**北京师范大学珠海分校**

**【计算机图形学】实验报告**

**班级：** 数媒2班 **学号：** 1801050056  **姓名：** 何金城

|  |  |  |  |
| --- | --- | --- | --- |
| **实验 （名称）实验十三 雾、透明和阴影** | | | |
| **成 绩** |  | **（日期、星期、节次）** |  |
| **指导教师** |  | **地点** |  |
| **一、实验目的：**  1. 了解雾的绘制与效果  2．利用OpenGL的Blending和对称功能实现透明效果  3。利用投影变换矩阵可实现阴影效果 | | | |
| **二、实验内容：**  **要求：**   1. 设计你的3D小场景，同时加上光照材质、雾、透明和阴影效果。   撰写实验报告，将效果截图和主程序拷贝到实验报告文档里，提交实验报告：学号姓名-计算机图形学实验报告-lab13.docx。 | | | |
| 1. **“雾、透明和阴影”程序设计（包括：程序代码、注释和运行结果截图等）**   **#include "stdafx.h"**  **#include <math.h>**  **#include <glut.h>**  **#include "GLTools.h"**  **#define PI 3.14159**  **float theta = -90.0; //圆环旋转角**  **float angle = 10; //左右场景每次旋转角**  **float sightangle = -90;**  **float s = 10; //前后直走步长**  **float R = 100;**  **int inner = 10, outer = 80;**  **float eyex = 0, eyey = 0, eyez = outer + 4 \* inner + 50; //初始视点位置**  **float atx = 0, aty = 0, atz = 0; //初始目标点位置**  **float atx1, atz1, eyex1, eyez1;**  **float tt = 0, tt2 = 0;**  **GLTMatrix mShadowMatrix;**  **GLTVector3 vPoints[3] = { { 0.0f, -outer - 4 \* inner, 0.0f },**  **{ 10.0f, -outer - 4 \* inner, 0.0f },**  **{ 5.0f, -outer - 4 \* inner, -5.0f } };**  **// void specialkeyboard(int key, int x, int y);**  **void mykeyboard(unsigned char key, int x, int y);**  **void Display(void);**  **void Reshape(int w, int h);**  **void myidle();**  **void drawground();**  **void drawsphere();**  **void drawwall();**  **void init();**  **int APIENTRY \_tWinMain(HINSTANCE hInstance,**  **HINSTANCE hPrevInstance,**  **LPTSTR lpCmdLine,**  **int nCmdShow)**  **{**  **UNREFERENCED\_PARAMETER(hPrevInstance);**  **UNREFERENCED\_PARAMETER(lpCmdLine);**  **char \*argv[] = { (char\*)"hello ", (char\*)" " };**  **int argc = 2; // must/should match the number of strings in argv**  **glutInit(&argc, argv); //初始化GLUT库；**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB); //设置显示模式；（缓冲，颜色类型）**  **glutInitWindowSize(500, 500);**  **glutInitWindowPosition(1024 / 2 - 250, 768 / 2 - 250);**  **glutCreateWindow("Rotating 3D World"); //创建窗口，标题为“Rotating 3D World”；**  **glutReshapeFunc(Reshape);**  **init();**  **glutDisplayFunc(Display); //用于绘制当前窗口；**  **glutKeyboardFunc(mykeyboard);**  **glutIdleFunc(myidle);**  **glutMainLoop(); //表示开始运行程序，用于程序的结尾；**  **return 0;**  **}**  **void init()**  **{**  **//define light position1**  **GLfloat light\_position1[] = { -outer,outer,outer + inner,0.0 };**  **// GLfloat light\_position1[]={0,0,0,0.0};**  **GLfloat light\_position2[] = { +outer,-outer,outer + inner,0.0 };**  **//GLfloat light color1**  **GLfloat light\_ambient1[] = { 1.0,1.0,1.0,1.0 };**  **GLfloat light\_diffuse1[] = { 1.0,1.0,1.0,1.0 };**  **GLfloat light\_specular1[] = { 1.0,1.0,1.0,1.0 };**  **//GLfloat light color2**  **GLfloat light\_ambient2[] = { 0.8,0.8,0.8,1.0 };**  **GLfloat light\_diffuse2[] = { 0.8,0.8,0.8,1.0 };**  **GLfloat light\_specular2[] = { 0.8,0.8,0.8,1.0 };**  **// light model- global light**  **GLfloat lmodel\_ambient[] = { 0.8,0.2,0.2,1.0 };**  **glLightModelfv(GL\_LIGHT\_MODEL\_AMBIENT, lmodel\_ambient);**  **glLightModeli(GL\_LIGHT\_MODEL\_LOCAL\_VIEWER, GL\_TRUE);**  **// set light source location**  **glLightfv(GL\_LIGHT0, GL\_POSITION, light\_position1);**  **glLightfv(GL\_LIGHT1, GL\_POSITION, light\_position2);**  **// set light source color;**  **glLightfv(GL\_LIGHT0, GL\_AMBIENT, light\_ambient1);**  **glLightfv(GL\_LIGHT0, GL\_DIFFUSE, light\_diffuse1);**  **glLightfv(GL\_LIGHT0, GL\_SPECULAR, light\_specular1);**  **// set light source color;**  **glLightfv(GL\_LIGHT1, GL\_AMBIENT, light\_ambient2);**  **glLightfv(GL\_LIGHT1, GL\_DIFFUSE, light\_diffuse2);**  **glLightfv(GL\_LIGHT1, GL\_SPECULAR, light\_specular2);**  **// set material let material fits color**  **glEnable(GL\_COLOR\_MATERIAL);**  **glColorMaterial(GL\_FRONT, GL\_AMBIENT\_AND\_DIFFUSE);**  **//make light source enable;**  **glEnable(GL\_LIGHTING);**  **glEnable(GL\_LIGHT0);**  **glEnable(GL\_LIGHT1);**  **GLfloat fogColor[] = { 1, 0.8, 0.8, 1 };**  **glFogfv(GL\_FOG\_COLOR, fogColor); //fog’s color**  **glFogf(GL\_FOG\_START, 250.0f); // how far to start**  **glFogf(GL\_FOG\_END, 290.0f); //how far to end**  **glFogi(GL\_FOG\_MODE, GL\_LINEAR); //which mode**  **glFogf(GL\_FOG\_DENSITY, 0.1f);**  **glEnable(GL\_FOG);**  **gltMakeShadowMatrix(vPoints, light\_position1, mShadowMatrix); //阴影变换矩阵**  **//clear color**  **glClearColor(0.0, 0.0, 0.0, 0.0);**  **//Color shade mode**  **glShadeModel(GL\_SMOOTH);**  **// polygon mode**  **glPolygonMode(GL\_FRONT, GL\_FILL);**  **// Anti-aliasing**  **/\* glEnable(GL\_POINT\_SMOOTH);**  **glHint(GL\_POINT\_SMOOTH\_HINT, GL\_NICEST);**  **glEnable(GL\_LINE\_SMOOTH);**  **glHint(GL\_LINE\_SMOOTH\_HINT, GL\_NICEST);**  **glEnable(GL\_POLYGON\_SMOOTH);**  **glHint(GL\_POLYGON\_SMOOTH\_HINT, GL\_NICEST);\*/**  **// glEnable(GLUT\_MULTISAMPLE);**  **// depth detecting**  **glEnable(GL\_DEPTH\_TEST);**  **}**  **void Display(void)**  **{**  **glClearColor(1, 1, 1, 1);**  **// glClear(GL\_COLOR\_BUFFER\_BIT);**  **glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);**  **glMatrixMode(GL\_MODELVIEW);**  **glLoadIdentity();**  **// gluLookAt(0,-10,350,0,0,0,0,1,0);**  **gluLookAt(eyex, eyey, eyez, atx, aty, atz, 0, 1, 0);**  **//about ground**  **glPushMatrix();**  **glFrontFace(GL\_CW);**  **glTranslatef(0, -outer - 4 \* inner, 0);**  **glScalef(1.0f, -1.0f, 1.0f);**  **glTranslatef(0, outer + 4 \* inner, 0);**  **drawsphere(1);**  **glFrontFace(GL\_CCW);**  **glPopMatrix();**  **//about top**  **glPushMatrix();**  **glFrontFace(GL\_CW);**  **glTranslatef(0, +outer + 4 \* inner, 0);**  **glScalef(1.0f, -1.0f, 1.0f);**  **glTranslatef(0, -outer - 4 \* inner, 0);**  **drawsphere(1);**  **glFrontFace(GL\_CCW);**  **glPopMatrix();**  **//about right wall**  **glPushMatrix();**  **glFrontFace(GL\_CW);**  **glTranslatef(+outer + 4 \* inner, 0, 0);**  **glScalef(-1.0f, 1.0f, 1.0f);**  **glTranslatef(-outer - 4 \* inner, 0, 0);**  **drawsphere(1);**  **glFrontFace(GL\_CCW);**  **glPopMatrix();**  **//about left wall**  **glPushMatrix();**  **glFrontFace(GL\_CW);**  **glTranslatef(-outer - 4 \* inner, 0, 0);**  **glScalef(-1.0f, 1.0f, 1.0f);**  **glTranslatef(+outer + 4 \* inner, 0, 0);**  **drawsphere(1);**  **glFrontFace(GL\_CCW);**  **glPopMatrix();**  **//about front wall**  **glPushMatrix();**  **glFrontFace(GL\_CW);**  **glTranslatef(0, 0, -outer - 4 \* inner);**  **glScalef(1.0f, 1.0f, -1.0f);**  **glTranslatef(0, 0, outer + 4 \* inner);**  **drawsphere(1);**  **glFrontFace(GL\_CCW);**  **glPopMatrix();**  **glDisable(GL\_LIGHTING);**  **glEnable(GL\_BLEND);**  **glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);**  **drawground();**  **drawwall();**  **glDisable(GL\_BLEND);**  **glEnable(GL\_LIGHTING);**  **// Draw shadows first**  **glDisable(GL\_DEPTH\_TEST);**  **glDisable(GL\_LIGHTING);**  **glPushMatrix();**  **glMultMatrixf(mShadowMatrix);**  **drawsphere(0);**  **glPopMatrix();**  **glEnable(GL\_LIGHTING);**  **glEnable(GL\_DEPTH\_TEST);**  **drawsphere(1);**  **glutSwapBuffers();**  **// glFlush();**  **}**  **void drawsphere(int flag)**  **{**  **float tr;**  **tr = (outer + 3 \* inner);**  **glRotatef(theta, 0, 1, 0);**  **glPushMatrix();**  **glPushMatrix();**  **if (flag == 1)**  **glColor3f(1.0, 0, 0.0);**  **else**  **glColor3f(0.5, 0.5, 0.5);**  **glutSolidTorus(inner, outer, 50, 80);**  **glPopMatrix();**  **glPushMatrix();**  **glTranslatef(outer, 0, 0);**  **glRotatef(theta, 0, 1, 0);**  **glTranslatef(-outer, 0, 0);**  **glPushMatrix();**  **glTranslatef(tr, 0, 0);**  **glRotatef(-45, 1, 0, 0);**  **if (flag == 1)**  **glColor3f(0.0, 1.0, 0);**  **else**  **glColor3f(0.5, 0.5, 0.5);**  **glutSolidSphere(inner, 40, 40);**  **glPopMatrix();**  **glPopMatrix();**  **glPopMatrix();**  **}**  **void drawground()**  **{**  **int colorflag = 1;**  **GLfloat mat\_specular1[] = { 1.0,1.0,1.0,1.0 };**  **GLfloat mat\_shininess1[] = { 80.0 };**  **glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular1);**  **glMaterialfv(GL\_FRONT, GL\_SHININESS, mat\_shininess1);**  **glNormal3f(0, 1, 0);**  **for (int i = -outer - 4 \* inner; i<outer + 4 \* inner; i += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **for (int j = -outer - 4 \* inner; j<outer + 4 \* inner; j += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **glBegin(GL\_QUADS);**  **glVertex3d(j, -outer - 4 \* inner, i);**  **glVertex3d(j, -outer - 4 \* inner, i + 2 \* inner);**  **glVertex3d(j + 2 \* inner, -outer - 4 \* inner, i + 2 \* inner);**  **glVertex3d(j + 2 \* inner, -outer - 4 \* inner, i);**  **glEnd();**  **}**  **}**  **glNormal3f(0, -1, 0);**  **colorflag = 1;**  **for (int i = -outer - 4 \* inner; i<outer + 4 \* inner; i += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **for (int j = -outer - 4 \* inner; j<outer + 4 \* inner; j += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **glBegin(GL\_QUADS);**  **glVertex3d(j, outer + 4 \* inner, i);**  **glVertex3d(j, outer + 4 \* inner, i + 2 \* inner);**  **glVertex3d(j + 2 \* inner, outer + 4 \* inner, i + 2 \* inner);**  **glVertex3d(j + 2 \* inner, outer + 4 \* inner, i);**  **glEnd();**  **}**  **}**  **}**  **void drawwall()**  **{**  **int i, j;**  **glNormal3f(1, 0, 0);**  **int colorflag = 1;**  **//left**  **for (i = -outer - 4 \* inner; i<outer + 4 \* inner; i += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **for (j = -outer - 4 \* inner; j<outer + 4 \* inner; j += 2 \* inner)**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **glBegin(GL\_QUADS);**  **glVertex3d(-outer - 4 \* inner, j, i);**  **glVertex3d(-outer - 4 \* inner, j + 2 \* inner, i);**  **glVertex3d(-outer - 4 \* inner, j + 2 \* inner, i + 2 \* inner);**  **glVertex3d(-outer - 4 \* inner, j, i + 2 \* inner);**  **glEnd();**  **}**  **}**  **colorflag = 1;**  **glNormal3f(0, -1, 0);**  **//right**  **for (i = -outer - 4 \* inner; i <= outer + 4 \* inner - 2 \* inner; i += 2 \* inner) //for z**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **for (j = -outer - 4 \* inner; j <= outer + 4 \* inner - 2 \* inner; j += 2 \* inner) //for y**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **glBegin(GL\_QUADS);**  **glVertex3f(outer + 4 \* inner, j, i);**  **glVertex3f(outer + 4 \* inner, j + 2 \* inner, i);**  **glVertex3f(outer + 4 \* inner, j + 2 \* inner, i + 2 \* inner);**  **glVertex3f(outer + 4 \* inner, j, i + 2 \* inner);**  **glEnd();**  **}**  **}**  **colorflag = 1;**  **glNormal3f(0, 0, 1);**  **//front**  **for (i = -outer - 4 \* inner; i <= outer + 4 \* inner - 2 \* inner; i += 2 \* inner) //for z**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **for (j = -outer - 4 \* inner; j <= outer + 4 \* inner - 2 \* inner; j += 2 \* inner) //for y**  **{**  **if (colorflag>0) glColor4f(1.0, 1.0, 0.0, 0.8);**  **else glColor4f(1.0, 1.0, 1.0, 0.8);**  **colorflag = -colorflag;**  **glBegin(GL\_QUADS);**  **glVertex3f(j, i, -outer - 4 \* inner);**  **glVertex3f(j + 2 \* inner, i, -outer - 4 \* inner);**  **glVertex3f(j + 2 \* inner, i + 2 \* inner, -outer - 4 \* inner);**  **glVertex3f(j, i + 2 \* inner, -outer - 4 \* inner);**  **glEnd();**  **}**  **}**  **}**  **void myidle()**  **{**  **theta += 0.5;**  **// if (theta>=360.0) theta-=360.0;**  **glutPostRedisplay();**  **}**  **void Reshape(GLsizei w, GLsizei h)**  **{**  **glMatrixMode(GL\_PROJECTION);**  **glLoadIdentity();**  **// glOrtho(-outer-6\*inner,outer+6\*inner,-outer-4\*inner,outer+4\*inner,20,2\*outer+8\*inner+50);**  **gluPerspective(90, w / h, 10, 2 \* outer + 8 \* inner + 250);**  **glViewport(0, 0, w, h);**  **glMatrixMode(GL\_MODELVIEW);**  **}**  **void mykeyboard(unsigned char key, int x, int y)**  **{**  **switch (key)**  **{**  **case 'W':**  **case 'w':// 向前走**  **eyex1 = eyex - s\*sin(sightangle \* 2 \* PI / 360 - PI / 2);**  **eyez1 = eyez - s\*cos(sightangle \* 2 \* PI / 360 - PI / 2);**  **atx1 = atx - s\*sin(sightangle \* 2 \* PI / 360 - PI / 2);**  **atz1 = atz - s\*cos(sightangle \* 2 \* PI / 360 - PI / 2);**  **eyex = eyex1;**  **eyez = eyez1;**  **atz = atz1;**  **atx = atx1;**  **break;**  **case 'S':**  **case 's'://向后走**  **eyex1 = eyex + s\*sin(sightangle \* 2 \* PI / 360 - PI / 2);**  **eyez1 = eyez + s\*cos(sightangle \* 2 \* PI / 360 - PI / 2);**  **atx1 = atx + s\*sin(sightangle \* 2 \* PI / 360 - PI / 2);**  **atz1 = atz + s\*cos(sightangle \* 2 \* PI / 360 - PI / 2);**  **eyex = eyex1;**  **eyez = eyez1;**  **atz = atz1;**  **atx = atx1;**  **break;**  **case 'A':**  **case 'a'://左转**  **/\* atx1=eyex+(atx-eyex)\*cos(angle\*2\*PI/360.0)+(eyez-atz)\*sin(angle\*2\*PI/360.0);**  **atz1=eyez-(eyez-atz)\*sin(angle\*2\*PI/360.0)+(atx-eyex)\*cos(angle\*2\*PI/360.0);\*/**  **atx1 = eyex + (atx - eyex)\*cos(angle \* 2 \* PI / 360.0) + (eyez - atz)\*sin(angle \* 2 \* PI / 360.0);**  **atz1 = eyez - (eyez - atz)\*cos(angle \* 2 \* PI / 360.0) - (-atx + eyex)\*sin(angle \* 2 \* PI / 360.0);**  **atx = atx1;**  **atz = atz1;**  **sightangle = sightangle + angle;**  **break;**  **case 'D':**  **case 'd'://右转**  **/\*atx1=eyex+(atx-eyex)\*cos(-angle\*2\*PI/360.0)+(eyez-atz)\*sin(-angle\*2\*PI/360.0);**  **atz1=eyez-(eyez-atz)\*sin(-angle\*2\*PI/360.0)+(atx-eyex)\*cos(-angle\*2\*PI/360.0);\*/**  **atx1 = eyex + (atx - eyex)\*cos(angle \* 2 \* PI / 360.0) - (eyez - atz)\*sin(angle \* 2 \* PI / 360.0);**  **atz1 = eyez - (eyez - atz)\*cos(angle \* 2 \* PI / 360.0) + (-atx + eyex)\*sin(angle \* 2 \* PI / 360.0);**  **atx = atx1;**  **atz = atz1;**  **sightangle = sightangle - angle;**  **break;**  **}**  **//参数修改后调用重画函数，屏幕图形将发生改变**  **glutPostRedisplay();**  **}** | | | |
| **四、实验总结：**  有点难理解，反复看了几遍回放才能运行，前面一直报错 | | | |