CGPractical7a

#include <GL/glut.h>

#include <iostream>

#include <cmath>

using namespace std;

class Point {

public:

int x, y;

void setxy(int \_x, int \_y) {

x = \_x;

y = \_y;

}

};

static int POINTSNUM = 0;

static Point points[4];

void init(void)

{

glClearColor(0.0, 0.0, 0.0, 0); // Background black

glColor3f(1.0, 1.0, 1.0); // Drawing color white

glPointSize(4.0); // Point size

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0.0, 600.0, 0.0, 480.0); // 2D Projection

}

void setPoint(Point p)

{

glBegin(GL\_POINTS);

glVertex2f(p.x, p.y);

glEnd();

glFlush();

}

void setLine(Point p1, Point p2)

{

glBegin(GL\_LINES);

glVertex2f(p1.x, p1.y);

glVertex2f(p2.x, p2.y);

glEnd();

glFlush();

}

Point setBezier(Point p1, Point p2, Point p3, Point p4, double t)

{

Point p;

double a1 = pow((1 - t), 3);

double a2 = 3 \* pow((1 - t), 2) \* t;

double a3 = 3 \* t \* t \* (1 - t);

double a4 = pow(t, 3);

p.x = a1 \* p1.x + a2 \* p2.x + a3 \* p3.x + a4 \* p4.x;

p.y = a1 \* p1.y + a2 \* p2.y + a3 \* p3.y + a4 \* p4.y;

return p;

}

void display()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glFlush();

}

void myMouseFunction(int button, int state, int x, int y)

{

if (state == GLUT\_DOWN)

{

points[POINTSNUM].setxy(x, 480 - y); // Flip y-coordinates for OpenGL

glColor3f(1.0, 0.0, 0.0); // Red for control points

setPoint(points[POINTSNUM]);

if (POINTSNUM > 0)

setLine(points[POINTSNUM - 1], points[POINTSNUM]);

if (POINTSNUM == 3)

{

glColor3f(1.0, 1.0, 0.0); // Yellow Bezier Curve

Point current = points[0];

for (double t = 0.0; t <= 1.0; t += 0.01)

{

Point next = setBezier(points[0], points[1], points[2], points[3], t);

setLine(current, next);

current = next;

}

POINTSNUM = 0; // Reset for next curve

}

else

{

POINTSNUM++;

}

}

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_RGB | GLUT\_SINGLE);

glutInitWindowSize(600, 480);

glutInitWindowPosition(100, 100);

glutCreateWindow("Bezier Curve - Practical 7a");

init();

glutMouseFunc(myMouseFunction);

glutDisplayFunc(display);

glutMainLoop();

return 0;

}