МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Кафедра компьютерной технологий и программной инженерии

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| асс. |  |  |  | И.М. Лозоватский |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5 |
| «Работа со списками, текстурой и NURBS-поверхностями» |
| по курсу: КОМПЬЮТЕРНАЯ ГРАФИКА |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4831 |  |  |  | К.А.Корюнщенков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2020

**Цель работы**

Целью работы является получение практических навыков по работе со списками, текстурой и NUMBS-поверхностями.

**Задание на лабораторную работу**

Вывести сложный трехмерный объект, сохранить его в виде списка и размножить в виде нескольких копий различного масштаба на экране путем повторного вывода списка. Выводимые объекты располагаются над рельефом,задаваемым NURBS-поверхностью. Наложить текстуру на выводимые объекты, сцена должна быть освещена, необходимо предусмотреть возможность включения эффекта тумана (при помощи клавиатуры или мыши).

**Ход работы**

**Листинг программы:**

#include <string.h>

#include <stdlib.h>

#include <stdio.h>

#include <cmath>

#include <GLUT/glut.h>

#include <cmath>

#include <stdlib.h>

#include <stdio.h>

#include <iostream>

#include <math.h>

GLfloat ctrlpoints[4][4][3];

GLUnurbsObj \*theNurb;

GLuint theTeapot;

#define stripeImageWidth 64

GLubyte stripeImage[4 \* stripeImageWidth];

GLuint texName;

GLfloat xequalzero[] = { 1.0, 0.0, 1.0, 0.0 };

GLfloat slanted[] = { 1.0, 0.0, 5.0, 0.0 };

GLfloat \*currentCoeff;

GLenum currentPlane;

GLint currentGenMode;

GLint fog;

void init\_nurb() {

//Рисуем подставку

int u, v;

for (u = 0; u<4; u++) {

for (v = 0; v<4; v++) {

ctrlpoints[u][v][0] = 3.0\*((GLfloat)u - 1.5);

ctrlpoints[u][v][1] = 2.0\*((GLfloat)v - 1.5);

if ((u == 1 || u == 2) && (v == 1 || v == 2))

ctrlpoints[u][v][2] = 3.0;

else

ctrlpoints[u][v][2] = -3.0;

}

}

GLfloat mat\_diffuse[] = { 1.0, 0.7, 0.3, 1.0 };

GLfloat mat\_specular[] = { 1.0, 0.5, 1.0, 1.0 };

GLfloat mat\_shininess[] = { 100.0 };

glClearColor(0.0, 0.0, 0.0, 0.0);

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, mat\_diffuse);

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat\_specular);

glMaterialfv(GL\_FRONT, GL\_SHININESS, mat\_shininess);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT0);

glEnable(GL\_DEPTH\_TEST);

glEnable(GL\_AUTO\_NORMAL);

glEnable(GL\_NORMALIZE);

theNurb = gluNewNurbsRenderer();

gluNurbsProperty(theNurb, GLU\_SAMPLING\_TOLERANCE, 5.0);

gluNurbsProperty(theNurb, GLU\_DISPLAY\_MODE, GLU\_FILL);

glutPostRedisplay();

}

//кривые полоски на чайниках

void makeStripeImage(){

int j;

for (j = 0; j < stripeImageWidth; j++){

stripeImage[4 \* j] = (GLubyte)((j <= 4) ? 255 : 0);

stripeImage[4 \* j + 1] = (GLubyte)((j>4) ? 255 : 0);

stripeImage[4 \* j + 2] = (GLubyte)0;

stripeImage[4 \* j + 3] = (GLubyte)255;

}

}

void display(){

GLfloat knots[8] = { 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 1.0, 1.0 };

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

glPushMatrix();

glRotatef(330.0, 20.0, 0.0, 0.0);

glScalef(2.0, 0.5, 1.0);

gluBeginSurface(theNurb);

gluNurbsSurface(theNurb, 8, knots, 8, knots, 4 \* 3, 3,&ctrlpoints[0][0][0], 4, 4, GL\_MAP2\_VERTEX\_3);

gluEndSurface(theNurb);

glPopMatrix();

glFlush();

glPushMatrix();

glTranslatef(-2.0, 1.1, 0.0);

glCallList(theTeapot);

glPopMatrix();

glFlush();

}

void init(){

glClearColor(1.0, 1.0, 1.0, 0.0);

glEnable(GL\_DEPTH\_TEST);

glShadeModel(GL\_SMOOTH);

makeStripeImage();

glPixelStorei(GL\_UNPACK\_ALIGNMENT, 5);

glGenTextures(1, &texName);

glBindTexture(GL\_TEXTURE\_1D, texName);

glTexParameteri(GL\_TEXTURE\_1D, GL\_TEXTURE\_WRAP\_S, GL\_REPEAT);

glTexParameteri(GL\_TEXTURE\_1D, GL\_TEXTURE\_MAG\_FILTER, GL\_LINEAR);

glTexParameteri(GL\_TEXTURE\_1D, GL\_TEXTURE\_MIN\_FILTER, GL\_LINEAR);

glTexImage1D(GL\_TEXTURE\_1D, 0, GL\_RGBA, stripeImageWidth, 0,

GL\_RGBA, GL\_UNSIGNED\_BYTE, stripeImage);

glTexEnvf(GL\_TEXTURE\_ENV, GL\_TEXTURE\_ENV\_MODE, GL\_MODULATE);

currentCoeff = xequalzero;

currentGenMode = GL\_OBJECT\_LINEAR;

currentPlane = GL\_OBJECT\_PLANE;

glTexGeni(GL\_S, GL\_TEXTURE\_GEN\_MODE, currentGenMode);

glTexGenfv(GL\_S, currentPlane, currentCoeff);

glEnable(GL\_TEXTURE\_GEN\_S);

glEnable(GL\_TEXTURE\_1D);

glEnable(GL\_CULL\_FACE);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT3);

glEnable(GL\_AUTO\_NORMAL);

glEnable(GL\_NORMALIZE);

glFrontFace(GL\_CW);

glCullFace(GL\_BACK);

GLfloat position[] = { 500.0, 1030.0, 950.0, 0.0 };

glEnable(GL\_DEPTH\_TEST);

glLightfv(GL\_LIGHT1, GL\_POSITION, position);

glEnable(GL\_LIGHTING);

glEnable(GL\_LIGHT1);

{

GLfloat mat1[3] = { 1.0, 1.0, 1.0 };

glMaterialfv(GL\_FRONT, GL\_AMBIENT, mat1);

mat1[0] = 1.0; mat1[1] = 0.0; mat1[2] = 0.0;

glMaterialfv(GL\_FRONT, GL\_DIFFUSE, mat1);

mat1[0] = 1.0; mat1[1] = 1.0; mat1[2] = 1.0;

glMaterialfv(GL\_FRONT, GL\_SPECULAR, mat1);

glMaterialf(GL\_FRONT, GL\_SHININESS, 5.0);

}

glEnable(GL\_FOG);

{

GLfloat fogColor[4] = { 0.5, 0.5, 0.5, 1.0 };

fog = GL\_EXP;

glFogi(GL\_FOG\_MODE, fog);

glFogfv(GL\_FOG\_COLOR, fogColor);

glFogf(GL\_FOG\_DENSITY, 0.35);

glHint(GL\_FOG\_HINT, GL\_NICEST);

glFogf(GL\_FOG\_START, 1.0);

glFogf(GL\_FOG\_END, 5.0);

}

//Рисуем чайник

theTeapot = glGenLists(1);

glNewList(theTeapot, GL\_COMPILE);

glTranslatef(0.0, 0.0, 0.0);

glutSolidTeapot(0.8);

glTranslatef(2.2, 0.0, 0.0);

glutSolidTeapot(0.6);

glTranslatef(1.5, 0.0, 0.0);

glutSolidTeapot(0.4);

glEndList();

}

void reshape(int w, int h){

glViewport(0, 0, (GLsizei)w, (GLsizei)h);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluPerspective(45.0, (GLdouble)w / (GLdouble)h, 3.0, 8.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

glTranslatef(0.0, 0.0, -5.0);

}

void keyboard(unsigned char key, int x, int y){

switch (key){

case 'x':

case 'X':

currentGenMode = GL\_EYE\_LINEAR;

currentPlane = GL\_EYE\_PLANE;

glTexGeni(GL\_S, GL\_TEXTURE\_GEN\_MODE, currentGenMode);

glTexGenfv(GL\_S, currentPlane, currentCoeff);

glutPostRedisplay();

break;

case 'w':

case 'W':

currentGenMode = GL\_OBJECT\_LINEAR;

currentPlane = GL\_OBJECT\_PLANE;

glTexGeni(GL\_S, GL\_TEXTURE\_GEN\_MODE, currentGenMode);

glTexGenfv(GL\_S, currentPlane, currentCoeff);

glutPostRedisplay();

break;

case 's':

case 'S':

currentCoeff = slanted;

glTexGenfv(GL\_S, currentPlane, currentCoeff);

glutPostRedisplay();

break;

case 'a':

case 'A':

currentCoeff = xequalzero;

glTexGenfv(GL\_S, currentPlane, currentCoeff);

glutPostRedisplay();

break;

case 'd':

case 'D':

if (fog == GL\_EXP){

fog = GL\_EXP2;

} else if (fog == GL\_EXP2){

fog = GL\_LINEAR;

} else if (fog == GL\_LINEAR) {

fog = GL\_EXP;

}

glFogi(GL\_FOG\_MODE, fog);

glutPostRedisplay();

break;

}

}

int main(int argc, char\*\* argv){

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB | GLUT\_DEPTH);

glutInitWindowSize(1000, 500);

glutCreateWindow("5 лаба");

init\_nurb();

init();

glutDisplayFunc(display);

glutReshapeFunc(reshape);

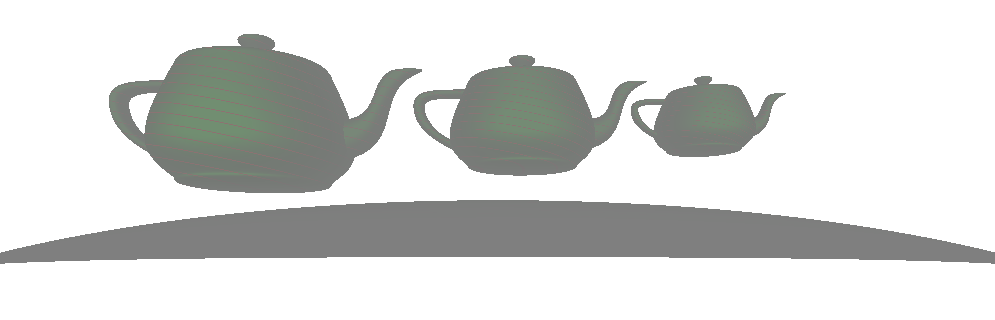
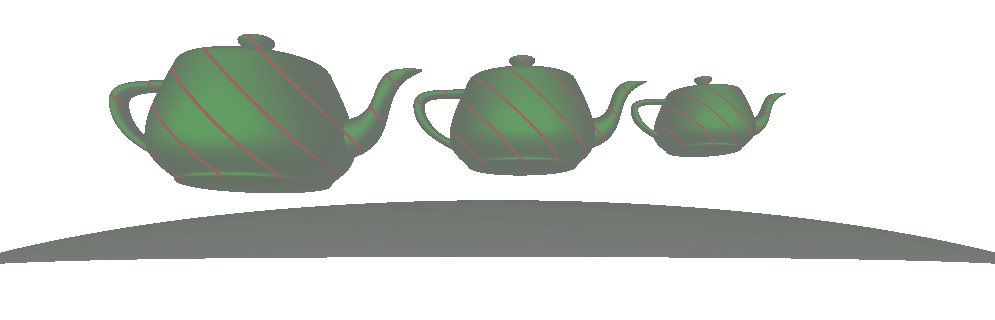
glutKeyboardFunc(keyboard);

glutMainLoop();

return 0;

}

**Пример выполнения программы:**

****

**Выводы:**

В данной лабораторной работе были получены практические навыки по работе со списками, текстурой и NUMBS-поверхностями