#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include<GL/glut.h>

#define PI 3.1416

GLsizei winWidth = 600, winHeight = 600;

GLfloat xwcMin = 0.0, xwcMax = 130.0;

GLfloat ywcMin = 0.0, ywcMax = 130.0;

int size,submenu;

GLint nCtrlPts = 4, nBezCurvePts =20;

static float theta = 0;

struct wcPt3D

{

GLfloat x;

GLfloat y;

GLfloat z;

} ;

wcPt3D ctrlPts[4] = {

{20, 100, 0},

{30, 110, 0},

{50, 90, 0},

{60, 100, 0}};

typedef struct wcPt3D cp;

void bino(GLint n, GLint \*C)

{

GLint k, j;

for(k=0;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, cp \*bezPt, GLint nCtrlPts, cp \*ctrlPts, GLint \*C)

{

GLint k, n=nCtrlPts-1;

GLfloat bezBlendFcn;

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

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(cp \*ctrlPts, GLint nCtrlPts, GLint nBezCurvePts)

{

cp bezCurvePt;

GLfloat u;

GLint \*C, k;

C= new GLint[nCtrlPts];

bino(nCtrlPts-1, C);

glBegin(GL\_LINE\_STRIP);

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

{

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

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

glVertex2f(bezCurvePt.x, bezCurvePt.y);

}

glEnd();

delete[]C;

}

void displayFcn()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3f(1.0, 1.0, 1.0);

glPointSize(5);

glPushMatrix();

glLineWidth(5);

glColor3f(255/255, 153/255.0, 51/255.0); //Indian flag: Orange color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glColor3f(1, 1, 1); //Indian flag: white color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glColor3f(19/255.0, 136/255.0, 8/255.0); //Indian flag: green color code

for(int i=0;i<8;i++)

{

glTranslatef(0, -0.8, 0);

bezier(ctrlPts, nCtrlPts, nBezCurvePts);

}

glPopMatrix();

glColor3f(0.7, 0.5,0.3);

glLineWidth(5);

glBegin(GL\_LINES);

glVertex2f(20,100);

glVertex2f(20,40);

glEnd();

glFlush();

glutPostRedisplay();

glutSwapBuffers();

}

void menufunc(int n)

{

switch(n)

{

case 1:

ctrlPts[1].x +=10\*sin(theta \* PI/180.0);

ctrlPts[1].y +=5\*sin(theta \* PI/180.0);

ctrlPts[2].x -= 10\*sin((theta+30) \* PI/180.0);

ctrlPts[2].y -= 10\*sin((theta+30) \* PI/180.0);

ctrlPts[3].x -= 4\*sin((theta) \* PI/180.0);

ctrlPts[3].y += sin((theta-30) \* PI/180.0);

theta+=0.1;

break;

case 2:

ctrlPts[1].x -=10\*sin(theta \* PI/180.0);

ctrlPts[1].y -=5\*sin(theta \* PI/180.0);

ctrlPts[2].x +=10\*sin((theta+30) \* PI/180.0);

ctrlPts[2].y +=10\*sin((theta+30) \* PI/180.0);

ctrlPts[3].x +=4\*sin((theta) \* PI/180.0);

ctrlPts[3].y -=sin((theta-30) \* PI/180.0);

theta-=0.1;

break;

case 3: exit(0);

}

//glutPostRedisplay();

}

void winReshapeFun(GLint newWidth, GLint newHeight)

{

glViewport(0, 0, newWidth, 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\_DOUBLE | GLUT\_RGB);

glutInitWindowPosition(50, 50);

glutInitWindowSize(winWidth, winHeight);

glutCreateWindow("Bezier Curve");

submenu=glutCreateMenu(menufunc);

glutCreateMenu(menufunc);

glutAddMenuEntry("Dwn-mov",1);

glutAddMenuEntry("Up-mov",2);

glutAddMenuEntry("Exit",3);

glutAttachMenu(GLUT\_RIGHT\_BUTTON);

glutDisplayFunc(displayFcn);

glutReshapeFunc(winReshapeFun);

glutMainLoop();

}