = ((i & 4 ) ^(j & 4 )) * 255;
                image[i][j][0] = (GLubyte)x;
                image[i][j][1] = 0;
                image[i][j][2] = 0;
            }
        }
}
            void init()
{
    glGenTextures(3, texture); //
    texload(0, "Monet.bmp");
    texload(1, "Crack.bmp");
    texload(3, "Spot.bmp");
        generateTex();
    glBindTexture(GL_TEXTURE_2D, texture[2]);
    glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
    glTexImage2D(GL_TEXTURE_2D, 0, 3, Width, Height, 0, GL_RGB,
GL_UNSIGNED_BYTE, image);
```

```
        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_REPEAT);
        glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
}
void Draw_Triangle()
{
        glEnable(GL_TEXTURE_2D);
        glBindTexture(GL_TEXTURE_2D, texture[status]);
        glPushMatrix();
        glTranslatef(0, 0, 4+1);
        glRotatef(90, 1, 0, 0);
        glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_MODULATE);
        glutSolidTeapot(3);
        glPopMatrix();
        glDisable(GL_TEXTURE_2D);
        glEnable(GL_TEXTURE_2D);
```

```
        glBindTexture(GL_TEXTURE_2D, texture[status2]);    texture[status2]
        glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
        //glDisable(GL_TEXTURE_2D);
}
void idle()
{
        glutPostRedisplay();
}
float eye[] = {0, 0, 8};
float center[] = {0, 0, 0};
void key(unsigned char k, int x, int y)
{
switch (k)
        {
        case 27:
        case ' ': {bAnim = !bAnim; break; }
        case 'o': {bWire = !bWire; break; }
```

```
case 'a': {
        eye[0] -= 0.2f;
        center[0] -= 0.2f;
        break;
}
case 'd': {
        eye[0] += 0.2f;
        center[0] += 0.2f;
        break;
}
case 'w': {
        eye[1] -= 0.2f;
        center[1] -= 0.2f;
        break;
}
case 's': {
        eye[1] += 0.2f;
```

```
        center[1] += 0.2f;
        break;
    }
    case 'z': {
        eye[2] -= 0.2f;
        center[2] -= 0.2f;
        break;
    }
    case 'c': {
        eye[2] += 0.2f;
        center[2] += 0.2f;
        break;
    }
    case 'r': { //切换茶壶纹理
        if (status == 0)status = 2;
        else if (status == 2)status = 0;
        break;
```

```
        }

        }

        updateView(wHeight, wWidth);

}

void redraw()

{

        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

        glLoadIdentity();



        gluLookAt(eye[0], eye[1], eye[2],

                center[0], center[1], center[2],

                0, 1, 0);

        if (bWire) {

                glPolygonMode(GL_FRONT_AND_BACK, GL_LINE);

        }

        else {

                glPolygonMode(GL_FRONT_AND_BACK, GL_FILL);
```

```
        }

        glEnable(GL_DEPTH_TEST);

        glEnable(GL_LIGHTING);

        GLfloat white[] = { 1.0, 1.0, 1.0, 1.0 };

        GLfloat light_pos[] = { 5,5,5,1 };

        glLightfv(GL_LIGHT0, GL_POSITION, light_pos);

        glLightfv(GL_LIGHT0, GL_AMBIENT, white);

        glEnable(GL_LIGHT0);

        glRotatef(fRotate, 0, 1.0f, 0);

        glRotatef(-90, 1, 0, 0);

        glScalef(0.2, 0.2, 0.2);

        Draw_Triangle();



        if (bAnim) fRotate        += 0.5f;

        glutSwapBuffers();

}
```

```
int main(int argc, char *argv[])

{

	glutInit(&argc, argv);

	glutInitDisplayMode(GLUT_RGBA | GLUT_DEPTH | GLUT_DOUBLE);

	    glutInitWindowSize(480, 480);

	int windowHandle = glutCreateWindow("OpenGL");



	glutDisplayFunc(redraw);

	glutReshapeFunc(reshape);

	glutKeyboardFunc(key);

	glutIdleFunc(idle);

	init();

	glutMainLoop();

	return 0;

}
```