|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **计算机图形学实验报告（六）** | | | | | | |
| 学 号 | 姓 名 | | 班 级 | | 报告日期 | |
| 180400715 | 杨卓宸 | | 1804102 | | 2020/11/24 | |
| 实验内容 | 光照模型实验 | | | | | |
| 实验目的 | 绘制一个Bezier曲面, 利用 Phong光照模型生成曲面的明暗效果。 | | | | | |
| 实验预备知识 | 1.Glut库:  本程序使用glut库，简化程序界面和交互操作的编写，glut的文件夹 glutdlls37beta.  将glut的dll文件拷贝到系统目录（system32）或者程序的debug/release目录下程序才 能正常运行。  2.Bezier曲面:      3.Phong光照明模型的实现： | | | | | |
| 实验过程描述 | 1. surfacepoint()计算Bezier 曲面上的点   计算Bezier曲面上一些采样点，把这些点连成一个三角形网格M.  void surfacepoint(double u, double v, double& x, double& y, double& z)  {  double ax0 = -0.5;  double ay0 = 0.1;  double az0 = -0.2;  double bx0 = 0.1;  double by0 = 0.4;  double bz0 = 0.1;  double cx0 = 0.5;  double cy0 = 0.1;  double cz0 = 0.3;  double ax1 = -0.5;  double ay1 = -0.2;  double az1 = 0.2;  double bx1 = 0.2;  double by1 = 0.2;  double bz1 = 0.3;  double cx1 = 0.5;  double cy1 = -0.2;  double cz1 = 0.2;  double ax2 = -0.5;  double ay2 = -0.6;  double az2 = 0.3;  double bx2 = 0.1;  double by2 = -0.4;  double bz2 = 0.5;  double cx2 = 0.5;  double cy2 = -0.3;  double cz2 = 0.1;  double u1 = (1 - u) \* (1 - u);  double u2 = 2 \* (1 - u) \* u;  double u3 = u \* u;  double ax = u1 \* ax0 + u2 \* ax1 + u3 \* ax2;  double ay = u1 \* ay0 + u2 \* ay1 + u3 \* ay2;  double az = u1 \* az0 + u2 \* az1 + u3 \* az2;  double bx = u1 \* bx0 + u2 \* bx1 + u3 \* bx2;  double by = u1 \* by0 + u2 \* by1 + u3 \* by2;  double bz = u1 \* bz0 + u2 \* bz1 + u3 \* bz2;  double cx = u1 \* cx0 + u2 \* cx1 + u3 \* cx2;  double cy = u1 \* cy0 + u2 \* cy1 + u3 \* cy2;  double cz = u1 \* cz0 + u2 \* cz1 + u3 \* cz2;  double v1 = (1 - v) \* (1 - v);  double v2 = 2 \* (1 - v) \* v;  double v3 = v \* v;  x = v1 \* ax + v2 \* bx + v3 \* cx;  y = v1 \* ay + v2 \* by + v3 \* cy;  z = v1 \* az + v2 \* bz + v3 \* cz;  }   1. phong()计算顶点的颜色   利用Phong光照模型计算网格M上所有顶点的颜色，绘制该网格曲面.  网格曲面的绘制过程为依次绘制网格曲面的每一个三角形。  void phong()  {  //每一个顶点计算一个颜色  GLfloat SZ[3];//入射光向量  GLfloat RZ[3];//镜面反射光向量  GLfloat VZ[3];//人眼的视角向量  //每一个顶点计算一个颜色  for (int i = 0; i < NUMV; i++) {  //计算每个网格的视角向量，---------GLfloat vertices[MAXNUMV][3];存放网格的顶点  VZ[0] = vertices[i][0] - xcam;//xcam、ycam、zcam是人眼的坐标  VZ[1] = vertices[i][1] - ycam;  VZ[2] = vertices[i][2] - zcam;  Normalize(VZ);//单位化  //计算光源的入向量，---------------GLfloat light\_position0[] = { 0.5, 0.5, 1.0, 1.0};光源的位置  SZ[0] = vertices[i][0] - light\_position0[0];  SZ[1] = vertices[i][1] - light\_position0[1];  SZ[2] = vertices[i][2] - light\_position0[2];  Normalize(SZ);//单位化  //计算n·s  GLfloat cos1 = SZ[0] \* normals[i][0] + SZ[1] \* normals[i][1] + SZ[2] \* normals[i][2];  //计算反射光的向量r = 2 (n·s)n - s；ppt上的公式  RZ[0] = 2 \* (cos1)\* normals[i][0] - SZ[0];  RZ[1] = 2 \* (cos1)\* normals[i][1] - SZ[1];  RZ[2] = 2 \* (cos1)\* normals[i][2] - SZ[2];  Normalize(RZ);//单位化  //计算反射光与人眼之间的夹角的cos值：r·v  GLfloat cos2 = RZ[0] \* VZ[0] + RZ[1] \* VZ[1] + RZ[2] \* VZ[2];  colors[i][0] = ambient[0] \* light\_ambient[0] + diffuse[0] \* light\_diffuse0[0] \* cos1 + specular[0] \* light\_specular0[0] \* pow(cos2, shininess);  colors[i][1] = ambient[1] \* light\_ambient[1] + diffuse[1] \* light\_diffuse0[1] \* cos1 + specular[1] \* light\_specular0[1] \* pow(cos2, shininess);  colors[i][2] = ambient[2] \* light\_ambient[2] + diffuse[2] \* light\_diffuse0[2] \* cos1 + specular[2] \* light\_specular0[2] \* pow(cos2, shininess);  }  }   1. DrawFace()绘制曲面   绘制一个三角形时，要指定每一个顶点的颜色（顶点颜色即采用Phong模型计算出的颜色）。  指定一个顶点的颜色 glColor3f(r,g,b)。 | | | | | |
| 实验结果 | 直接运行结果：    利用鼠标旋转结果：  捕获 | | | | | |
| 实验当中问题  及解决方法 | 问题：起初直接运行已有代码，程序报错显示各种头文件无法找到。  解决方法：没有仔细查看题目详解后面的提示，只是单纯的上网查找原因，经过一些简单的修改仍未成功。之后发现要配置库，配置好之后还是不能正常运行，之后通过同学的建议尝试更改了Windows SDK版本问题得到解决。 | | | | | |
| 成绩（教师打分） | 优秀 | 良好 | | 及格 | | 不及格 |