实验十四 曲线曲面生成实验

时间：2022年6月8日

地点：信息学院机房2202

1、实验内容

使用opengl，书写教材P323代码，运行Bezier曲线生成程序。

2、实验目的

采用Bezier函数验证曲线生成

3、实验代码

#include <windows.h>

#include <GL/glut.h>

#include <stdlib.h>

#include <math.h>

/\* Set initial size of the display window. \*/

GLsizei winWidth = 600, winHeight = 600;

/\* Set size of world-coordinate clipping window. \*/

GLfloat xwcMin = -50.0, xwcMax = 50.0;

GLfloat ywcMin = -50.0, ywcMax = 50.0;

class wcPt3D {

public:

GLfloat x, y, z;

};

void init (void)

{

/\* Set color of display window to white. \*/

glClearColor (1.0, 1.0, 1.0, 0.0);

}

void plotPoint (wcPt3D bezCurvePt)

{

glBegin (GL\_POINTS);

glVertex2f (bezCurvePt.x, bezCurvePt.y);

glEnd ( );

}

/\* Compute binomial coefficients C for given value of n. \*/

void binomialCoeffs (GLint n, GLint \* C)

{

GLint k, j;

for (k = 0; k <= n; k++) {

/\* Compute n!/(k!(n - k)!). \*/

C [k] = 1;

for (j = n; j >= k + 1; j--)

C [k] \*= j;

for (j = n - k; j >= 2; j--)

C [k] /= j;

}

}

void computeBezPt (GLfloat u, wcPt3D \* bezPt, GLint nCtrlPts,

wcPt3D \* ctrlPts, GLint \* C)

{

GLint k, n = nCtrlPts - 1;

GLfloat bezBlendFcn;

bezPt->x = bezPt->y = bezPt->z = 0.0;

/\* Compute blending functions and blend control points. \*/

for (k = 0; k < nCtrlPts; k++) {

bezBlendFcn = C [k] \* pow (u, k) \* pow (1 - u, n - k);

bezPt->x += ctrlPts [k].x \* bezBlendFcn;

bezPt->y += ctrlPts [k].y \* bezBlendFcn;

bezPt->z += ctrlPts [k].z \* bezBlendFcn;

}

}

void bezier (wcPt3D \* ctrlPts, GLint nCtrlPts, GLint nBezCurvePts)

{

wcPt3D bezCurvePt;

GLfloat u;

GLint \*C, k;

/\* Allocate space for binomial coefficients \*/

C = new GLint [nCtrlPts];

binomialCoeffs (nCtrlPts - 1, C);

for (k = 0; k <= nBezCurvePts; k++) {

u = GLfloat (k) / GLfloat (nBezCurvePts);

computeBezPt (u, &bezCurvePt, nCtrlPts, ctrlPts, C);

plotPoint (bezCurvePt);

}

delete [ ] C;

}

void displayFcn (void)

{

/\* Set example number of control points and number of

\* curve positions to be plotted along the Bezier curve.

\*/

GLint nCtrlPts = 4, nBezCurvePts = 1000;

wcPt3D ctrlPts [4] = { {-40.0, -40.0, 0.0}, {-10.0, 200.0, 0.0},

{10.0, -200.0, 0.0}, {40.0, 40.0, 0.0} };

glClear (GL\_COLOR\_BUFFER\_BIT); // Clear display window.

glPointSize (4);

glColor3f (1.0, 0.0, 0.0); // Set point color to red.

bezier (ctrlPts, nCtrlPts, nBezCurvePts);

glFlush ( );

}

void winReshapeFcn (GLint newWidth, GLint newHeight)

{

/\* Maintain an aspect ratio of 1.0. \*/

glViewport (0, 0, newHeight, newHeight);

glMatrixMode (GL\_PROJECTION);

glLoadIdentity ( );

gluOrtho2D (xwcMin, xwcMax, ywcMin, ywcMax);

glClear (GL\_COLOR\_BUFFER\_BIT);

}

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

{

glutInit (&argc, argv);

glutInitDisplayMode (GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowPosition (50, 50);

glutInitWindowSize (winWidth, winHeight);

glutCreateWindow ("Bezier Curve");

init ( );

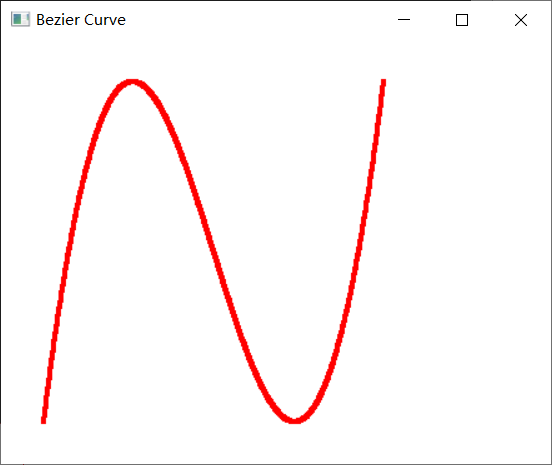
glutDisplayFunc (displayFcn);

glutReshapeFunc (winReshapeFcn);

glutMainLoop ( );

}

4、实验结果



5、实验总结

1.glClearColor：red、green、blue、alpha分别是红、绿、蓝、不透明度，值域均为[0,1]。即设置颜色，为后面的glClear做准备，默认值为（0,0,0,0）。切记：此函数仅仅设定颜色，并不执行清除工作。

2.gluOrtho2D：参数分别代表（左下角x坐标，右上角x坐标，左下角y坐标，右上角y坐标）—坐标相对于窗口左下角——原点）。